Berner
Fachhochschule

Analyse von Scans im Internet

Master Thesis

Studiengang	MAS Cyber Security
Autor	Mauro Guadagnini
Themensponsoring	Berner Fachhochschule TI Cyber Security Lab
Lead-Experte	Prof. Hansjürg Wenger
Co-Experten	Prof. Dr. Bruce Nikkel
	Prof. Rolf Lanz
Datum	7. März 2025

Abstract

Zugangspunkte im Internet erfahren dauerhaft Anfragen (nachfolgend „Scans“ genannt) in Formen wie Ping-Anfragen oder Port-Scans. Diese dienen unter anderem der Aufklärung und werden von diversen Quellen ausgeführt. Das Ziel dieser Arbeit liegt darin, den Ursprung und die Absicht der Scans zu ermitteln, inklusive der Menge an bösartigen Scans. Zuletzt bekannte Erkenntnisse zu diesem Thema finden sich in einer Arbeit von 2018.

Zur Ermittlung wird eine nachvollziehbare Analyse-Umgebung aufgebaut, die folgendes beinhaltet: Das Software-Produkt „Malcolm“ zur zentralen Auswertung und global platzierte Server (Scan-Ziele), die ihren Netzwerkverkehr zu ersterem spiegeln. Die Auswertungsfunktionalität wird anhand eigener Skripts und automatisiert angereicherten Tabellen erweitert. Diese beinhalten Merkmale bekannter Scan-Quellen zur Identifikation. Anhand aufgezeichneter Verbindungen werden Scans mit entsprechenden Quellen oder Inhalten detektiert und mit einer Intention („good“ oder „bad“) versehen. Auswertungen sind anhand von Malcolm oder selbst erstellten Übersichten und Visualisierungen dynamisch einseh- und filterbar.

Untersuchungen ergeben, dass im Durchschnitt ungefähr 5 % der detektierten Scans mit Intention „bad“ ausfallen. Hauptsächlich handelt es sich hierbei um Quellen, deren IP-Adresse in Verbindung mit bösartigen Aktivitäten stehen (Indicators of Compromise, IoC). Scans, die sich durch ihre Anwendung oder Quelle entsprechend ausweisen („good“), stammen aus diversen Scan-Institutionen wie Censys oder Shodan sowie der Applikation „ZMap“. Der Anteil an erfahrenen IPv6-Verbindungen beträgt im Vergleich zu IPv4 0,007 %. Diese Ergebnisse haben einen Einfluss auf die Erwartungen zum eintreffenden Netzwerkverkehr auf öffentlichen IP-Adressen. Die hierbei aufgebaute Analyse-Umgebung wird nach dieser Arbeit weiter betrieben und bietet zusätzliche Auswertungs- sowie Erweiterungsmöglichkeiten.

Titelbild: *Fingerprint scanning on circuit board. secure system concept with a fingerprint. Cyber security technology concept abstract background futuristic Hi-tech style. Vector and Illustration* (S and V Design 2024) [1]
LaTeX Vorlage: *BFH-CI – Corporate Design LaTeX Templates for Bern University of Applied Sciences* (Peischl und Habegger 2024) [2]
Kreisförmige Symbole sowie Filter- und Handzeichen-Symbole: *The fontawesome5 package* (Font Awesome und Krüger 2022) [3]
Flaggen: *worldflags – Drawing flags with TikZ* (Haager 2023) [4], Ausnahme Hong Kong: *Flag of Hong Kong* (Rohsopht 2021) [5]

Inhalt

1. Einleitung

Dieses Kapitel beschreibt die Ausgangslage sowie Zielsetzung dieser Arbeit. Im Grundlagen-Kapitel sind der Stand der Forschung und potenzielle Identifikationsmöglichkeiten von Scan-Quellen aufgeführt. Das methodische Vorgehen wird nachvollziehbar im entsprechenden Methoden-Kapitel dokumentiert. Daraus resultierende Informationen sind im Kapitel „Ergebnisse“ vorzufinden. Mit dem Abschluss-Kapitel wird diese Arbeit abgerundet.

Nachfolgende Kapitel beinhalten entsprechende Verzeichnisse inklusive Glossar (siehe ab Seite 68) sowie Anhänge mit weiteren Details. Das Anhangsverzeichnis befindet sich auf Seite 102.

Zielgruppe dieser Arbeit sind Personen mit Kenntnissen in Netzwerkkommunikation inklusive den Protokollen IPv4 [6], IPv6 [7], ICMP [8], ICMPv6 [9], TCP [10] und UDP [11].

1.1. Ausgangslage

Zugangspunkte im Internet erfahren dauerhaft Untersuchungen („Scans“) von fremden Quellen in Formen wie Ping-Anfragen und Port-Scans [12–14]. Mittels Ping-Anfragen kann die Erreichbarkeit einer Adresse im Internet geprüft werden. Bei Port-Scans wird dasselbe auf spezifische Ports einer Adresse unternommen, auf welchen Serverdienste wie Webserver oder Datenbanken zur Verfügung gestellt werden.

Diese Untersuchungen können alle mit einem Internetzugang durchführen, um zum Beispiel verwundbare Dienste für Angriffe oder neue Suchmaschinen-Einträge aufzudecken. Beteiligte, die Internetweite Scans öffentlich oder rechtmässig durchführen, verwenden unter anderem nachvollziehbare IP-Adressen oder weisen sich per Inhalt beziehungsweise der „Payload“ eines Scan-Netzwerkpakets aus [15–18]. Darunter finden sich bekannte Organisationen wie die Suchmaschinen Shodan [19] und Censys [20] oder die Shadowserver-Stiftung [21], das National Cyber Security Centre (NCSC) aus dem Vereinigten Königreich [17] sowie Universitäten und diverse Scan-Projekte [16, 22]. Fallen die Absichten bei einem Scan bösartig aus, kann sich solch eine Scan-Quelle nicht oder mit Fehlinformationen ausweisen [15, 16, 23].

Eine Auswertung von Heo und Shin aus 2018 zeigt, dass ungefähr 5 % der Scan-Quellen sich öffentlich ausweisen [22]. Diese Ausweisung erfolgte per Webseite unter der Quell-IP-Adresse, dem zugehörigen Domain-Namen oder über die Bezeichnung des entsprechenden Quell-AS [22]. Den grössten Anteil dieser Quellen bildeten Shodan und Shadowserver [22]. Weitere Recherchen lassen eine solche Betrachtung der Scan-Quellen aussen vor und/oder nennen lediglich den geografischen Standort (Länder) [13, 14, 24, 25].

Mit RFC 9511 [15] aus 2023 werden Techniken zur Identifizierung der Scan-Quellen unter anderem auf Layer 3 vorgeschlagen [15]. Der Artikel schlägt vor, eine „Probe Description URI (Uniform Resource Identifier)“ zu hinterlegen, die auf eine Textdatei mit dem Beschrieb entsprechend durchgeführter Messungen zeigt [15]. Die URI soll u. a. über den rückwärts aufgelösten Domain-Namen der Scan-Quelle oder direkt über dessen IP-Adresse aufgerufen werden können [15]. Als Alternative wird vorgeschlagen, Informationen direkt im Netzwerkpaket des Scans zu hinterlegen [15].

Der aktuelle Stand der Forschung ist in Kapitel 2.1 ausführlicher beschrieben. Aussagen zur Menge an bösartigen Scans im Internet konnten nicht ermittelt werden. Dies gilt ebenfalls für Aussagen nach 2018 zu sich ausweisenden Scan-Quellen.

Themensponsoring und Auftraggebende dieser Arbeit bilden das TI Cyber Security Lab der Berner Fachhochschule [26], an dem auch Experten dieser Arbeit (Prof. Hansjürg Wenger und Prof. Dr. Bruce Nikkel) beteiligt sind.

1.2. Zielsetzung

Der Zweck dieser Forschungsarbeit bildet die Untersuchung, aus welchem Ursprung die Scans im Internet stammen und mit welcher Absicht sie durchgeführt werden. Hierzu gehört das geografische Auftreten der Scans, gescannte Dienste/Ports, mögliche Identifikatoren oder Verhaltensmuster sowie die Ermittlung möglicher Scan-Absichten. So entsteht eine Übersicht der zu erwartenden Quellen und deren potenziellen Absichten beim Anschluss eines Zugangspunktes beispielsweise in Form eines Servers am Internet.

Die zu beantwortende Hauptfrage liegt in der Klassierung der Scan-Quellen im Internet zum aktuellen Zeitpunkt, um unter anderem die Menge an bösartigen Scans aufzuzeigen. Die Basis bildet eine praktische Analyse mit dem Aufbau einer nachvollziehbaren und begründeten Analyse-Umgebung. Diese Umgebung beinhaltet global verteilte Server, um eine möglichst weltumfassende Ansicht zu erzielen. Die Hypothese zu Beginn der Arbeit ist u. a. aufgrund der Arbeit von Heo und Shin [22], dass unter 10 % der Scan-Quellen sich öffentlich mit ihren Absichten erkenntlich zeigen und die Menge an bösartigen Scans mit einer bestimmten Fehlertoleranz ausfallen wird.

Die Server dieser Analyse-Umgebung, nachfolgend auch Scan-Ziele oder Scan-Targets genannt, werden zur Einhaltung der gegebenen Zeit[1] auf einem niedrigen Interaktions-Level betrieben. Dies bedeutet, dass sie bei Ping-Anfragen auf dessen IPv4- und IPv6-Adressen antworten, aber keine TCP- oder UDP-Ports zum Internet geöffnet haben. DNS-Domain-Einträge für die Scan-Ziele haben keine Priorität und werden nur bei genügend zur Verfügung stehender Zeit betrachtet. Jede Anfrage auf eine öffentliche IP-Adresse eines Scan-Ziels wird als Scan gewertet. Die Aufzeichnung des Netzwerkverkehrs der Analyse-Umgebung findet auf einem zentralen Server statt, an welchen der Netzwerkverkehr der Scan-Ziele gespiegelt wird (Port-Mirroring). Dies bedeutet, dass nur einzelne Ziele mit aktiven Scan-Anfragen und keine IP-Adressbereiche, zentral geschaltete Firewalls oder Netzwerkteleskope betrachtet werden.

Aus Zeitgründen kann die Analyse-Umgebung nur eine bestimmte Anzahl an Server umfassen. Die Wahl der Dienstleistungsunternehmen zur Platzierung der Scan-Ziele wird möglichst repräsentativ für eine globale Abdeckung gewählt, hat jedoch sicherlich einen Einfluss auf die daraus ermittelten Ergebnisse. Entfällt beispielsweise die Platzierung in einem Land, das als Ziel von Angriffen anderer Staaten gilt, entfallen auch entsprechende Aufzeichnungen.

Der Anschluss der Scan-Ziele ans Internet soll möglichst ohne davor geschaltete Schutzmechanismen erfolgen, jedoch können diese nicht komplett ausgeschlossen werden. Dies da von Hosting-Dienstleistungsunternehmen ein Grundschutz ihrer Infrastruktur inklusive Verwerfen bestimmter Anfragen erwartet werden kann.

Es gilt die Analyse-Umgebung sowie den Auswertungsvorgang nachvollziehbar und erweiterbar aufzubauen, um gegebenenfalls zu einem späteren Zeitpunkt einen Vergleich zu den hier ermittelten Informationen ziehen zu können.

Ein ausführlicher Beschrieb der Arbeitspakete, die Zeitplanung sowie Muss- und Kann-Ziele sind im Anhang in Kapitel A.1 definiert.

Personen und Organisationen mit Kontakt zu Komponenten mit öffentlichen IPv4- und/oder IPv6-Adressen erfahren Informationen zu Scans und deren Ausprägungen, um beispielsweise Schutzvorkehrungen entsprechend priorisieren zu können.

[1] Ein Semester (Herbstsemester 2024) steht für diese Arbeit zur Verfügung. Das Arbeitsjournal ist im Anhang in Kapitel A.2 aufgeführt.

1.3. Lieferobjekte

Das Ergebnis dieser Forschungsarbeit ist ein Bericht inklusive Recherche mit empirisch erarbeiteten Erkenntnissen und einem dokumentierten Aufbau der Analyse-Umgebung sowie des Auswertungsvorgangs.

Die zugehörige Installationsdokumentation wird im Anhang in Kapitel C festgehalten.
Erstellter Code wird dem Auftraggeber übergeben und ist zusätzlich öffentlich einsehbar unter `https://guadm.github.io`.

Aus Übersichtsgründen wird der Code nicht komplett, jedoch mit für Aussagen relevanten Ausschnitten dieser Arbeit angehängt (siehe Kapitel D). Die ermittelten Daten werden aus demselben Grund nicht komplett aufgeführt (nur für Aussagen relevante Objekte).

Somit entsteht eine Basis, auf welcher weitere Forschungen betrieben werden können. Hierbei kann die Analyse-Umgebung vergrössert oder dessen Funktionalität erweitert ausfallen. Diese Umgebung wird am Ende der Arbeit ebenfalls dem Auftraggeber übergeben, kann aber anhand der Dokumentation und dem Code nachvollziehbar reproduziert werden. Bezüglich aufgebauter Scan-Ziele wird lediglich das bereits beim Auftraggeber implementierte übergeben.

Übersicht Lieferobjekte

- Bericht / Thesis mit Recherche, Erkenntnissen und Installationsdokumentation
  Anhänge siehe Anhangsverzeichnis auf Seite 102
- Code zur Auswertung von Netzwerkverkehr und Aufbau der Analyse-Umgebung
  Ausschnitte in Thesis oder dessen Anhang enthalten, öffentlich unter `https://guadm.github.io` einsehbar
  Siehe auch Kapitel D im Anhang
- Analyse-Umgebung Infrastruktur
  Übergabe an Auftraggeber (siehe vorbereitetes Protokoll in Kapitel A.4.10)
  Reproduktion mit Thesis und Code möglich

2. Grundlagen

Dieses Kapitel erläutert die Grundlagen dieser Thesis. Nachfolgend sind in einzelnen untergeordneten Kapiteln der Stand der Forschung sowie die potenziellen Identifikationsmöglichkeiten von Scan-Quellen aufgeführt.

2.1. Stand der Forschung

Eine Vielzahl an ermittelten Forschungsarbeiten betrachten unterschiedliche Aspekte von Internet-weiten Scans. Diese beinhalten unter anderem die Ursprungsländer der Scans [13, 14, 22, 27, 28] oder deren anvisierte Serverdienste bzw. Port-Adressen [14, 22, 25, 27, 29]. Weitere Arbeiten betrachten spezifische Techniken, Quellen oder Anwendungen für Internet-weite Scans [12, 16, 30–33] sowie die Anwendung von IPv6 [29, 34–36].

Recherchen zum Stand der Forschung führen zur Identifikation einer Arbeit von Heo und Shin aus 2018, deren Aussagen die Erkenntnisse dieser Arbeit indizieren können. Hierbei haben Heo und Shin die Verbindungsinformationen zweier Firewalls ihrer Universität in Südkorea von Juni bis Juli 2016 ausgewertet [22]. Die entsprechende Autorschaft bezeichnet ungefähr 5 % der Scan-Quellen als „verantwortungsvoll“ beziehungsweise weisen sich diese 5 % öffentlich aus (per Webseite unter der entsprechenden IP-Adresse, Domain-Namen oder Besitzer-Bezeichnung eines dazugehörigen AS) [22]. Zusätzlich nennen Heo und Shin die entsprechenden Quellen beim Namen, wobei Shodan und Shadowserver den grössten Anteil davon aufzeigen [22]. Kommunikationen auf Basis von IPv6 haben Heo und Shin nicht betrachtet [22].

Aussagen, die den Zielaussagen dieser Arbeit entsprechen (siehe Kapitel 1.2) und innerhalb der letzten 4 Jahre getroffen wurden, können nicht ermittelt werden.

2.1.1. Geografische Lage

Die geografische Lage der Scan-Quellen und deren Netzwerkaufbau sowie die Stabilität der Verbindungen haben einen Einfluss auf die daraus resultierenden Ergebnisse [24, 28]. 2019 und 2020 haben Wan et al. Internet-weite Scans auf Basis von IPv4 mit den Anwendungen ZMap [37] und ZGrab [37] zu den Protokollen HTTP, HTTPS und SSH durchgeführt [28]. Hierbei stammten die Scans aus unter anderem Censys [20] und akademischen Institutionen in Australien, Brazilien, Deutschland, Japan und den USA [28]. Individuelle Quellen verfehlten hierbei 1,6-8,4 % der global ermittelten HTTP-, 1,5-4,6 % der HTTPS- und 8,3-18,2 % der SSH-Hosts abzudecken [28].

Port-Scans werden global ausgeführt [12–14]. Geografische Einschränkungen diverser Länder üben ebenfalls einen Einfluss auf die Erreichbarkeit der Ziele aus [14, 28]. Hierbei kommt es vor, dass bestimmter Netzwerkverkehr nur innerhalb des eigenen Landes erlaubt wird oder die Kommunikation aus bestimmten Ländern geblockt wird [14, 28]. Wurde eine öffentliche IP-Adresse schon mal für Scans genutzt, kann diese bereits auf Blocklisten eingetragen sein [28]. Ein solcher Eintrag kann zur Folge haben, dass zukünftige Scans von derselben IP-Adresse nicht ihr Ziel erreichen [28]. Es wird beobachtet, dass Censys aufgrund dessen verhältnismässig hoher Anzahl an Scans von diversen Infrastrukturen geblockt wird [28]. Internet- oder Hosting-Dienstleistungsunternehmen, aber auch akademische Institutionen und Firmen blockieren Scan-Verkehr [14]. Dies wird besonders bei Organisationen beobachtet, die ganze AS kontrollieren können [14].

Im Bezug zu Scan-Ursprungsländern beobachtet Wilhoit 2013 u. a. folgende Länder als Quellen von Angriffen auf dessen, in den USA platzierte Honeypot-Infrastruktur: Russland, China, Deutschland, USA und Palästina [24].

Durumeric et al. sehen 2014 in deren Arbeit bei der Analyse von Netzwerkverkehr aus einem grossen Darknet vermehrt Scans aus China, den USA, Deutschland, den Niederlanden und Russland [14]. Richter und Berger nennen 2019 bei deren Untersuchung als Länder mit den meisten Scan-Quell-IP-Adressen die Ukraine, die USA, Niederlanden, China und das Vereinigte Königreich [25].

Serbanescu et al. treffen 2015 in deren Amazon-Cloud-Umgebung mit mehreren, geografisch verteilen Servern einen Grossteil von Verbindungen aus China und Spanien an [13]. Shodan wird hierbei ausgeklammert [13].

Heo und Shin sehen auf den Firewalls deren Universität in Südkorea im Jahr 2016 die meisten Scans-Quellen aus China, den USA, Brasilien, Taiwan und Vietnam [22].

2.1.2. IPv6

Aufgrund der hohen Anzahl möglicher IPv6-Adressen sind Internet-weite Scans mit IPv6 anders ausgeprägt als ihre IPv4-Variante [14, 29, 34–36]. Scans mit IPv6 können von Quellen kommen, die dabei eine grosse Anzahl von unterschiedlichen Quell-Adressen im selben Präfix verwenden [29]. Einzelne IPv6-Quelladressen weisen dann eine geringe Anzahl an Netzwerkpaketen auf [29]. Internet-weite Scans mit IPv6 scheinen gemäss Richter et al. in 2022 meist von High-Performance-Rechenzentren und Cloud-Dienstleistungsunternehmen zu stammen (IPv4-Scans hingegen von diversen Netzwerk-Konstellationen oder Botnets) [29].

IPv6-Zieladressen werden über DNS-Einträge oder sogenannte Hitlisten ermittelt [29, 34, 36, 38]. Hitlisten mit IPv6-Adressen werden u. a. generiert aufgrund deren Vorkommen im beobachteten Netzwerkverkehr, Einträgen bei Reverse DNS (rDNS) Zonen oder Traceroute-Messungen (beinhalten Router-IP-Adressen) [34–36]. Zufällig ermittelte, nahe bei bekannten liegende oder bestimmten Mustern/Begriffen[2] entsprechende IPv6-Adressen bilden zusätzliche Ziele [29, 34, 38]. Eine Vielzahl von IPv6-Zielen wird anhand von DNS-Einträgen ermittelt [29]. Nicht jede Organisation veröffentlicht ihre DNS-Zonen-Daten, was Einfluss auf die ermittelten IPv6-Ziele ausübt [34, 35]. Murdock et al. präsentieren mit „6Gen“ einen Algorithmus zur Ermittlung von ansprechbaren IPv6-Adressen (Target Generation Algorithm, TGA) [34].

Der von Gasser et al. erstellten IPv6-Hitlist-Service kann über folgende Adresse beantragt werden: `https://ipv6hitlist.github.io` [36, 39]. Dienste zur Einsicht von Standortinformationen zu IPv4- und IPv6-Adressen werden von diversen Dienstleistungsunternehmen angeboten [40–43].

Ein Grossteil der ermittelten Arbeiten klammern IPv6 aus und betrachten nur den IPv4-Adressraum [13, 14, 16, 22, 25, 28, 31].

2.1.3. Scan-Methoden und -Anwendungen

Ein Scan kann in diversen Ausführungen und auf unterschiedlichen Kommunikationsschichten (Layer) stattfinden. Beispielsweise können bei Port-Scans mittels TCP dessen initiale Netzwerkpakete ein bestimmtes Flag gesetzt haben (u. a. TCP SYN Scan, TCP ACK Scan oder TCP FIN Scan) [28, 32, 33, 44, 45]. Je nach TCP-Scan-Typ kann ein anderes Verhalten erzielt werden [44]. UDP-Scans sind ebenfalls im Internet anzutreffen, jedoch aufgrund dessen Protokoll nur mit Variationen entsprechend dessen Payload bzw. Paketinhalt [33, 44]. Anwendungsspezifische Scans wie ein HTTP-GET-Request, TLS- oder SSH-Handshake prüfen nicht nur die Erreichbarkeit eines Ports sondern auch die dahinterliegende Applikation [28, 33]. Ping-Anfragen mittels ICMP („Echo Requests“ [8, 9]) dienen ebenfalls zum Prüfen der Erreichbarkeit unter einer IP-Adresse, wobei von dieser auf eine solche Anfrage ein „Echo Reply“ [8, 9] als Antwort versendet werden kann [29, 33].

[2] Die Verwendung von hexadezimalen Ziffern in einer IPv6-Adresse ermöglicht das Bilden von visuell lesbaren Wörtern wie zum Beispiel `DEADBEEF` [7, 34]

Fällt bei einer Scan-Anfrage eine Antwort aus, kann dies auch bedeuten, dass eine Sicherheitskomponente wie eine Firewall oder ein Intrusion Prevention System (IPS) die Anfrage blockiert [12, 44]. Somit kann sich hinter dem Ziel einer Anfrage ohne Antwort dennoch etwas befinden.

Bezüglich dem Scan-Verhalten sind diese je nach Absicht unterschiedlich ausgeprägt [12]. Anfragen können unter anderem zeitlich verteilt (langsam ausgeführt) oder manipuliert (z.B. Paketfragmentierung oder Header-/Paketmodifikation) werden, um weniger aufzufallen oder Fehlverhalten in Sicherheitskomponenten auszunutzen [12, 44]. Sie können auf einzelne oder mehrere Ports ausgerichtet sein und von einzelnen oder unterschiedlichen Scan-Quellen stammen (z.B. von Botnets) [12, 22, 29, 33, 46]. Je nach Scan-Ausprägung wird das dahinterliegende Betriebssystem oder eine entsprechende Applikation gegebenenfalls mit Versionsnummer ermittelt (Banner Grabbing [47]) [33].

Die IP-Adressen und Ports von Komponenten im Internet werden konstant analysiert [12, 14, 30]. Sobald eine Adresse (IP oder IP mit TCP-/UDP-Port) in einer Suchmaschine wie Shodan oder Censys aufgeführt ist, wird diese häufiger oder intensiver gescannt [13, 31, 48]. Bennett et al. beobachten, dass zu Komponenten mit geschlossenen Ports weniger Netzwerkverkehr generiert wird als zu welchen mit offenen Ports [31].

Recherchen ergeben eine Vielzahl an Möglichkeiten und Anwendungen zum Durchführen von Scans. Hierbei können diese mit anwendungsspezifischen Client-Applikationen durchgeführt werden wie Webbrowser oder Mail-/SSH-/VPN-Clients. Produkte zur Durchführung weitflächiger Scans im Internet bilden u. a. ZMap und ZGrab [16, 27, 28, 34] sowie masscan [49] [16, 34].

Tundis et al. nennen 2018 unter den automatischen Netzwerk-Schwachstellen-Scan-Werkzeugen mit öffentlich einsehbaren Resultaten folgende Beispiele: Shodan, Censys, Thingful[3] [50], Punkspider[4] [51] und Zoomeye (Suchmaschine ähnlich wie Shodan oder Censys) [53] [30]. In einer weiteren Kategorie mit persönlichen, interaktions-basierenden Schwachstellen-Scannern werden Nessus [54], skipfish [55], Acunetix [56], IVRE [57], Vulners [58] und Vega [59] genannt [30]. Zusätzlich nennen Tundis et al. weitere Scan-Tools mit diversen Ausprägungen und Zielen wie OpenVAS [60], Wireshark [61], Nikto [62], Angry IP Scanner [63], Advanced IP Scanner [64], Nexpose [65] oder das Metasploit Framework [66] [30]. Ebenfalls wird Qualys FreeScan genannt, welches 2021 sein End-Of-Life-Datum erreicht hat und durch Qualys Community Edition ersetzt wurde [67, 68]. Der Quellcode von Vega wurde zuletzt 2016 öffentlich angepasst [69]. Dies weist darauf hin, dass sich die entsprechende Produktauswahl fortlaufend ändern kann.

Nmap [44] ist in dessen Verwendung sehr verbreitet und wird häufig für Scans eingesetzt [12, 16, 70]. Für Nmap gibt es zusätzlich eine grafische Oberfläche namens Zenmap [44].

Durumeric et al. stellen fest, dass die Scan-Aktivitäten im Internet sich zwischen 2004 und 2014 signifikant verändert haben [14]. Sie bestätigen, dass 2019 Scan-Verkehr aus AS von Bulletproof Hosting-Infrastrukturen die Mehrheit bildet und horizontale Scans (denselben Port auf mehreren Zielen prüfen) üblich wurden [14]. Richter und Berger zeigen beim Zählen der Quell-IP-Adressen jedoch im Vergleich dazu ein anderes Bild: Dann bilden Internet-Service-Provider die Mehrheit, was auf infizierte Endgeräte als Quellen hindeuten lässt [25].

[3] Thingful war eine Suchmaschine für das „Internet of Things“ und wurde 2022 eingestellt [50]

[4] Punkspider durchsucht das Internet nach Schwachstellen bei Webseiten und präsentiert die Resultate in einer eigenen Webbrowser-Erweiterung [51, 52]

2.1.4. Identifizierung von Scan-Quellen

Über eine IP-Adresse können diverse Informationen ermittelt werden, wie deren Zuweisung bei der entsprechenden Internet-Registrierungsstelle [71]. Eine solche Stelle ist die RIPE, die für die Zuweisung von IP-Adressen im Raum Europa, dem Mittleren Osten und Zentralasien zuständig ist [71]. Die Datenbank der RIPE ist öffentlich durchsuchbar und enthält Kontaktinformationen wie Standortdaten, E-Mail-Adressen oder Telefonnummern zu einer entsprechenden IP-Adresse oder ASN („Whois"-Verzeichnis) [72]. Dies wird auch von anderen Registrierungsstellen, z.B. der APNIC für die Asien-Pazifik-Region angeboten [71, 73].

Diese Datenbanken ermöglichen abfragbare IP-Geolokalisierungsdienste, die jedoch keine hundertprozentige Genauigkeit gewährleisten [43]. Ähnliche Informationen können anhand eines Domain-Namens bei entsprechenden Stellen abgefragt werden [74]. Abhängig von Domäne und Registrierungsstelle können zum Schutz der Privatsphäre die Besitzerinnen und Besitzer einer Domäne anstelle ihren Daten im Whois-Eintrag die Angaben der Registrierungsstelle eintragen lassen („Whois Privacy", „Domain Privacy" oder auch „Domain Shield" genannt) [75–78].

Beispielsweise bei einem unidirektionalen Angriff (wie einem Denial-of-Service in Form von SYN-Flooding) kann eine Quell-IP-Adresse gefälscht sein [23]. In diesem Fall kann aus der entsprechend aufgeführten Adresse nicht direkt auf die entsprechende Quelle geschlossen werden [23]. Bei einem Angriff mit Datenabfluss muss das Ziel der Daten angegeben werden, was Informationen zu den Beteiligten oder dem Angriff preisgibt [23].

Scans können Informationen enthalten, die bestimmten Indicators of Compromise (IoC) entsprechen und auf bösartige oder kompromittierte Quellen schliessen [15, 79].

Öffentliche Organisationen oder Institutionen wie das NCSC UK [17], Shodan, Censys oder Projekte wie RIPE Atlas [80] weisen mindestens aus, dass sie Scans durchführen (teilweise werden auch verwendete IP-Adressen, Domänen oder Netzwerkpaketinhalte offengelegt) [17, 19, 81].

Bodenheim et al. stellen 2014 fest, dass Shodan dessen Geräte mit einer hohen Wahrscheinlichkeit nach ein paar Wochen in dessen Datenbank zur Verfügung stellt [48]. Viele Angriffsziele werden über das Abfragen von Shodan ermittelt [24, 82]. Bennett et al. beobachten 2021 spezifisch Censys und Shodan. Hierbei stellen sie fest, dass Censys-Anfragen nur aus den USA stammen und Anfragen von Shodan global verteilt und näher am Zielobjekt ausgeführt werden [31].

Die Anwendung ZMap vermerkt im IPv4-Identifikations-Feld [83] den Wert `54321` [14, 27, 84]. Abspaltungen bzw. „Forks" von ZMap entfernen dieses Identifikationsmerkmal [27].

Trapickin listet in dessen Recherche-Arbeit folgende, öffentlich bekannte Quellen von Internet-weiten Scans: CAIDA [85, 86], Universität von Michigan (USA) [87], Project Sonar [87], Shodan, Open Resolver Projekte [88–90] sowie zugehörige, vergangene Ereignisse und Projekte (u. a. Internet Census 2012 [91] oder das Project HACIENDA der NSA und GCHQ [92]) [16].

Trapickin kategorisiert die genannten Beteiligten in einem eigens vorgeschlagenen Modell [16]:

- Organisation / Institution
  - Akademisch
  - Staatlich
  - Kommerziell
  - Individuen
- Absicht
  - Schwachstellen-Ausnutzung
  - Forschung / Audit
  - Erkundung
- Verteilung der Scan-Quellen
  - Lokal
  - Verteilt
- Resultate einsehbar
  - Unveröffentlicht
  - Zusammengefasst
  - Öffentlich
- Verwendete Software öffentlich verfügbar
  - Ja
  - Nein
- Durchführungszeitpunkt
  - Vergangen
  - Laufend

Collins verwaltet in einem eigenen Repository auf GitLab eine Übersicht mit IP-Adressen von bekannten Scan-Quellen im Internet [18]. Darunter sind auch zuvor genannte Organisationen vorzufinden. Weitere Listen mit bekannten Scannern oder Adressen mit Beziehung zur Handhabung von Malware sind im Internet anzutreffen. Darunter sind zum Beispiel das IP-Adress-Repository von ShadowWhisperer [93] oder IoC-Listen von ThreatFox (abuse.ch) [94].

ONYPHE, die selber Scans im Internet betreiben, listen auf ihrer Webseite „10 Gebote zu ethischen Scans im Internet“ auf [95]. Diese beinhalten u. a. das Betreiben eines Web-Serverdienstes mit einem Scan-Beschrieb inklusive verwendeter IP-Adressen auf jeder Scan-Quelle [95]. Zur Erreichbarkeit listet ONYPHE auf, E-Mail-Adressen für Scan-Exklusions-Anfragen bereitzustellen sowie Organisations- und Missbrauch-Adressen („abuse“) in den Whois-Einträgen zu führen [95]. Rückwärts aufgelöste DNS-Einträge sollen auf das Projekt oder die Organisation verweisen [95]. Zusätzlich soll ein ethischer Scan im Internet von fixen IP-Adressen stammen, Standard-Pakete/-Protokolle verwenden und langsam ausgeführt werden [95].

Ein Vorschlag, mit der sich Scan-Quellen ausweisen können, existiert mit RFC 9511 [15]. Dieser wird im nächsten Kapitel genauer erläutert.

2.1.5. RFC 9511

Mit RFC 9511 „Attribution of Internet Probes“ (Status „Informational“) wird eine Technik zur Identifizierung der Scan-Quellen („Probes“) und dessen Scans vorgeschlagen [15].

Der Artikel schlägt vor, eine „Probe Description URI“ zu hinterlegen, beispielsweise `https://example.net/.well-known/probing.txt`.

Die URI zeigt auf eine Textdatei, die entsprechend durchgeführte Messungen beschreibt (siehe Quelltext 2.1)[5] [15]. Die URI soll über den rückwärts aufgelösten FQDN der Scan-Quelle oder auch direkt über dessen IP-Adresse aufgerufen werden können [15]. Weiterhin kann die „Probe Description URI“ in einem Scan mitgeführt werden, wie z.B. im Datenfeld einer ICMP-Echo-Request-Anfrage, was jedoch das Verhalten des Scans beeinflussen kann [15].

```
# Canonical URI (if any)
Canonical: https://example.net/measurement.txt

# Contact address
Contact: mailto:lab@example.net

# Validity
Expires: 2023-12-31T18:37:07z

# Languages
Preferred-Languages: en, es, fr

# Probes description
Description: This is a one-line string description of the probes.
```

Quelltext 2.1: Beispiel einer `probing.txt`-Datei entsprechend RFC 9511 [15]

Der RFC 9511 Artikel weist darauf hin, dass der hiermit extrahierten Information nicht blind vertraut werden kann, diese jedoch einen Hinweis auf eine Scan-Quelle geben könnte [15]. Eine absichtlich platzierte Fehlinformation ist nicht auszuschliessen [15].

Die `probing.txt` aus RFC 9511 ist seit dem 1. September 2023 bei der IANA als „Well-Known-URI“ registriert [97]. Dies ergibt Anlass zur Vermutung, dass dessen Verwendung zum aktuellen Zeitpunkt noch keine grosse Verbreitung erreicht hat[6].

[5] Das Format entspricht der Textdatei `security.txt` aus RFC 9116 „A File Format to Aid in Security Vulnerability Disclosure“ [96]
[6] Analog zur Textdatei `security.txt` aus RFC 9116 [96], die seit 2021 bei der IANA registriert ist [97–99]

2.2. Identifikationsmöglichkeiten

Entsprechend RFC 9424 „Indicators of Compromise (IoCs) and Their Role in Attack Defence“ existieren unter anderem folgende IoCs, anhand welcher Scan-Quellen gegebenenfalls identifiziert werden können [100]:

- IPv4- und IPv6-Adressen
- Domain-Namen und Fully Qualified Domain Name (FQDN)
- Merkmale wie Header-Informationen oder Hash-Werte bei der Kommunikation mit Protokollen wie HTTP oder TLS
- Charakteristiken verwendeter Applikationen (z.B. ZMap in Kapitel 2.1.4)

Einige dieser IoCs können mit geringem Aufwand verändert werden, weshalb entsprechende Merkmale möglichst aktuell zu halten sind (siehe „Pyramid of Pain“ in RFC 9424) [100].

Mit `probing.txt` aus Kapitel 2.1.5 hat eine Scan-Quelle eine Möglichkeit sich entsprechend auszuweisen [15]. Wie in Kapitel 2.1.4 erwähnt, weisen beispielsweise öffentliche Organisationen bereits ihre Scans aus, teilweise auch mit entsprechend verwendeten Quell-IP-Adressen oder -FQDNs [17, 101].

Die genannten Merkmale sind ggf. entweder aus dem Netzwerkverkehr direkt auszulesen oder werden durch weitere Anfragen ermittelt [100]. Dies kann z.B. das Ermitteln eines FQDNs anhand der IP-Adresse mittels DNS, Decodierung von Inhalten oder die Analyse eines zugehörigen AS sein [22, 100].

Aus den Scans gewonnene Informationen können verwendet werden, um Weiteres über die Scan-Quelle zu erfahren. Entspricht die IP-Adresse z.B. einem Tor Exit Node, weist dies darauf hin, dass der Scan aus dem Tor Netzwerk stammt und die Scan-Quelle sich nicht öffentlich zu erkennen geben will [102, 103]. Tor Exit Nodes sind im Internet auf diversen Seiten einsehbar, wobei einige nicht überwacht werden und somit nicht in entsprechenden Listen auftauchen [102, 104, 105].

Mit JA4+ wird eine Auswahl an Netzwerk-Fingerprinting-Methoden bereitgestellt [106]. Diese JA4+-Methoden werden von diversen Anwendungen und Diensten wie Wireshark, dem Netzwerk-Analyse-Werkzeug Arkime [107] oder Censys unterstützt [106, 108]. Darunter befinden sich folgende Implementationen [106]:

Tabelle 2.1.: JA4+ Implementationen [106]

Name	Kurzname	Beschreibung
JA4	JA4	TLS Client Fingerprinting
JA4Server	JA4S	TLS Server Response / Session Fingerprinting
JA4HTTP	JA4H	HTTP Client Fingerprinting
JA4Latency	JA4L	Client to Server Latency Measurment / Light Distance
JA4LatencyServer	JA4LS	Server to Client Latency Measurement / Light Distance
JA4X509	JA4X	X509 TLS Certificate Fingerprinting
JA4SSH	JA4SSH	SSH Traffic Fingerprinting
JA4TCP	JA4T	TCP Client Fingerprinting
JA4TCPServer	JA4TS	TCP Server Response Fingerprinting
JA4TCPScan	JA4TScan	Active TCP Fingerprint Scanner [109]

Diese Implementationen berechnen jeweils einen entsprechenden Fingerprint, mit welchen Fingerprint-Datenbanken durchsucht werden können [110].

Anhand des Protokolls ICMP kann der Inhalt eines Echo-Request-Pakets (Ping-Anfrage) Hinweise auf das verwendete Betriebssystem geben [111]. Ein genaueres Ermitteln des Betriebssystems erfolgt dennoch mittels entsprechenden TCP- und UDP-Anfragen, beispielsweise durch Nmap [44, 111].

2.3. Aufzeichnungs- und Analyse-Werkzeuge

Eine Recherche nach Analyse-Werkzeugen mit JA4+ im Fokus weist auf Werkzeuge wie Wireshark, Arkime und Zeek hin [106]. Software-Sammlungen oder „-Suiten“ bündeln einige der genannten Werkzeuge in einem Produkt, wie z.B. IVRE [112] oder Malcolm [113]. IVRE dient der aktiven Netzwerk-Aufklärung (Internet-Scans selber ausführen) während Malcolm der Analyse aufgezeichneter Verbindungen dient [112, 113].

Malcolm ist Open-Source-Software und wird in den USA vom Idaho National Laboratory (INL) zusammen mit der Cybersecurity Infrastructure Security Agency (CISA) (Departement von Homeland Security) entwickelt [114]. Die Anwendung reichert gesammelte Daten mit zusätzlichen Abfragen, u. a. dem geografischen Standort von IP-Adressen und JA4+-Fingerprinting an [115]. Erweiterungsmöglichkeiten sind dokumentiert und können in vorhandenen Anwendungen wie Arkime oder Zeek implementiert werden [116–119]. Malcolm ermöglicht die Korrelation von Zeek-Logs mit Arkime-Daten und erfährt regelmässig neue Releases [120, 121].

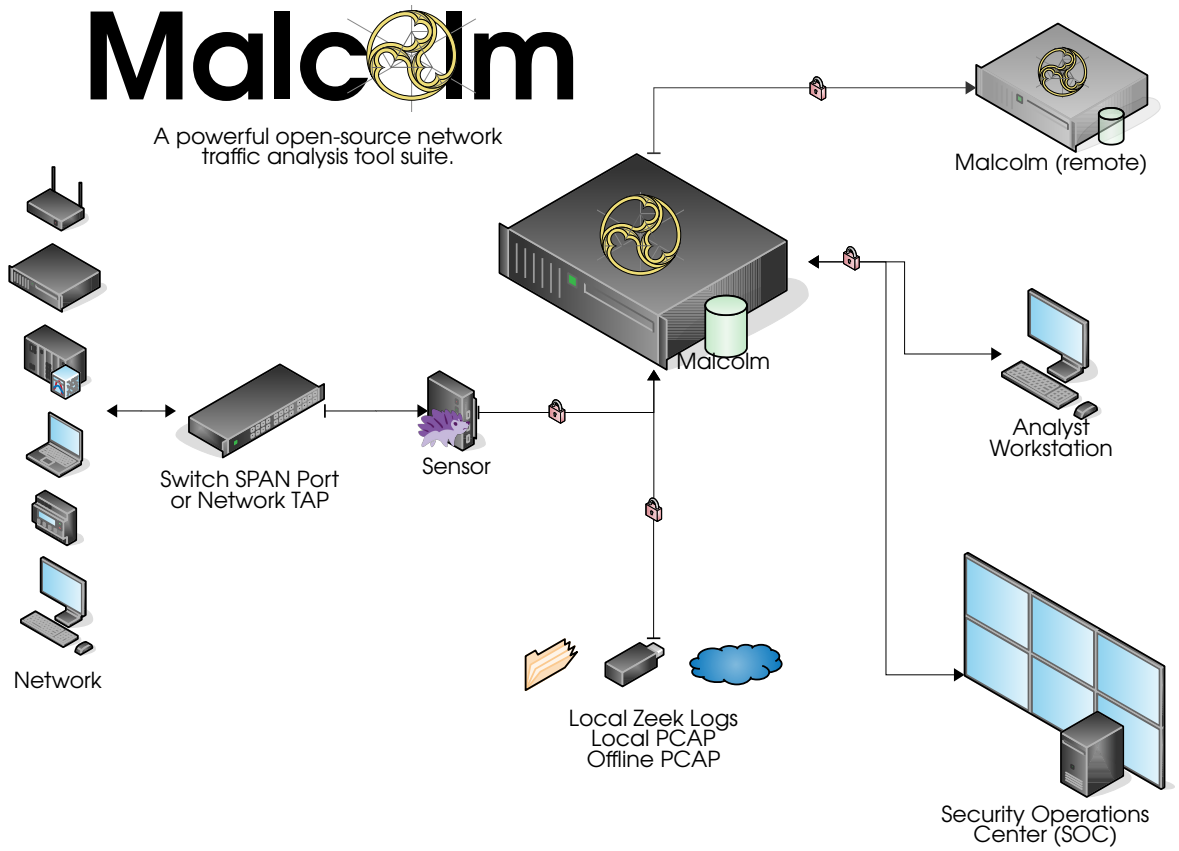


Abbildung 2.1.: Malcolm Netzwerkdiagramm [122]

Mit Malcolm ist ein Werkzeug gegeben, dass dank Open-Source-Aufbau und Dokumentation nachvollziehbar gepflegt wird und bereits etablierte Analyse-Software kombiniert [114, 121]. Die in Malcolm enthaltene Lösung Arkime bietet die Funktionalität, Netzwerkverkehr in Form von PCAP-Dateien zur anderweitigen Auseinandersetzung zu exportieren [120].

Die Verwendung der Programmbibliothek „libpcap“ [123], die als Industriestandard gilt und in unter anderem Wireshark sowie Arkime verwendet wird, ermöglicht es Aufzeichnungen mit einer Vielzahl von Anwendungen zu analysieren [61, 123–125]. Beispielsweise können so bestimmte exportierte Netzwerkpakete bei Bedarf mit Anwendungen wie TShark weiter verarbeitet werden.

Mit Abbildung 2.1 wird ein Beispiel zur Einbindung von Malcolm in eine Infrastruktur gezeigt [122]. Die Auflistung der Komponenten von Malcolm befindet sich in Abbildung 2.2 [126].

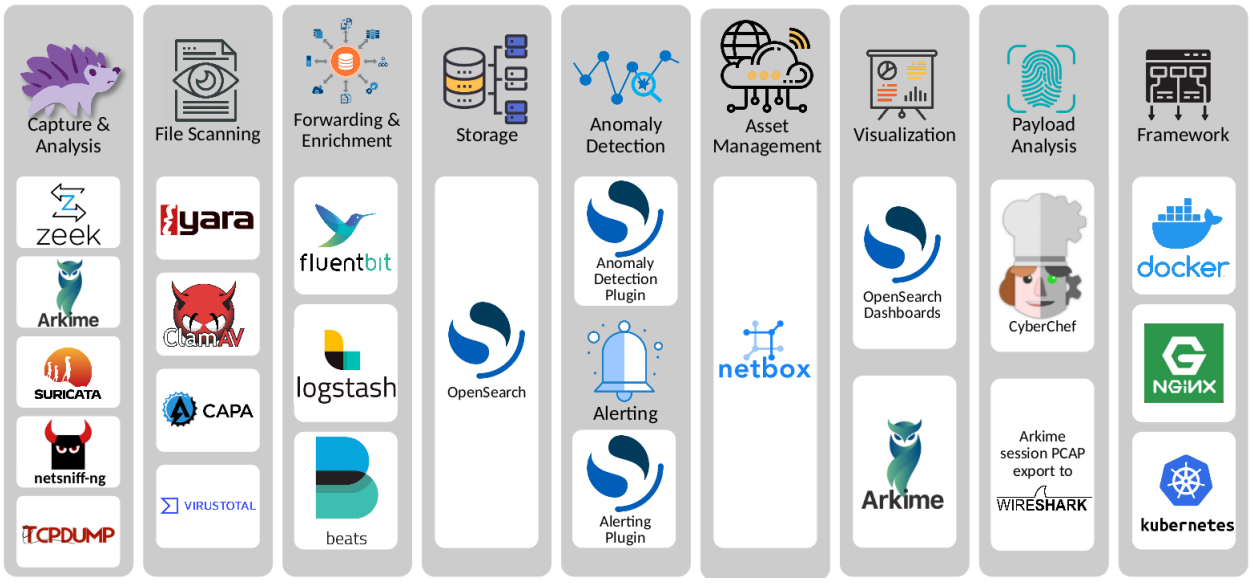


Abbildung 2.2.: Malcolm Komponentenübersicht [126]

Die Server der Analyse-Umgebung (Scan-Ziele und Malcolm-Komponenten) werden mittels VPN-Protokoll WireGuard [127] miteinander verbunden. Recherchen zu auf WireGuard basierenden Projekten führen zum Produkt innernet [128] als Konfigurationssystem für WireGuard [129]. Diese Software wird als Faktor zur Erfüllung des Arbeitsziels „Automatisierter und erweiterbarer Aufbau eines Scan-Ziels mit Debian Linux“ (siehe Kapitel A.1.3) gewählt.

Aufzuzeichnender Netzwerkverkehr findet auf den Scan-Zielen statt. Dieser Verkehr muss schnellstmöglich zur Malcolm-Instanz gelangen. ERSPAN ist ein von Cisco entwickeltes Protokoll, um Kopien von Netzwerkpaketen (Port-Mirroring) über IP-Netzwerke zu senden [130, 131]. Die Anwendung TC (`tc`) [132] ermöglicht es zusammen mit dem Befehl `ip` [133], Port-Mirroring mittels ERSPAN einzurichten[7].

Eine Malcolm-Instanz bietet keine Unterstützung dafür, direkt Netzwerkverkehr zu dessen Arkime-Komponente aufzuzeichnen, wenn diese eine lokale Instanz von OpenSearch [140] betreibt [141, 142]. Das Malcolm-Projekt empfiehlt dafür, mittels Hedgehog Linux [143] eine Appliance zur Aufzeichnung zu verwenden [141]. Diese leitet dann die Daten an die Malcolm-Instanz weiter (siehe „Sensor“ in Abbildung 2.1).

Ein Scan-Ziel gilt es automatisiert mit Debian Linux als Betriebssystem aufzubauen (siehe Kapitel A.1.3). Die Debian-Community empfiehlt die Installation mit dem offiziellen Debian-Installationsprogramm zu automatisieren („Preseeding Debian-Installer“) [144–146]. Mittels „Preseeding“ können Antworten auf Fragen während dem Installationsvorgang in einer Textdatei hinterlegt werden [145]. Diese Datei kann dem Debian-Boot-Medium beigefügt oder über das Netzwerk geladen werden [145, 146]. Nach der Installation des Betriebssystems kann die weitere Konfiguration über SSH durchgeführt werden, beispielsweise automatisiert mittels Ansible [147].

Malcolm und Hedgehog Linux werden mit der Software-Firewall ufw [148] entsprechend vorkonfiguriert ausgeliefert [149, 150]. Aus diesem Grund wird die Firewall-Konfiguration auf dem Scan-Ziel ebenfalls mit ufw getätigt.

Zur Verifikation der aufgebauten Umgebung in Kapitel 3.1.4 werden unter anderem spezifisch präparierte Netzwerkpakete mit Scapy verwendet [151].

[7] `ip` und `tc` sind beides Befehle der Software-Suite „iproute2“ des Linux Kernels [130, 134–139]

3. Methoden

Dieses Kapitel beschreibt das Vorgehen, um Scans im Internet zu erfassen und auszuwerten. Dazu wird eine Analyse-Umgebung aufgebaut, mit welcher der Netzwerkverkehr eines Scan-Ziels im Internet an die zentrale Analyse-Infrastruktur übermittelt wird.

3.1. Analyse-Umgebung

Aus der Evaluation der Aufzeichnungs- und Analyse-Werkzeuge in Kapitel 2.3 ergibt sich folgender Aufbau (siehe auch Abbildung 3.1):

- Malcolm-Instanz
  zur Analyse und Auswertung
- Appliance / Sensor mit Hedgehog Linux
  - VPN-Server mit innernet (WireGuard)
  - Aufzeichnung des Netzwerkverkehrs (Empfängt ERSPAN-Pakete von den Scan-Zielen)
- Scan-Ziele mit direkter Anbindung ans Internet
  - VPN-Client mittels innernet
  - Kopiert den Netzwerkverkehr in ERSPAN-Paketen über den VPN-Tunnel zum VPN-Server (Port-Mirroring)

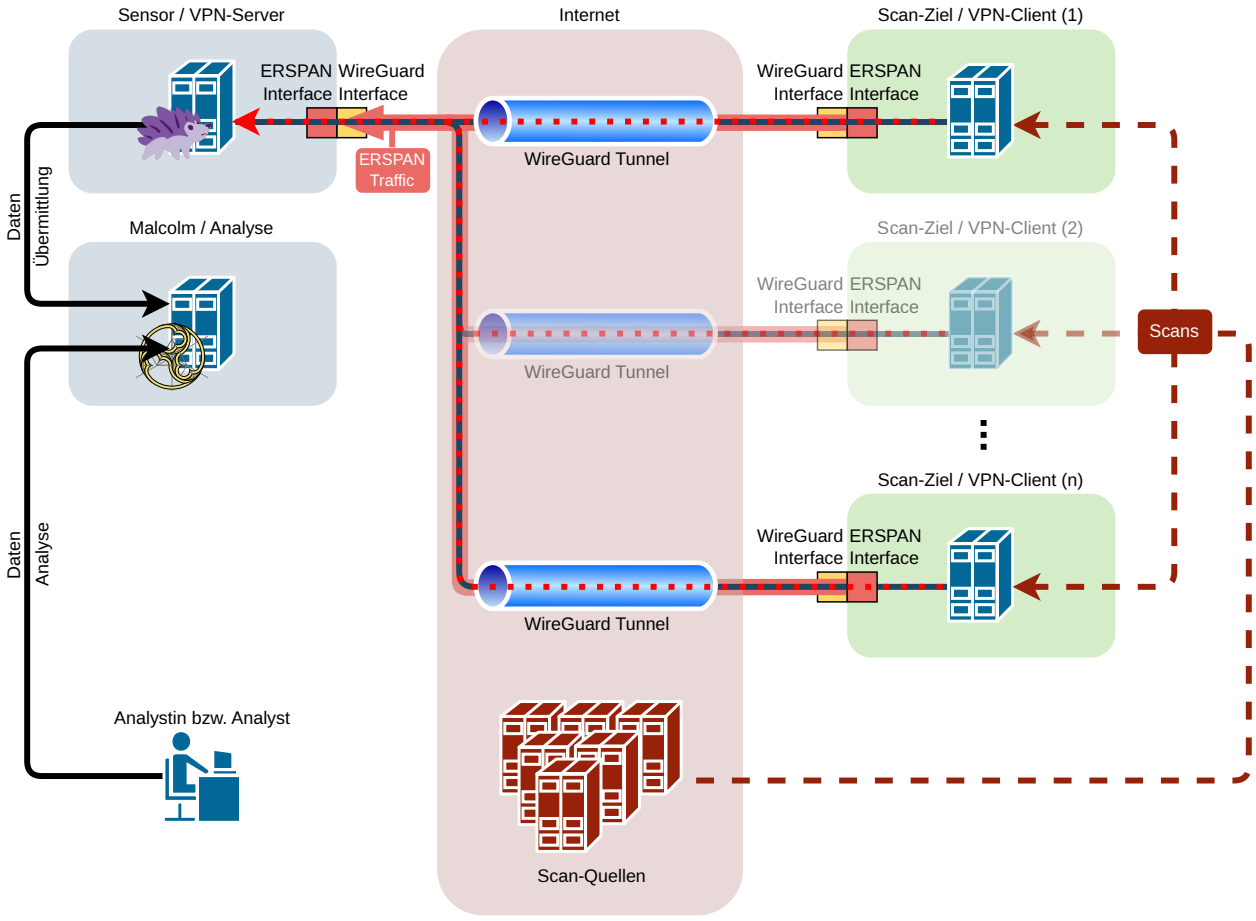


Abbildung 3.1.: Aufbau Analyse-Umgebung inklusive Port-Mirroring mittels ERSPAN [143, 152]

3.1.1. Installation

Die konkreten Installationsschritte sind im Anhang in Kapitel C dokumentiert, wobei in diesem Kapitel der grobe Aufbau aufgeführt wird. Auf sämtlichen Systemen wird sichergestellt, dass die Zeitsynchronisation u. a. mittels NTP konfiguriert ist.

Sämtlicher angefertigter Code ist der Arbeit beigelegt.
In Tabelle D.1 in Kapitel D befindet sich eine Übersicht entsprechender Dateien mit zugehöriger Beschreibung.
Nachfolgend relevante Ausschnitte befinden sich ebenfalls in Kapitel D.

Malcolm-Instanz

Die Malcolm-Instanz sowie die Appliance mit Hedgehog Linux werden gemäss Herstellerdokumentation aufgebaut [153]. Die Appliance wird gemäss nachfolgenden Ausführungen erweitert.

Hedgehog Linux Sensor / VPN-Server

Die Appliance fungiert gleichzeitig als VPN-Server, zu welchem die Scan-Ziele ihre Verbindung etablieren. Hierfür wird mit der Anwendung innernet eine Server-Konfiguration erstellt und aktiviert sowie für jeden Client eine spezifische Konfigurationsdatei angelegt [128].

Das Port-Mirroring mittels ERSPAN wird über virtuelle Netzwerkadapter des Typs ERSPAN implementiert [154]. Das ERSPAN-Paketformat basiert auf dem Format des Protokolls GRE [130, 131]. Ein Skript (`erspan-decapsule.sh`, siehe Kapitel D.2) sorgt dafür, dass auf dem VPN-Server ein dediziertes ERSPAN-Interface angelegt wird, eines für jede eindeutige Quell-IP-Adresse bei eintreffenden GRE-Paketen[8] [154, 156]. Netzwerkverkehr, der auf den einzelnen ERSPAN-Anbindungen eintrifft, wird mittels TC an ein definiertes „ERSPAN-Haupt-Interface“ kopiert [137, 157]. Dort wird dann von der Software der Hedgehog-Linux-Appliance die Aufzeichnung angesetzt [153]. Ermittelte Daten werden dann an die Malcolm-Instanz weitergeleitet. Damit unter Hedgehog Linux das aufzuzeichnende, virtuelle ERSPAN-Haupt-Interface nach einem Systemneustart wieder von der Anwendung gefunden wird, muss diesem eine statische MAC-Adresse zugewiesen werden (siehe Zeile 13 in Quelltext D.2 aus Kapitel D.2).

Zur Erweiterung der Analyse- und Detektionsfunktionalitäten wird die Sensor-Appliance um ein Zeek-Skript erweitert (siehe Kapitel 3.1.3 und D.2).

Scan-Ziel

Das Betriebssystem des Scan-Ziels wird, sofern möglich, mittels modifizierter Debian-ISO-Datei installiert. Dieses beinhaltet eine „Preseed“-Datei, mit welcher die Fragen der Debian-Installation automatisiert beantwortet werden [144, 158]. Mittels in der Preseed-Datei hinterlegter SSH-Public-Schlüssel kann eine Verbindung zum Scan-Ziel aufgebaut werden. Per Ansible-Playbook mit zugehörigen Rollen wird das Scan-Ziel weiter konfiguriert, um die nachfolgend beschriebene Funktionalität automatisiert aufzubauen.

Ein Scan-Ziel baut mittels zuvor angelegter innernet-Konfigurationsdatei einen VPN-Tunnel zum VPN-Server auf [128]. Hier wird ebenfalls ein ERSPAN-Interface mit dem VPN-Server als Ziel konfiguriert [154]. Dann wird der Netzwerkverkehr des Scan-Ziel-Interfaces an dessen ERSPAN-Interface mittels TC geklont [137, 157].

[8] Trifft ein GRE-Paket auf dem VPN-Interface des Servers ein, das eine noch unbekannte Quell-IP-Adresse hat, wird dafür ein ERSPAN-Interface angelegt. Ein praktisches Limit bezüglich der Anzahl an Netzwerk-Interfaces gibt es im Linux Kernel nicht [155]

Sollten die ERSPAN-Pakete über die VPN-Verbindung auf dem zu spiegelnden Interface gesendet werden, sind weitere Vorkehrungen zu treffen. Hierbei wird auf das Kopieren von GRE-ERSPAN-Paketen sowie des VPN-Verkehrs (hier UDP Port 51820) verzichtet. Somit werden Schleifen ausgeschlossen, um beispielsweise keine ERSPAN-Pakete zu gesendeten ERSPAN-Paketen zu generieren [137]. Um die Belastung des Sensors mit Hedgehog Linux zu verringern, werden nur eingehende Netzwerkpakete repliziert sowie bestimmte, lediglich lokal verwendete Ziel-Adressen und Protokolle ausgeschlossen (u. a. IPv6 Link-Local-Adressen [7] und ARP [159]).

Die Software-Firewall auf dem Scan-Ziel wird mittels ufw aktiviert. Dessen Standardeinstellungen blockieren eingehende Verbindungen und erlauben Ping-Anfragen mittels ICMP („Echo Requests“ [8, 9]) [160]. Zusätzlich werden eingehende SSH-Verbindungen auf TCP-Port 22 von vorgesehenen IP-Adressen erlaubt (in diesem Falle nur vom VPN-Server bzw. dem Sensor mit Hedgehog Linux). Dieser Port ist somit nicht aus dem Internet erreichbar.

Zur Einrichtung der Interfaces werden Skripts und systemd-Dienste angelegt, um nach einem Neustart schnellstmöglich wieder Port-Mirroring betreiben zu können (siehe Quelltext D.1).

Wurde eine öffentliche IP-Adresse eines Scan-Ziels zuvor für andere Zwecke verwendet, kann dies einen Einfluss auf das zu beobachtende Verhalten ausüben. In der Analyse wird dies nicht weiter beachtet, da hier die Selektion entsprechender IP-Adressen von den Hosting-Dienstleistungsunternehmen ausgeht und keine weitere Steuerungsmöglichkeit besteht.

Zusammenfassung Installation

Zusammenfassend wird mit diesem Aufbau der Netzwerkverkehr der Scan-Ziel-Interfaces mittels ERSPAN über einen VPN-Tunnel zur zentralen Appliance (VPN-Server mit Hedgehog Linux) kopiert. Dort wird der Verkehr aufgezeichnet und an die Malcolm-Instanz weitergeleitet. Die ERSPAN-Kapselung am Scan-Ziel und ERSPAN-Entkapselung auf dem VPN-Server mit dessen Interfaces ermöglicht es, Port-Mirroring über das Internet zu betreiben.

3.1.2. Konfiguration MTU

Um IP-Paketfragmentierungen bei der Weiterleitung von aufgezeichneten Paketen zu vermeiden, sind entsprechende MTU-Werte zu konfigurieren. Der übliche MTU-Wert eines Ethernet-Interfaces liegt bei 1500 Bytes [161]. Die Standard-Konfiguration von WireGuard definiert einen Wert von 1420 Bytes als MTU für dessen Netzwerkanbindung (also 80 Bytes Differenz zu 1500 Bytes) [127]. Bei der Verwendung von Port-Mirroring mittels ERSPAN wird zwischen dem ursprünglichen Paket und dem zugehörigen ERSPAN-Paket ein Grössenunterschied von 36 Bytes beobachtet (siehe Abbildung 3.2) [130].

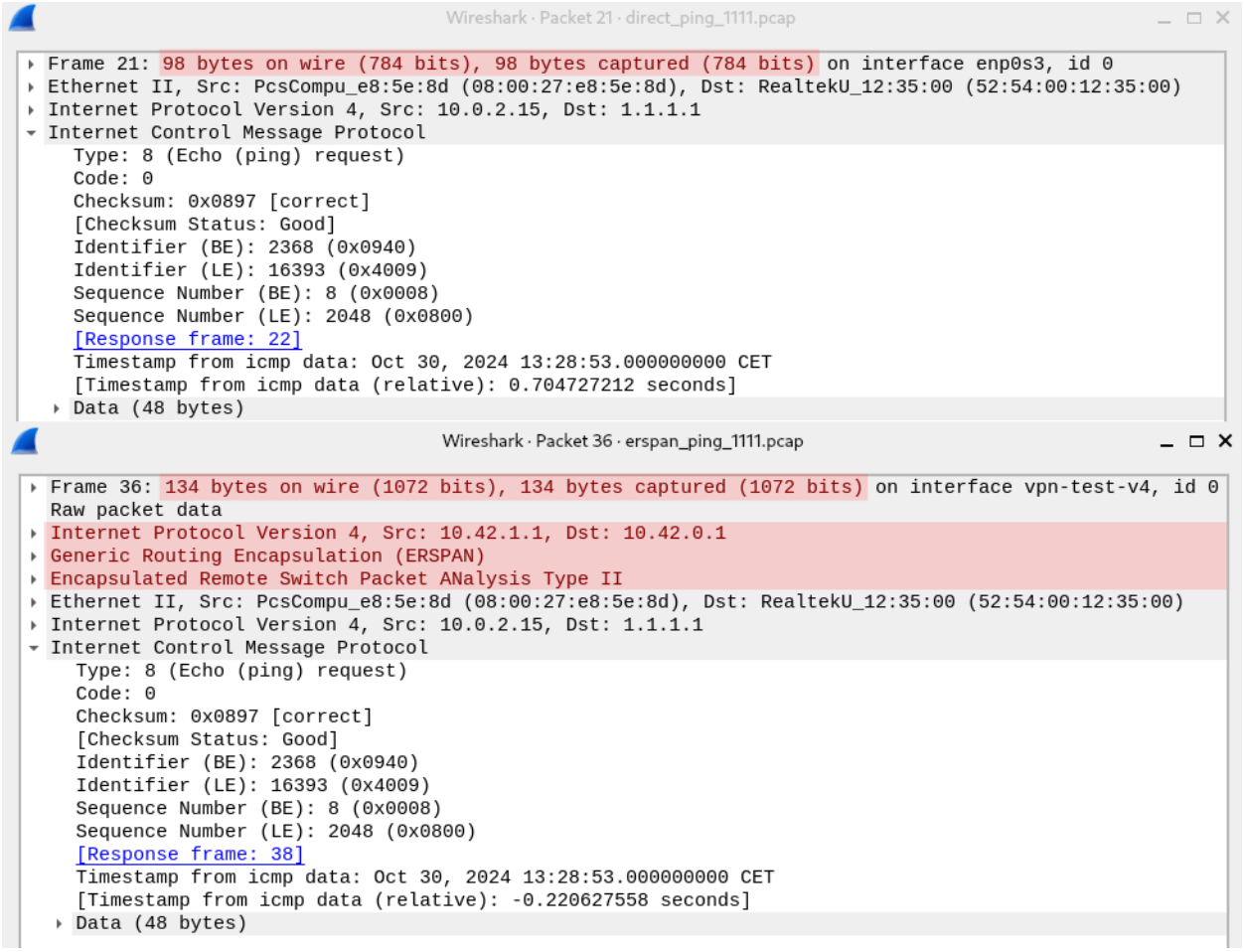


Abbildung 3.2.: Zusätzliche Header bei Port-Mirroring mittels ERSPAN, verifiziert mittels Ping-Anfrage in Prototyp-Umgebung (Oben Anfrage direkt auf Host aufgezeichnet, unten via ERSPAN erhaltenes Paket)

Bei der Konfiguration eines virtuellen ERSPAN-Interfaces mittels Linux-Befehl `ip` wird als MTU-Wert 50 Bytes weniger als der Wert des ERSPAN-Sende-Interfaces gesetzt[9].

[9] Dies anstelle der zuvor ermittelten 36 Bytes. Wird z.B. ein ERSPAN-Interface definiert, das über ein VPN-Interface mit einer MTU von 1280 Bytes senden soll, wird für die MTU des ERSPAN-Interfaces 1230 Bytes konfiguriert

Demnach ist mindestens folgender MTU-Wert zu wählen, um aufgezeichnete Pakete von einem Netzwerk-Interface am Internet ohne Fragmentierung via ERSPAN weiterleiten können:

$$1500 \text{ Bytes} - 80 \text{ Bytes} - 50 \text{ Bytes} = 1370 \text{ Bytes}$$

Die für IPv6 vorgeschriebene minimale Link-MTU mit 1280 Bytes bleibt somit erfüllt [162]. Die 1370-Bytes-MTU wird am Internet-Interface jedes Scan-Ziels definiert, was an folgender Abbildung veranschaulicht wird:

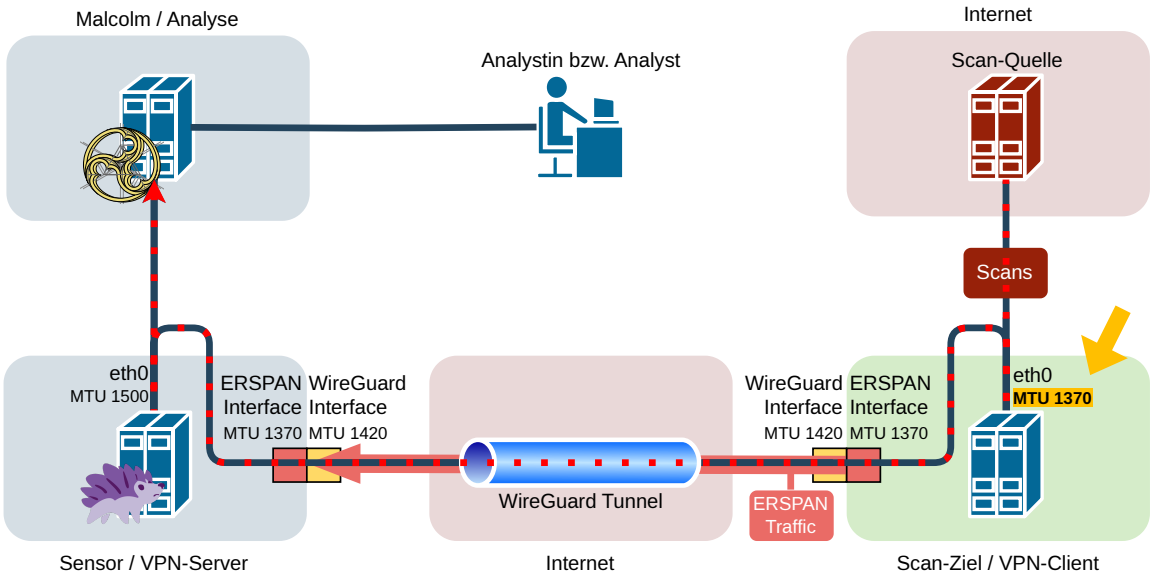


Abbildung 3.3.: MTU-Konfiguration des Internet-Interfaces eines Scan-Ziels inklusive Port-Mirroring mittels ERSPAN [143, 152]

Somit werden vom Scan-Ziel weitergeleitete Pakete über ERSPAN nicht fragmentiert. Andernfalls könnten Interpretationsfehler entstehen (siehe Abbildung C.1 im Anhang). Im Laufe dieser Arbeit werden nicht die ERSPAN-Pakete selbst betrachtet, sondern direkt damit weitergeleitete Netzwerkpakete ohne ERSPAN-Header. Auf die Konfiguration der MTU wird dennoch geachtet, um bei einer allfälligen Analyse besagte Fehlinterpretationen zu vermeiden.

3.1.3. Erweiterung der Analysefunktionalitäten

Die in Malcolm und Hedgehog Linux enthaltene Anwendung Zeek wird um ein Skript erweitert (`scannerdetection.zeek`, siehe Quelltext D.3) [117, 163]. Dieses wird auf dem Hedgehog-Linux-Sensor installiert und erweitert die Analyse des Netzwerkverkehrs um folgende Punkte [164–167]:

- Detektion der „Probe Attribution“ gemäss RFC 9511 [15]
  - Out-of-Band Probe Attribution
    - Unter der Quell-IP-Adresse oder entsprechendem Reverse-DNS-Eintrag ist via HTTP oder HTTPS die URI `/.well-known/probing.txt` erreichbar
      Zur Verringerung von fehlerhaften Detektionen muss hierfür der HTTP-Response-Code [168] `200` lauten und im Inhalt der String „contact“ [15] vorkommen (Gross-/Kleinschreibung wird nicht unterschieden).
      Zur Verringerung der Belastung durch das Zeek-Skript wird das Detektions-Ergebnis pro URI in einer Variable zwischengespeichert. Ist dieselbe Adresse nochmals zu prüfen, wird eine zuvor positive Detektion nochmals geloggt.
      Die Prüfung wird auf der Appliance bzw. dem Sensor mit Hedgehog Linux ausgeführt. Ist die zu prüfende Adresse von dort aus nicht erreichbar, kann keine Detektion hierzu erfolgen.
  - In-Band Probe Attribution
    - Eine „Probe Description URI“ im Format der Out-of-Band Probe Attribution, einer Mail-Adresse oder Telefonnummer ist in der Anfrage selbst enthalten, beispielsweise `https://example.net/.well-known/probing.txt`, `mailto:lab@example.net` oder `tel:+1-201-555-0123`
    - Die URI befindet sich zu Beginn des Paketinhalts bzw. der Payload und wird mittels Hexadezimal-Wert `0x00` terminiert, andernfalls muss vor der URI der Wert `0x00` platziert sein
    - Die Payload kann sich hierbei in einem ICMPv4- oder ICMPv6-Echo-Request-Paket, UDP-Datagram oder TCP-Segment befinden
    - In einem IPv6-Paket kann die URI entsprechend „PadN“-Option in einem „Hop-by-Hop“- oder „Destination Options“-Header mitgeführt werden
    - RFC 9511 führt keine abschliessende Aufzählung, jedoch prüft das Skript sämtliche TCP- und UDP-Payloads sowie mögliche IPv6-Extension-Header [162, 169, 170]
- Detektion von ZMap mittels Wert `54321` im IPv4-Identifikations-Feld [14, 27]
- Prüfen der Quell-Adresse anhand vorbereiteter Tabellen [171]
  - Hierbei wird die IP-Adresse sowie der rückwärts aufgelöste DNS-Eintrag geprüft
  - Ist kein FQDN, sondern eine Domain (beispielsweise `shodan.io`) in der Tabelle hinterlegt, werden alle zugehörigen FQDN gemeldet
  - Die Tabellen sind aufgeteilt in bekannte DNS-Einträge, IP-Adressen und -Subnetze
    In letzteren zwei Tabellen sind IPv4- und IPv6-Adressen bzw. -Subnetze möglich

Die Tabellen für das Zeek-Skript[10] werden halb-automatisiert angereichert. Hierbei wird zusätzlich eine entsprechende Intention vermerkt. Bekannte Scanner werden als „`good`“ (👍) deklariert, Adressen in Verbindung mit Malware oder sonstigen bösartigen Aktivitäten als „`bad`“ (👎).

Folgende Quellen werden für die Tabellen verwendet:

- 👍 Scanner-Liste von Collins [18]
  Siehe Quelltexte D.5 und D.6
- 👍 Censys [101]
  Siehe Quelltext D.7
- 👍 NCSC-UK-Domain [17]
  Manueller Eintrag `scanner.scanning.service.ncsc.gov.uk`, IP-Adressen bereits in Liste von Collins [18]
- 👍 RIPE Atlas Probe Liste [172, 173]
  Siehe Quelltext D.10
- 👍 Shadowserver-Domain [21]
  Manueller Eintrag `shadowserver.org`
- 👍 Shodan-Domain [19]
  Manueller Eintrag `shodan.io`
- 👍 Scanner-Liste von ShadowWhisperer [93]
  Siehe Quelltext D.8
- 👎 Malware-Listen von ShadowWhisperer [93]
  Siehe Quelltext D.8
- 👎 IoC-Listen von ThreatFox (abuse.ch) [94]
  Siehe Quelltext D.9
- 👎 Tor Exit Node Liste [174] Siehe Quelltext D.11
  Die Nutzung von Tor wird aufgrund dessen Eigenschaft, die Quelle zu tarnen, im Zusammenhang mit Scans hier als „bösartig“ eingestuft [103]

Die Reihenfolge, in welcher die Einträge zu den Tabellen hinzugefügt werden, fliesst in das Entfernen doppelter Einträge mit ein. Älteste Einträge geniessen hierbei die höchste Priorität. Nach der Anreicherung werden doppelte Einträge entfernt und nur die jeweils Ältesten in der Tabelle belassen mittels `awk -i inplace '!seen[$2]++' knownscanners*.table` [175, 176].

Domain-Einträge ohne konkrete FQDN werden nach prüfen einzelner zugehöriger IP-Einträge aus anderen Listen hinzugefügt. Da in dieser Arbeit jegliche Verbindungen zu den Scan-Zielen als Scan betrachtet werden, können beispielsweise sämtliche Einträge von Shodan (Domäne `shodan.io` [19]) als Scanner eingestuft werden.

Bei einer erfolgreichen Detektion wird eine sogenannte „Zeek-Notice“ angelegt. Hierbei wird je nach Art der Erkennung eine zugehörige Unterkategorie beigefügt (z.B. `ZMap` oder `ProbeAttribution_Inband`).

Daraus resultiert die Möglichkeit u. a. im „Zeek Notices“-Dashboard (siehe Abbildung 3.4) nach der Kategorie `ScannerDetection` zu filtern und eine Übersicht zu detektierten Verbindungen einzusehen.

Im Arkime-Dashboard der Malcolm-Instanz können entsprechende Einträge mit dem Filter `rule.category == ScannerDetection` ebenfalls ausgelesen werden.

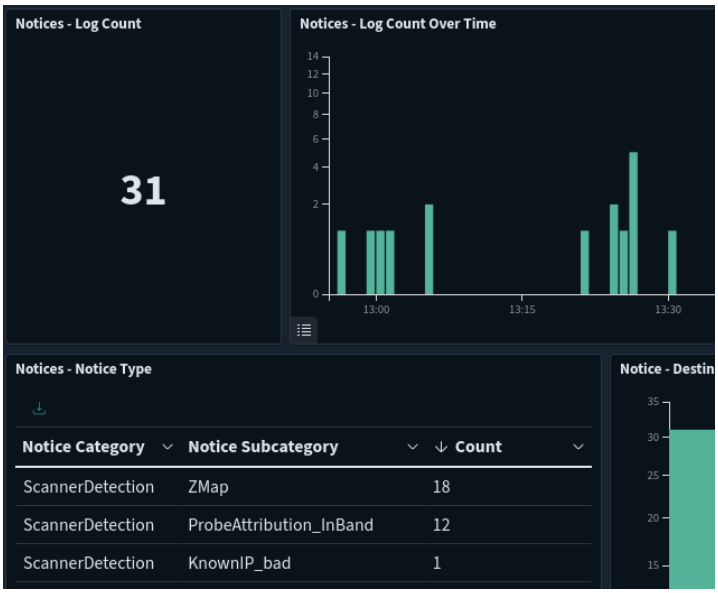


Abbildung 3.4.: Ausschnitt aus dem Dashboard „Zeek Notices“ mit detektierten Beispielverbindungen

[10] Konkret handelt es sich hierbei um die Dateien `knownscannersfqdn.table`, `knownscannersip.table` und `knownscannerssubnet.table`, die neben dem Zeek-Skript platziert sind (siehe Tabelle D.1 in Kapitel D)

Mit dem Skript `loaddata.sh` (siehe Tabelle D.1 in Kapitel D) auf dem Sensor mit Hedgehog Linux werden zuvor referenzierte Quelltexte kombiniert und täglich ausgeführt, um die Daten in den Tabellen[11] möglichst aktuell zu halten. Ermittelte Daten werden an die bestehenden Tabellen angefügt, wobei diese am Schluss wieder mit `awk` von doppelten Einträgen befreit werden. Manuell angelegte Einträge bleiben somit intakt.

Nachdem erste Scans erfolgt und das „Zeek Notices"-Dashboard entsprechend angereichert wird, stehen im Zeek-Log ermittelte FQDNs der Scan-Quellen zur Verfügung. Fehlerhafte HTTP-Anfragen für die Out-of-Band Probe Attribution des Zeek-Skripts werden in der Datei `reporter.log` gespeichert [177]. Dieses Log kann auf dem Sensor mit Hedgehog Linux beispielsweise mittels nachfolgendem Befehl ausgewertet werden, um aufgelöste FQDNs auszulesen.

```
zcat /home/sensor/zeek_logs/logs/*/reporter.*.gz \
| grep -Eo "http(s?):\/\/[^\/]*" | grep -E ".[a-z]+$" \
| sed 's/^\(http\|https\):\/\///g' | sort | uniq | less
```

Quelltext 3.1: Auslesen ermittelter FQDNs von Scan-Quellen aus dem Log fehlerhafter HTTP-Anfragen des Zeek-Skripts

Ermittelte Domänen werden manuell mittels Suchmaschinen und allfällig gefundenen Webseiten geprüft. Sind dort Scans im Internet ausgewiesen, erfahren die zuvor erwähnten Tabellen[11] entsprechende Einträge. Folgende Domänen werden hierbei ermittelt und zusätzlich als bekannte Scanner („good", 👍) deklariert:

- 👍 `internet-measurement.com` [178]
  Ausgewiesene IP-Adressen und Subnetze zusätzlich zur Domäne erfasst
- 👍 `internet-census.org` [179]
- 👍 `censys-scanner.com` [101, 180, 181][12]
- 👍 `research-scanner.com` [183]
- 👍 `netsecscan.net` [184]
- 👍 `recyber.net` [185]
- 👍 `cyberresilience.io` [186]
- 👍 `probe.onyphe.net` [187, 188]
- 👍 `ant.isi.edu` [189]
- 👍 `skipa.cyberok.ru` [190]
- 👍 `stretchoid.com` [191]
- 👍 `scan.leakix.org` [192]
  `leakix.org` leitet weiter auf `leakix.net`
- 👍 `googlebot.com` [193]
  `googlebot.com` leitet auf ref. Artikel weiter
- 👍 IP-Adresse
  Cambridge Cybercrime Centre [194]
  Hierbei wird weder Domäne noch FQDN angegeben

```
zcat /home/sensor/zeek_logs/logs/*/reporter.*.gz \
| grep -Eo "http(s?):\/\/[^\/]*" | grep censys-scanner.com -B1
```

Quelltext 3.2: Auslesen der Einträge nahe bei Anfragen zu `censys-scanner.com` von Scan-Quellen aus dem Log fehlerhafter HTTP-Anfragen des Zeek-Skripts (Es gehören nicht alle hier ermittelten Einträge zu Censys, jede IP-Adresse muss manuell geprüft werden)

Danach werden gemäss zuvor aufgeführtem `awk`-Befehl die Tabellen von doppelten Einträgen befreit.

[11] Konkret handelt es sich hierbei um die Dateien `knownscannersfqdn.table`, `knownscannersip.table` und `knownscannerssubnet.table`, die neben dem Zeek-Skript platziert sind (siehe Tabelle D.1 in Kapitel D)

[12] Zu `censys-scanner.com` ist weder eine Webseite noch ein Whois-Eintrag mit klarer Scan-Ausweisung ersichtlich [182]. Werden im Log zur Domäne zugehörige IP-Adressen ausgelesen (siehe Quelltext 3.2), sind diese zusätzlich wie folgt zu prüfen: Whois-Eintrag ermitteln (muss zu Censys gehören), IP-Adresse rückwärts auflösen (muss zu `censys-scanner.com` gehören) und ausgewiesene IP-Adressen von Censys prüfen [101] (muss aufgelistet sein). Ermittelte FQDNs wie `scanner-11.ch1.censys-scanner.com` lassen sich mittels DNS nicht zu einer IP-Adresse auflösen. Daher müssen entsprechende Quell-IP-Adressen direkt betrachtet werden. Solche Stichproben führen zur Erkenntnis, dass die Domäne `censys-scanner.com` zu Censys gehört

Weitere Domänen potenzieller Scan-Institutionen wie `binaryedge.ninja`, `scan.bufferover.run` oder `internet-research-project.net` stehen gemäss Recherchen in Verbindung mit Internet-Scan-Projekten, weisen deren Scan-Quellen jedoch nicht aus [195, 196]. Auf Anfragen per E-Mail am 18. Dezember 2024 bleiben Antworten ausstehend. Einige IP-Adressen von „Binaryedge“ und „bufferover-run“ sind bereits in der Liste bekannter Scan-Quellen von Collins enthalten und werden somit in die Analyse miteinbezogen [18].

Wie erwähnt verwendet Malcolm mehrere Komponenten zur Analyse des Netzwerkverkehrs wie Arkime und Zeek (siehe Abbildung 3.5) [197]. Das Zeek-Skript zur Detektion der Scan-Quellen reichert lediglich die Daten aus Zeek an.

Die Konsequenz daraus ist, dass zur Betrachtung der Scans Zeek im Fokus liegt und beispielsweise die Ergebnisse von Arkime gegebenenfalls für weiterführende Analysen dazu gezogen werden können.

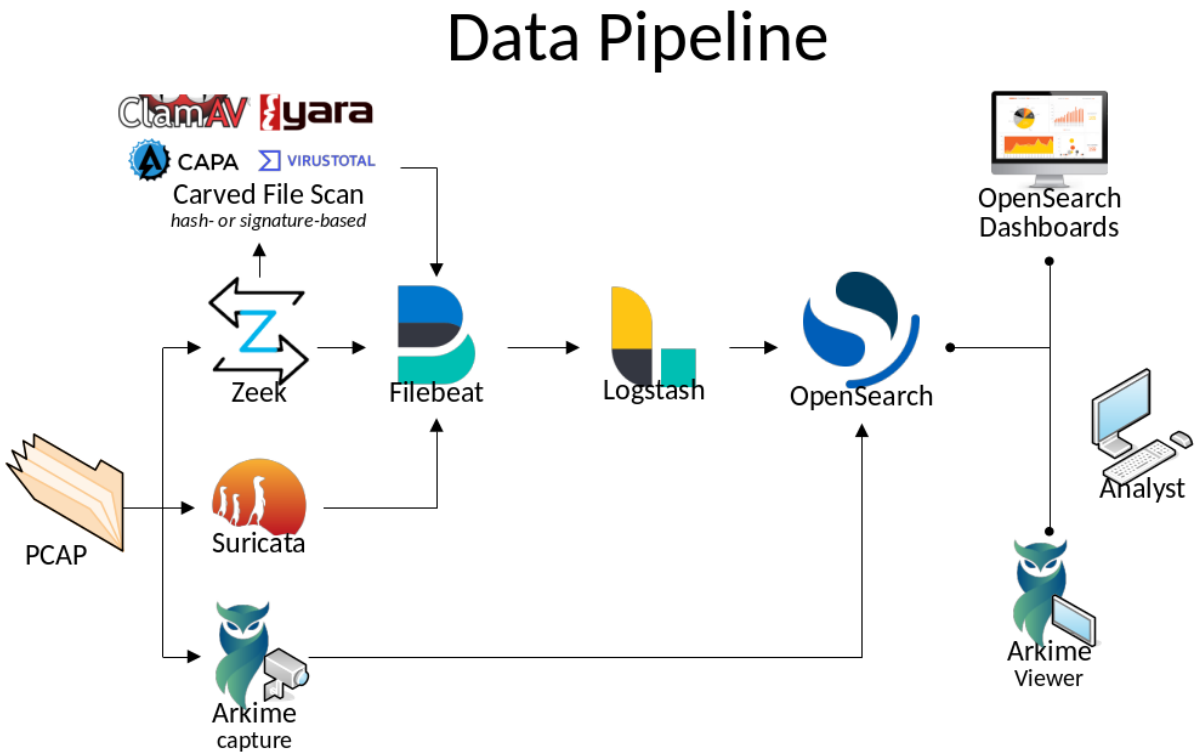


Abbildung 3.5.: Datenverarbeitung in Malcolm [115]

3.1.4. Verifikation

Die aufgebaute Analyse-Umgebung gilt es vor dem produktiven Einsatz zu verifizieren.

Auf der Appliance mit Hedgehog Linux sollen die ERSPAN-Pakete der Scan-Ziele eintreffen. Loggt dort das zugehörige Skript eine Verbindung mit der VPN-Client-IP-Adresse des Scan-Ziels, gilt die VPN-Kommunikation als bestätigt. Pro Scan-Ziel soll auf der Appliance ein virtuelles ERSPAN-Interface anzutreffen sein (hier mit Präfix `myerspan` und einer eindeutigen Nummer).

Zur weiteren Verifikation des Port-Mirrorings können die zuvor genannten ERSPAN-Interfaces mittels `tcpdump` betrachtet werden. Der Netzwerkverkehr des entsprechenden Scan-Ziels muss dort ersichtlich sein. Es gilt zum Scan-Ziel Netzwerkverkehr zu generieren, beispielsweise mittels ICMP-Echo-Request- bzw. Ping-Anfragen. Das ERSPAN-Haupt-Interface (hier mit Namen `myerspan`, ohne Suffix) auf der Appliance vereint sämtliche ERSPAN-Interfaces, was ebenfalls mittels `tcpdump` geprüft wird.

Bei einer funktionierenden Weiterleitung der Daten von der Appliance zur Malcolm-Instanz, kann der Netzwerkverkehr in dessen Weboberfläche eingesehen werden. Hierfür wird zur HTTPS-Adresse der Malcolm-Instanz navigiert und die bei der Installation hinterlegten Login-Daten eingegeben. Daraufhin erscheint eine Übersicht, in welcher unter anderem die Einträge „Dashboards“ und „Arkime“ zur Auswahl stehen. Durch Klick auf „Dashboards“ wird das „Overview“-Dashboard angezeigt, in welchem nach Anpassung des zu fokussierenden Zeitfensters eine Übersicht zu vergangenen Verbindungen aufgezeigt wird. Das gleiche Verhalten ist bei der Wahl von „Arkime“ inklusive Zeitfenster-Anpassung festzustellen, wobei es sich hierbei um eine andere Ansicht handelt.

Arkime bietet eine Ansicht zum beobachteten Netzwerkverkehr sowie den Download entsprechender Aufzeichnungen in Form von PCAP-Dateien [120]. Daten von Zeek werden in Arkime von Malcolm integriert, was das Korrelieren entsprechender Logs ermöglicht [120]. Mit OpenSearch Dashboards können eigene Visualisierungen mit denselben Daten erstellt werden [198].

Das Verhalten des Zeek-Skripts (siehe Quelltext D.3) und beispielsweise deren Erkennung einer „In-Band Probe Attribution“ gemäss RFC 9511 [15] bedarf an präparierten Netzwerkpaketen. Diese werden unter anderem mittels Scapy aufgebaut und an ein Scan-Ziel gesendet. In Kapitel D.3.2 sowie der nachfolgenden Tabelle sind zugehörige Befehle vorzufinden. Das Scan-Ziel betreibt hierbei aktiv Port-Mirroring zum Sensor mit Hedgehog Linux und befindet sich in der aufgebauten Analyse-Umgebung.

Je nach Auslastung des Sensors ist nach Versand des Test-Pakets innerhalb von Sekunden bis Minuten ein entsprechender Eintrag in Malcolm vorzufinden[13]. Führen die Eigenschaften eines Netzwerkpakets zur Detektion mehrerer Punkte, werden sämtliche Treffer aufgeführt. Ist beispielsweise eine Quell-IP-Adresse in der Liste bekannter Adressen und enthält die Payload eine In-Band Probe Attribution, so werden zwei Einträge angelegt.

[13] Das Zeek-Skript prüft jedes Netzwerkpaket auf dessen Inhalt sowie deren Absender-Adresse inklusive DNS- und Web-Anfragen (`probing.txt`). Bereits durchgeführte Optimierungen des Zeek-Skripts und -Auswertungsvorgangs sowie die Erhöhung der CPU-Ressourcen führen schneller zu Ergebnissen. Zu UDP hat Zeek eine Minute als Standardwert zur Zeitüberschreitung bei Verbindungen hinterlegt („UDP Inactivity Timeout“) [199]

Tabelle 3.1.: Verifikation des Zeek-Skripts mittels Test-Netzwerkpaketen [200–202] (Der gemäss RFC 9511 verwendete Hexadezimal-Wert 0x00 wird in der Payload-Beschreibung so beibehalten (siehe Liste in Kapitel 3.1.3))

Beschreibung und Befehl	Detektion ✅ Erfolgreiche Detektion ❌ Keine Detektion
UDP-Datagramm von Quelle aus gesendet, die `probing.txt`-Datei unter `http://IP/.well-known/probing.txt` hinterlegt hat URI von Sensor-Appliance mit Hedgehog Linux aus erreichbar `echo "somepayload" > /dev/udp/10.0.2.22/45678` Quelltext 3.3: Test-Netzwerkpaket: UDP-Datagramm mit nicht-triggernder Payload von Host mit erreichbarer `probing.txt`-Datei unter `http://IP/.well-known/probing.txt`	Out-of-Band Probe Attribution ✅
ICMPv4 Echo-Request mit E-Mail-Adresse zu Beginn der Payload Terminiert mit 0x00 `(echo "from scapy.all import *"; echo 'sr(IP(dst ="10.0.2.22")/ICMP()/"\x6d\x61\x69\x6c\x74\x6f\x3a\ x6d\x61\x69\x6c\x40\x65\x78\x61\x6d\x70\x6c\x65\x2e \x63\x6f\x6d\x00", timeout=3)') | python3` Quelltext 3.4: Test-Netzwerkpaket: ICMPv4 Echo-Request mit Payload `mailto:mail@example.com0x00`	In-Band Probe Attribution mit E-Mail-Adresse ✅
ICMPv4 Echo-Request mit E-Mail-Adresse zu Beginn der Payload **Nicht** terminiert mit 0x00 `(echo "from scapy.all import *"; echo 'sr(IP(dst ="10.0.2.22")/ICMP()/"\x6d\x61\x69\x6c\x74\x6f\x3a\ x6d\x61\x69\x6c\x40\x65\x78\x61\x6d\x70\x6c\x65\x2e \x63\x6f\x6d", timeout=3)') | python3` Quelltext 3.5: Test-Netzwerkpaket: ICMPv4 Echo-Request mit Payload `mailto:mail@example.com`	In-Band Probe Attribution mit E-Mail-Adresse ❌ Nicht detektiert, da Payload nicht mit 0x00 terminiert
ICMPv6 Echo-Request mit `probing.txt`-URI, Telefonnummer und E-Mail-Adresse nicht zu Beginn der Payload 0x00 vor jeder einzelnen Angabe `(echo "from scapy.all import *"; echo 'sr(IPv6(dst=" fe80::a00:27ff:fe95:bb91")/ICMPv6EchoRequest()/"\ x68\x65\x6c\x6c\x6f\x20\x77\x6f\x72\x6c\x64\x00\x68 \x74\x74\x70\x3a\x2f\x2f\x74\x68\x69\x73\x2e\x69\ x73\x2e\x75\x72\x6c\x2f\x2e\x77\x65\x6c\x6c\x2d\x6b \x6e\x6f\x77\x6e\x2f\x70\x72\x6f\x62\x69\x6e\x67\ x2e\x74\x78\x74\x00\x74\x65\x6c\x3a\x2b\x31\x2d\x32 \x30\x31\x2d\x35\x35\x35\x2d\x30\x31\x32\x33\x00\ x6d\x61\x69\x6c\x74\x6f\x3a\x6d\x61\x69\x6c\x40\x65 \x78\x61\x6d\x70\x6c\x65\x2e\x63\x6f\x6d", timeout =3)') | python3` Quelltext 3.6: Test-Netzwerkpaket: ICMPv6 Echo-Request mit Payload `hello world0x00 http://this.is.url/.well-known/probing.txt0x00 tel:+1-201-555-01230x00mailto:mail@example.com`	In-Band Probe Attribution mit `probing.txt`-URI ✅ Telefonnummer ✅ E-Mail-Adresse ✅ Dreifache Detektion

Verifikation Zeek-Skript mittels Test-Netzwerkpaketen Fortsetzung

Beschreibung und Befehl	Detektion ✓ Erfolgreiche Detektion ✗ Keine Detektion
ICMPv6 Echo-Request mit `probing.txt`-URI, Telefonnummer und falsch angegebener E-Mail-Adresse nicht zu Beginn der Payload `0x00` nur vor `probing.txt`-URI und Telefonnummer `(echo "from scapy.all import *"; echo 'sr(IPv6(dst="fe80::a00:27ff:fe95:bb91")/ICMPv6EchoRequest()/"\x68\x65\x6c\x6c\x6f\x20\x77\x6f\x72\x6c\x64\x00\x68\x74\x74\x70\x3a\x2f\x2f\x74\x68\x69\x73\x2e\x69\x73\x2e\x75\x72\x6c\x2f\x2e\x77\x65\x6c\x6c\x2d\x6b\x6e\x6f\x77\x6e\x2f\x70\x72\x6f\x62\x69\x6e\x67\x2e\x74\x78\x74\x00\x74\x65\x6c\x3a\x2b\x31\x2d\x32\x30\x31\x2d\x35\x35\x35\x2d\x30\x31\x32\x33\x6d\x61\x69\x6c\x74\x6f\x3a\x6d\x61\x69\x6c\x40\x65\x78\x61\x6d\x70\x6c\x65\x2e\x63\x6f\x6d", timeout=3)') | python3` Quelltext 3.7: Test-Netzwerkpaket: ICMPv6 Echo-Request mit Payload `hello world0x00 http://this.is.url/.well-known/probing.txt0x00 tel:+1-201-555-0123mailto:mail@example.com`	In-Band Probe Attribution mit `probing.txt`-URI ✓ Telefonnummer ✓ E-Mail-Adresse ✗ Nicht detektiert, da der Wert `0x00` nicht davor platziert ist Zweifache Detektion
IPv4-Paket mit Wert `54321` im IPv4-Identifikations-Feld `(echo "from scapy.all import *"; echo 'sr(IP(dst="10.0.2.22",id=54321)/TCP(), timeout=3)') | python3` Quelltext 3.8: Test-Netzwerkpaket: IPv4-Paket mit Wert `54321` im IPv4-Identifikations-Feld	ZMap ✓
ICMPv4 Echo-Request von Quelle aus gesendet, deren rückwärts aufgelöste Adresse zur Domain `shadowserver.org` gehört IP-Adresse ist **nicht** in Tabelle bekannter IP-Adressen aus Kapitel 3.1.3 Die Domäne `shadowserver.org` ist in Tabelle bekannter Domänen und FQDNs hinterlegt `(echo "from scapy.all import *"; echo 'sr(IP(src="154.9.2.12",dst="10.0.2.22")/ICMP(), timeout=3)') | python3` Quelltext 3.9: Test-Netzwerkpaket: ICMPv4 Echo-Request mit nicht-triggernder Payload von bekannter Quell-Adresse aus Domain `shadowserver.org`	Quell-Domäne ✓ IP-Adresse wird zu `mail.shadowserver.org` rückwärts aufgelöst und `shadowserver.org` befindet sich in Tabelle bekannter Domänen
ICMPv4 Echo-Request von Quelle aus gesendet, deren rückwärts aufgelöste Adresse `scanner.scanning.service.ncsc.gov.uk` lautet IP-Adresse ist in Tabelle bekannter IP-Adressen aus Kapitel 3.1.3 Der FQDN `scanner.scanning.service.ncsc.gov.uk` ist zusätzlich in Tabelle bekannter Domänen und FQDNs hinterlegt `(echo "from scapy.all import *"; echo 'sr(IP(src="35.177.10.231",dst="10.0.2.22")/ICMP(), timeout=3)') | python3` Quelltext 3.10: Test-Netzwerkpaket: ICMPv4 Echo-Request mit nicht-triggernder Payload von bekannter Quell-Adresse mit FQDN `scanner.scanning.service.ncsc.gov.uk`	Quell-FQDN ✓ IP-Adresse wird zu `scanner.scanning.service.ncsc.gov.uk` rückwärts aufgelöst und befindet sich in Tabelle bekannter FQDNs Quell-IP-Adresse ✓ Zweifache Detektion

Verifikation Zeek-Skript mittels Test-Netzwerkpaketen Fortsetzung

Beschreibung und Befehl	Detektion Erfolgreiche Detektion Keine Detektion
ICMPv6 Echo-Request von Quelle aus gesendet, deren IP-Adresse aus einem bekannten Subnetz stammt IP-Adresse ist **nicht** in Tabelle bekannter IP-Adressen aus Kapitel 3.1.3 Zugehöriges IP-Netzwerkwerk ist in Tabelle bekannter IP-Subnetze `(echo "from scapy.all import *"; echo 'sr(IPv6(src="2620:96:e000:b0cc:e::dead:beef",dst="fe80::a00:27ff:fe95:bb91")/ICMPv6EchoRequest(), timeout=3)') | python3` Quelltext 3.11: Test-Netzwerkpaket: ICMPv6 Echo-Request mit nicht-triggernder Payload von bekanntem IP-Subnetz	Quell-Subnetz IP-Adresse befindet sich in bekanntem Subnetz `2620:96:e000:b0cc:e::/80`

Weitere Befehle und Detektionsverhalten können der Tabelle D.2 in Kapitel D.3.2 entnommen werden. Eine Kombination der aufgeführten Befehle wird ebenfalls durchgeführt (anderes Protokoll mit gleicher Payload, Variationen der In-Band URIs und dessen Kombinationen, etc.), wird jedoch aus Übersichtsgründen nicht aufgelistet.

Die daraus entstehenden „Zeek-Notices“ können entsprechend nachfolgender Abbildung im zugehörigen Dashboard auf der Malcolm-Instanz eingesehen werden. Dieselben Daten sind ebenfalls in der Arkime-Oberfläche derselben Instanz ersichtlich (siehe Abbildung 3.7).

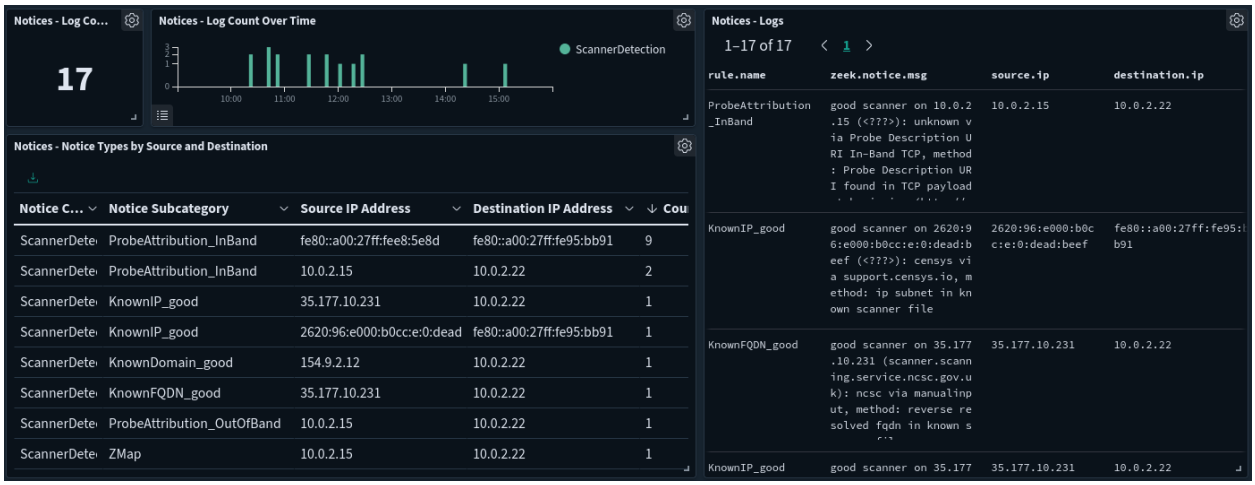
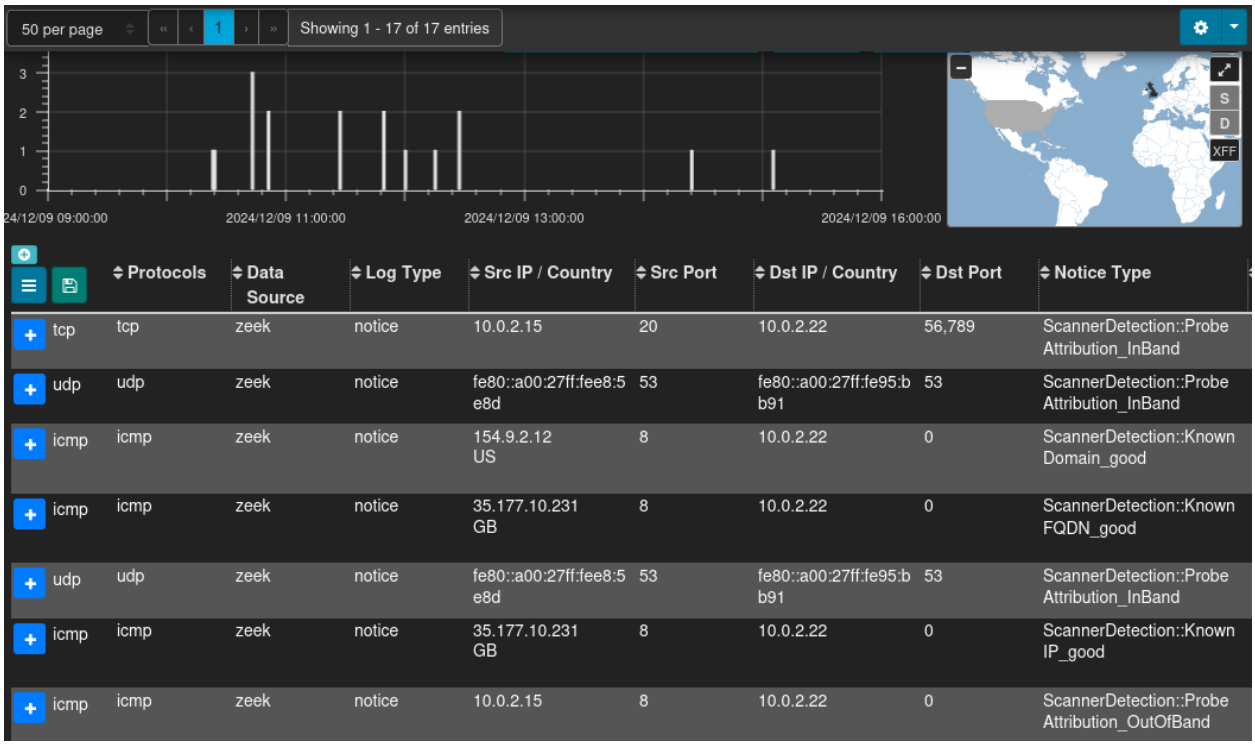


Abbildung 3.6.: Zeek-Notices gemäss Verifikation aus Tabellen 3.1 und D.2 („Zeek Notices“-Dashboard optisch angepasst)



Protocols	Data Source	Log Type	Src IP / Country	Src Port	Dst IP / Country	Dst Port	Notice Type
tcp	zeek	notice	10.0.2.15	20	10.0.2.22	56,789	ScannerDetection::Probe Attribution_InBand
udp	zeek	notice	fe80::a00:27ff:fee8:5e8d	53	fe80::a00:27ff:fe95:bb91	53	ScannerDetection::Probe Attribution_InBand
icmp	zeek	notice	154.9.2.12 US	8	10.0.2.22	0	ScannerDetection::Known Domain_good
icmp	zeek	notice	35.177.10.231 GB	8	10.0.2.22	0	ScannerDetection::Known FQDN_good
udp	zeek	notice	fe80::a00:27ff:fee8:5e8d	53	fe80::a00:27ff:fe95:bb91	53	ScannerDetection::Probe Attribution_InBand
icmp	zeek	notice	35.177.10.231 GB	8	10.0.2.22	0	ScannerDetection::Known IP_good
icmp	zeek	notice	10.0.2.15	8	10.0.2.22	0	ScannerDetection::Probe Attribution_OutOfBand

Abbildung 3.7.: Zeek-Notices in Arkime-Oberfläche gemäss Verifikation aus Tabellen 3.1 und D.2

Einzelne Server und Komponenten nehmen bei einer korrekten Konfiguration nach einem Neustart automatisch wieder den Betrieb auf. Somit können Scans im Internet erfasst und ausgewertet werden.

3.2. Manuelle Auswertung

Mit dem Aufbau der Analyse-Umgebung wird der Aufwand für manuelle Auswertungen signifikant verringert. Trotzdem gilt es nachfolgend aufgeführte Analysen manuell vorzunehmen.

JA4+-Fingerprints werden mittels mitgelieferten Arkime- und Zeek-Plugins berechnet [203, 204]. Die öffentlich verwaltete JA4+-Datenbank beinhaltet über fünfzigtausend Einträge, jedoch lässt sich nicht von jedem Eintrag auf die Absicht der Quelle schliessen [110]. Aus diesem Grund werden die berechneten JA4+-Fingerprints manuell mit der zugehörigen Datenbank abgeglichen und übereinstimmende Einträge einzeln betrachtet.

Hierfür kann beispielsweise mittels Arkime und dem Ausdruck `event.dataset == ja4*` nach JA4+-spezifischen Einträgen gefiltert werden.

Zusätzlich gilt es Standort-relevante Angaben aufgrund der späteren Ausführungen in Kapitel 3.5.2 manuell auszuwerten.

3.3. Hosting-Wahl und globaler Aufbau

Die Malcolm-Instanz sowie der Sensor bzw. die Appliance mit Hedgehog Linux sind in der Cyber-Security-Lab-Infrastruktur der Berner Fachhochschule (BFH) platziert (siehe Kapitel A.4.8 und C). Beide befinden sich im gleichen Netzwerk, wobei der VPN-Serverdienst der Appliance über das Internet erreichbar ist (via UDP-Port 51820).

Zusätzlich befindet sich ein Scan-Ziel in derselben Infrastruktur. Dieses ist auf Netzwerkebene von den zuvor genannten Hosts getrennt und direkt am Internet mit öffentlichen IPv4- und IPv6-Adressen angebunden.

> Eine Übersicht der Analyse-Umgebung inklusive Hosting-Dienstleistungsunternehmen, IP-Adressen und Aufzeichnungszeitraum ist in Kapitel C.4 aufgelistet.

Aufgrund ihres Bekanntheitsgrades werden unter anderem die Produkte Alibaba Cloud ECS [205], Amazon EC2 [206], Hetzner Cloud [207] und Microsoft Azure [208] zur Platzierung von Scan-Zielen gewählt. Mit dem Ziel, eine möglichst globale Abdeckung zu erreichen (siehe Kapitel A.1.3), werden somit folgende Hosting-Betriebe bzw. -Produkte mit zugehörigem geografischen Standort ausgewählt:

- Alibaba Cloud ECS [205] — China, Hong Kong
- Amazon EC2 [206] — USA
- BFH Cyber-Security-Lab — Schweiz
- Hetzner Cloud [207] — Deutschland
- JustHost [209] — Russland
- Microsoft Azure [208] — Australien
- Serverspace Cloud [210] — Brasilien
- Ultahost VPS [211] — Singapur
- VPS.us [212] — Nigeria
- VPSServer.com [213] — UK

Somit werden 10 Scan-Ziele aufgebaut. Ein Scan-Ziel betreibt lediglich Port-Mirroring über einen VPN-Tunnel ohne selber aufzuzeichnen. Bezüglich dessen Ressourcen-Anspruch genügt im lokalen Betrieb 1 CPU-Kern sowie 1 GB Arbeitsspeicher. Daher wird aus den Hosting-Produkten jeweils das Preisgünstigste ab 1 GB Arbeitsspeicher ausgewählt und die Auslastung regelmässig geprüft[14].

Nicht alle Hosting-Dienstleistungsunternehmen bieten öffentliche IPv6-Adressen für deren Server an. HOSTAFRICA [214] beispielsweise unterstützt auf Anfrage lediglich IPv4-Adressen, so auch Serverspace [210]. Bei Letzterem wird aufgrund falscher Annahme (IPv6-Unterstützung gegeben) trotzdem ein Scan-Ziel betrieben.

Bei Ultahost [211] und VPSServer.com [213] müssen Support-Anfragen erstellt werden, um öffentliche IPv6-Adressen zu erhalten. Hierbei unterstützen nicht alle deren geografischen Standorte IPv6, sodass im Falle Ultahost die Standortwahl von Indien auf Singapur geändert werden muss.

Weitere Unterschiede zwischen den Server-Bezugsquellen liegen darin, dass die Scan-Ziele entweder direkt öffentliche oder interne IP-Adressen erhalten. Dies hat einen Einfluss auf die zu betrachtende Ziel-Adresse bei der Analyse, die der Adresse des Scan-Ziel-Netzwerk-Interfaces entspricht.

Pro Hosting werden angebotene Schutzmechanismen bestmöglich deaktiviert und/oder mittels Firewall-Regeln sämtlicher Netzwerkverkehr zum Scan-Ziel weitergereicht.

[14] Mit einem eigenen Ansible-Playbook `monitoring.yml` (siehe Tabelle D.1 in Kapitel D) werden die Scan-Ziele regelmässig über den VPN-Tunnel zu deren Erreichbarkeit sowie CPU-, Arbeitsspeicher- und Disk-Auslastung überprüft. Der Betrieb einer Monitoring-Umgebung zur automatisierten Überwachung der Scan-Ziele befindet sich ausserhalb dem Rahmen dieser Arbeit

Der Versuch, ein Scan-Ziel in Festlandchina zu betreiben, schlägt aufgrund der Konto-Verifikation bei Alibaba Cloud fehl. Stattdessen wird dieses mittels Alibaba Cloud in Hong Kong aufgebaut.

Beim Aufbau der Scan-Ziele ermöglichen nicht alle Hosting-Plattformen das Hochladen von ISO-Dateien zur Installation. Einige bieten stattdessen das Bereitstellen von eigenen Images virtueller Maschinen oder lediglich vorgegebene Vorlagen an. Wo Hochladevorgänge scheitern[15] oder nur besagte Vorlagen zur Verfügung stehen, werden diese mit Debian-12-Betriebssystem für die Scan-Ziele gewählt. Danach werden benötigte SSH-Zugänge mittels Public-Key-Authentisierung sichergestellt und VPN-Client-Konfigurationen pro Scan-Ziel erstellt. Daraufhin erfolgt die Konfiguration mittels Ansible, nach welcher das Port-Mirroring mit ERSPAN via VPN-Tunnel in Betrieb ist (siehe Kapitel C.3).

Die Funktionalität und Detektion der Analyse-Umgebung im Internet wird gemäss den generierten Netzwerkpaketen in Kapitel 3.1.4, A.1.4 und D.3.2 verifiziert. Hierbei werden die IP-Adressen der Scan-Ziele entsprechend Tabelle C.1 in Kapitel C.4 als Ziel-Adressen gesetzt. In den zuvor erwähnten Oberflächen der Malcolm-Instanz sind daraufhin zugehörige Detektionen und Verbindungen ersichtlich.

3.4. DNS-Einträge für IPv6-Adressen

Innerhalb von 20 Tagen werden zu IPv6 15 Verbindungen und 4 Detektionen registriert, wobei die Angaben zu IPv4 im einstelligen Millionenbereich liegen (siehe Kapitel 4.2).

Zur Förderung der IPv6-Kommunikation werden während des Aufzeichnungszeitraums am 17. Januar 2025 DNS-Einträge (Typ „AAAA“ [215]) zu den IPv6-Adressen der Scan-Ziele angelegt (nur IPv6, siehe Tabelle C.1 in Kapitel C.4).

Hierzu wird die Domäne `pinelair.com` registriert und jeder IPv6-Adresse einen entsprechenden DNS-Eintrag zugewiesen[16]. Um einen möglichst neutralen Standpunkt zu repräsentieren wird mit `.com` anstelle einer länderspezifischen eine generische Top-Level-Domäne gewählt. Die Namen der Domäne sowie einzelnen Einträge stammen von einem Passphrasen-Generator (hier unter Anwendung der Applikation „KeePassXC“[17] [216]). Dies dient dazu die Scan-Ziele möglichst gleich zu behandeln und einzelne Ziele nicht mit bekannten Einträgen wie `www` oder `mail` höher zu gewichten.

3.5. Benutzeroberflächen zur Analyse

Zur Rekapitulation: Es stehen zwei Web-Benutzeroberflächen über die Malcolm-Instanz zur Analyse der ermittelten Daten zur Verfügung [120, 197, 198]: OpenSearch Dashboards und Arkime.

Mittels OpenSearch Dashboards ist es möglich, eigene Visualisierungen und Dashboards beziehungsweise Übersichten zu erstellen [197]. Um nur zu den Scan-Zielen eingehende Verbindungen zu sehen, bedarf es die Daten nach diesen zu filtern. Dazu werden folgende Filter in Kombination angewendet (IP-Adressen der Scan-Ziele sind, sofern veröffentlicht, in Tabelle C.1 aufgeführt):

- Quell-Adressen (`source.ip`): Keine IP-Adressen der Scan-Ziele
Keine IP-Adressen der Autorschaft
- Ziel-Adressen (`destination.ip`): Sämtliche IP-Adressen der Scan-Ziele

Dieser Filter wird zur wiederholten Anwendung entsprechend abgespeichert (in OpenSearch Dashboards als „Query“ und in Arkime als „View“).

[15] Primär handelt es sich hierbei um Hochladevorgänge, die währenddessen wiederholt abgebrochen und aus Zeitgründen nicht weiter verfolgt werden

[16] Ab dem 20. Februar 2025 befindet sich die hierbei verwendete Domäne `pinelair.com` nicht mehr im Besitz der Autorschaft. Sie war in folgendem Zeitraum im Rahmen dieser Arbeit aktiv: 17. Januar bis 20. Februar 2025

[17] Bei Anlegung eines neuen Eintrags wird für das Passwort-Feld eine Passphrase mit der Wörterliste `eff_large.wordlist` und 11 Wörter generiert. Die ersten zwei Wörter in Kombination bilden den Namen der Domäne, wobei die restlichen Wörter einzelne Subdomänen bilden (eine pro Scan-Ziel, `st007` 🇧🇷 hat keine IPv6-Adresse und erhält somit keinen DNS-Eintrag zugewiesen)

Mittels Zeek-Skript angelegte Daten werden in OpenSearch Dashboards visualisiert. Dies ermöglicht eine grafische Einsicht der Informationen dynamisch zu eigens festgelegten Zeiträumen. Beispielsweise wird die Anzahl an Detektionen pro Minute oder Stunde (je nach Zeitraumgrösse) oder über den gesamten Zeitraum angezeigt. Anteile der angewandten Detektionsmethoden sowie der zugehörigen Intention („good“ / „bad“) wie auch das Vorkommen von IPv4 und IPv6 sind grafisch aufbereitet.

Das hierzu erstellte Dashboard „Scanner Detection“ und zugehörige Visualisierungen in OpenSearch sind über die Option „Dashboards Management“ → „Saved objects“ exportiert und dieser Arbeit beigelegt (siehe `opensearch-objects` in Tabelle D.1 / Kapitel D). Somit wird ein Import besagter Oberflächen in weitere Malcolm-Instanzen über dieselbe Option ermöglicht[18].

Neben den erstellten Visualisierungen beinhaltet das „Scanner Detection“-Dashboard durch Malcolm mitgelieferte Elemente wie Tabellen oder Visualisierungen. Die Abbildungen in Kapitel E.2 im Anhang zeigen eine Momentaufnahme dieses Dashboards.

Durch Selektion einer Detektionsmethode (z.B. `KnownDomain_good`) im Dashboard können die Daten entsprechend gefiltert werden. Dasselbe gilt für unter anderem IP-Adressen, Ports oder AS. Zusätzlich gilt es sich des angewandten Zeitfensters und Filter bewusst zu sein. Am Ende des Dashboards befinden sich zugehörige Log-Einträge mit jeweils weiteren Informationen zur vertieften Analyse.

Bezeichnungen der Scan-Quellen entstammen den Tabellen aus Kapitel 3.1.3. Dies ermöglicht eine Nachvollziehbarkeit der Intention detektierter Scan-Quellen.

3.5.1. Filter anhand Visualisierungen

Einige Visualisierungen enthalten Painless-Skripts u. a. zur Gruppierung der ermittelten Scanner [217–219]. Zeek-Notices in der derzeitigen Anwendung beinhalten die meisten Informationen innerhalb eines Strings im Feld `zeek.notice.msg` [220]. Mittels Painless-Skripts findet ein Parsen oder Modifizieren dieses Strings in der Visualisierung statt.

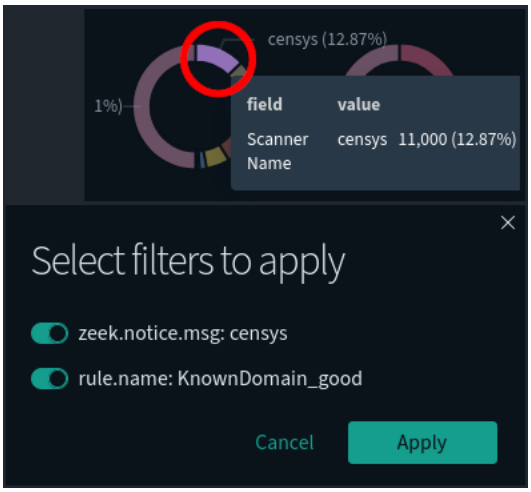


Abbildung 3.8.: OpenSearch Dashboards: Filter bei Selektion in einer Visualisierung mit Feldern `zeek.notice.msg` und `rule.name` (Erster Filter nicht ohne Asteriske anwendbar)

Eine Konsequenz daraus ist, dass nicht mehr direkt durch die Selektion eines Werts in einer Grafik gefiltert werden kann. Wenn zum Beispiel durch eine solche Selektion die Filter `zeek.notice.msg: censys` und `rule.name: KnownDomain_good` entstehen, ist der erste Filter nicht anwendbar (siehe Abbildung 3.8). `zeek.notice.msg` beinhaltet nicht nur besagten Text. In `zeek.notice.msg` befinden sich Werte wie „`good scanner on ...: censys via manualinput, method: reverse resolved domain (censys-scanner.com) in known scanner file`“. Daher sind Asteriske im Filter zu `zeek.notice.msg` anzuwenden. Im besagten Beispiel wäre somit der korrekte Filter `zeek.notice.msg: *censys*` anstelle von `zeek.notice.msg: censys`. Die restlichen Felder können durch Selektionen in den Visualisierungen direkt als Filter angewendet werden.

[18] Für einen kontrollierten Import-Vorgang der Objekte wird empfohlen, hierbei die Import-Optionen „Check for existing objects“ und „Request action on conflict“ zu wählen. Es gilt die Visualisierungen vor dem Dashboard zu importieren

3.5.2. Geografische Merkmale

Aufgrund gewählter geografischer Standorte der Scan-Ziele ist ein Vergleich zwischen den ermittelten Werten von Zeek und Arkime möglich. Der geografische Standort derselben IP-Adresse unterscheidet sich zwischen Arkime und Zeek, da diese unterschiedliche Quellen verwenden [221]. Daraus ergibt sich die Erkenntnis, dass bei unterschiedlichen Standortangaben zu derselben IP-Adresse die Angaben in Arkime eher zutreffen. Dienstleitungsunternehmen, die entsprechende Datenbanken anbieten, versprechen bei ihren Angaben keine 100%ige Genauigkeit [40, 43]. Malcolm verwendet kostenfreie, als ungenauer bezeichnete Datenbanken von MaxMind („GeoLite2“ [222]) [107, 197, 221, 223].

Interne, private IP-Adressen von Scan-Zielen werden mit keiner Standortangabe versehen und können somit nicht in den Angaben-Vergleich miteinbezogen werden. Hierbei ragen die Standortangaben der öffentlichen IPv4-Adressen der Scan-Ziele `st002` , `st007` und `st009` (siehe Tabelle C.1) heraus. `st002` in Nigeria wird von Arkime korrekt angezeigt, wobei Zeek als Standort Russland vermerkt. Analog wird dasselbe bei `st007` festgestellt, dessen Standort Brasilien von Arkime korrekt angezeigt wird, während Zeek wieder Russland aufführt. Für `st009` notiert Arkime nochmals korrekt den Standort Singapur. Zeeks Einträge verweisen hierbei auf Schweden.

Die Visualisierung in Abbildung 3.9[19] zeigt die vermerkten Länder in Form von Codes mit zwei Buchstaben („alpha-2“) des ISO-Standards 3166 [224, 225]. Die Grafik repräsentiert sämtliche IP-Adressen der aufgebauten Scan-Ziele, die während Dezember 2024 einen solchen Code erfahren haben.

Zuvor aufgeführte Unterschiede sind ersichtlich, mit folgenden, zusätzlichen Erkenntnissen:

- Standortangaben zu IPv6-Adressen scheinen nur von Arkime zu stammen
- Nicht zu jeder IP-Adresse werden Standortangaben ermittelt
- Bei Scan-Ziel `st006` wird eine IPv6-Adresse aus dem öffentlichen Adressraum als Interne verwendet, zu welcher ebenfalls eine Standortangabe notiert wird
  - Der Whois-Eintrag zur internen IPv6-Adresse von `st006` gehört zu „Microsoft Singapore Pte. Ltd.“ [73]
- Einige Scan-Ziele haben dieselbe korrekte Standortangabe unter Arkime und Zeek vermerkt

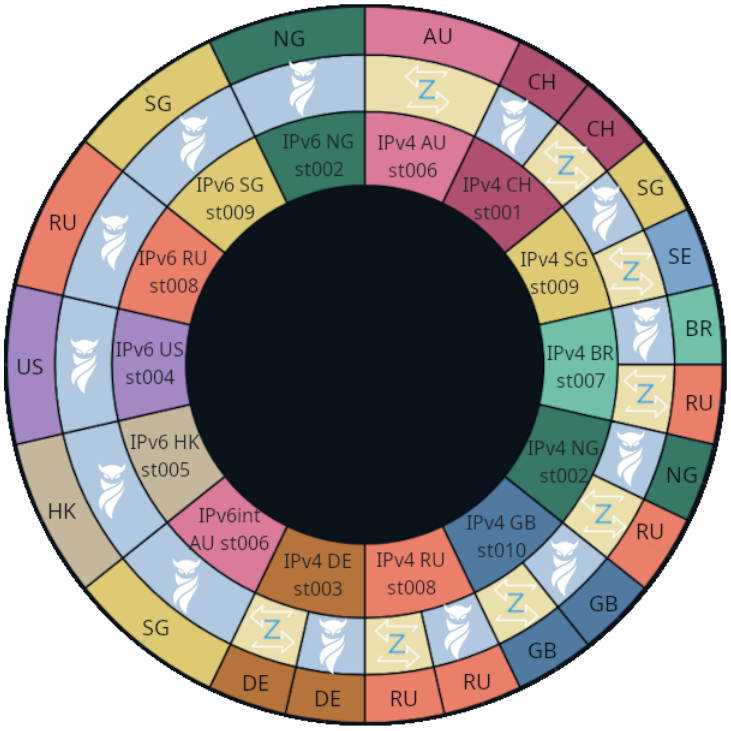


Abbildung 3.9.: Vergleich der Standortangaben zu den Ziel-IP-Adressen der Scan-Ziele zwischen Arkime [107] und Zeek [226]

Aus diesen Gründen sind Standort-relevante Analysen primär unter Arkime durchzuführen. Zusätzlich gilt es zu berücksichtigen, dass hierbei nicht sämtliche detektierten Scan-Quellen berücksichtigt werden können. Im Rahmen dieser Arbeit ist es nicht möglich, die Standortangaben aus Arkime auf die detektierten Quell-IP-Adressen der Zeek-Notices des Zeek-Skripts anzuwenden[20]. Daher werden diese manuell analysiert.

[19] Visualisierung „Country ISO Code vs Event Provider“, siehe `opensearch-objects` in Tabelle D.1

[20] Von einer tieferen Modifikation der Malcolm-Instanz oder des Sensors mit Hedgehog Linux wird abgesehen, um mögliche Fehler oder einen automatisierten Abbau der Modifikationen bei Aktualisierungen zu vermeiden

4. Ergebnisse

Dieses Kapitel zeigt die ermittelten Analyse-Ergebnisse des aufgezeichneten Netzwerkverkehrs der Scan-Ziele. Der betrachtete Zeitraum liegt zwischen dem 27. Dezember 2024 bis und mit dem 9. Februar 2025[21].

Zusätzlich wird aufgrund eines Aufzeichnungsausfalls der Scan-Ziele `st002` und `st003` (siehe Tabelle C.1) folgender Filter hinzugefügt (`zeek.ts` entspricht dem Zeitstempel beziehungsweise „Timestamp“ eines Zeek-Eintrags):
`zeek.ts is not between 2025-01-13T00:00:00Z → 2025-01-17T09:00:00Z`

Damit wird der besagte Zeitraum innerhalb des betrachteten Zeitraums ausgeklammert, um die Daten der Scan-Ziele vergleichen zu können.

Mit Ausnahme von ausgewiesenen, manuellen Analyse-Schritten stammen die Ergebnisse aus der Malcolm-Instanz beziehungsweise Arkime oder OpenSearch Dashboards mit unter anderem Visualisierungen und Auswertungen des Zeek-Skripts (siehe Kapitel 3.1.3 und D). Bei nachfolgend erwähnten Scan-Detektionen handelt es sich um entsprechend protokollierte, detektierte Verbindungen dieses Skripts (Filter `rule.category: ScannerDetection`).

Mittels Arkime kann nach einzelnen Verbindungen gesucht und diese gegebenenfalls als PCAP-Datei exportiert werden. Der PCAP-Export ermöglicht eine tiefere Auseinandersetzung mit Anwendungen wie beispielsweise Wireshark.

Sofern es nicht anders aufgeführt ist, werden stets die Filter aus Kapitel 3.5 angewendet. Somit stammen Quell-IP-Adressen weder von den Scan-Zielen noch der Autorschaft[22]. Betrachtete Ziel-IP-Adressen gehören stets zu den Scan-Zielen.

Zur Rekapitulation: Die Scan-Ziele werden nicht für produktiven Netzwerkverkehr verwendet. Jede Anfrage wird als Scan gewertet. Daraus ergibt sich die Aussage, dass in dieser Arbeit keine Ergebnisse zum Anteil von Scans am allgemeinen Netzwerkverkehr möglich sind. Stattdessen wird auf den Anteil an detektierten Scans inklusive deren Intention eingegangen. Eine zur Scan-Detektion zugehörige Verbindung wird ebenfalls geloggt, aber nicht jede Verbindung führt zu einer Scan-Detektion.

Zusammenfassung und Fazit sind in Kapitel 5 aufgeführt.

4.1. Filter-Aspekte

Dieses Kapitel beinhaltet Punkte zur weiteren Modifikation des angewendeten Filters. Hierbei werden Aspekte beleuchtet, die einen signifikanten Einfluss auf das Gesamtergebnis und daraus gezogene Erkenntnisse ausüben könnten.

Die Anzahl an Scan-Detektionen wird mit zusätzlichem Filter `rule.category: ScannerDetection` dem Dashboard „Scanner Detection“ ausgelesen.

4.1.1. ERSPAN-Skript und verfügbare IP-Versionen

Das ERSPAN-Skript zum Port-Mirroring-Betrieb auf den Scan-Zielen (siehe Kapitel 3.1.1 und D.1) sendet zur Aufrechterhaltung alle 10 Sekunden eine Ping-Anfrage. Auf diese Anfrage gilt es eine Antwort zu erhalten, um die Funktionalität des Skripts zu gewährleisten[23].

[21] Ab diesem Startzeitpunkt sind sämtliche aufgebauten Scan-Ziele verifiziert und deren Aufzeichnungsverhalten fixiert (siehe Ziele in Kapitel A.1.4 sowie Arbeitsjournal in Kapitel A.2)
[22] Dies dient der Ausschliessung von Anfragen zur Verifikation u. a. aus Kapitel 3.1.4
[23] Ohne diese regelmässigen Anfragen erhalten zum aktuellen Zeitpunkt die Scan-Ziele das Port-Mirroring nicht aufrecht

Die Ping-Anfrage muss hierbei auf dem zu spiegelnden Interface erfolgen. Dies schliesst eine Anfrage auf die eigene IP-Adresse aus, da diese auf dem Loopback-Interface erfolgt. Mittels IPv6 ist dies lösbar beispielsweise zur Adresse `ff01::1` (Multicast-Adresse „all nodes“ in Scope „Interface-Local“) [7]. Dadurch wird ein Netzwerkpaket mittels ERSPAN zur Aufrechterhaltung repliziert und aufgrund dessen Scope verworfen (und übt somit keinen Einfluss auf die Aufzeichnungen aus).

Ohne IPv6 erfolgt die Ping-Anfrage zur Gateway-Adresse. Dies da Anfragen zur entsprechenden IPv4-Broadcast-Adresse aus Sicherheitsgründen zu vermeiden sind und von Linux-Systemen standardmässig ignoriert werden [227]. Solche Anfragen zur Gateway-Adresse erfolgen auf dem zu spiegelnden Interface, werden aber nicht verworfen und fliessen somit in die Aufzeichnungen ein.

Scan-Ziel `st007` 🇧🇷 unterstützt als einziges IPv6 nicht. Die Gateway-IPv4-Adresse zu Scan-Ziel `st007` 🇧🇷 wird somit zusätzlich von den Quell-IP-Adressen mittels Filter von der Analyse ausgeschlossen.

4.1.2. Aktualisierungsvorgänge des Betriebssystems

Die Scan-Ziele führen aufgrund deren Konfiguration automatisiert Aktualisierungen derer Debian-Betriebssysteme durch.

Folgende Adressen werden zusammengefasst bei den Aktualisierungsvorgängen mittels HTTP oder HTTPS aufgerufen (ermittelt durch Prüfen der Konfigurationsdateien jeweils unter `/etc/apt`):

- `https://cdn-aws.deb.debian.org/debian`
- `https://cdn-aws.deb.debian.org/debian-security`
- `http://deb.debian.org/debian`
- `http://ftp.debian.org/debian`
- `http://httpredir.debian.org/debian`
- `http://security.debian.org/debian-security`
- `https://tommie.github.io/innernet-debian/debian`

Besagte Aktualisierungen führen ebenfalls zu aufgezeichneten Verbindungen. Mittels DNS-Anfragen[24] und Datenbanken für Passive DNS werden zugehörige bzw. ehemals zugehörige IP-Adressen im betrachteten Zeitraum ermittelt [229–232]. Die Ermittlungen ergeben, dass die hinterlegten Quellen zu Content Delivery Networks (CDNs) führen [233]. Hierbei verwendete AS bilden AS16509 „Amazon.com, Inc.“ (über 18’000 IP-Präfixe) [234] und AS54113 „Fastly, Inc.“ (über 1300 IP-Präfixe) [235].

Aufgrund ihrer hohen Anzahl an IP-Präfixen steigt die Wahrscheinlichkeit, dass bei dem Filtern entsprechend dieser AS Daten von Scans verloren gehen.

Die Scan-Ziele zeichnen nur eingehende Verbindungen auf und rufen die Aktualisierungsquellen mittels HTTP und HTTPS ab. Die Anwendung des Filters auf die Quell-Portnummern 80 und 443 [236] zeigt mehrere detektierte Scan-Quellen[25]. Hierbei werden 1067 Verbindungen und 295 Scan-Detektionen (Zeek Notices) geloggt. Dies beträgt bei 6,5 Millionen Verbindungen und 3,3 Millionen Scan-Detektionen jeweils gerundet einen Anteil von 0,02 % und 0,01 %[26].

Dasselbe gilt für die automatische Zeitsynchronisation mittels NTP-Port 123 [236]. Bei Anwendung des Filters `source.port: 123` werden 1399 Verbindungen und 141 Scan-Detektionen aufgelistet (Anteile gerundet 0,02 % und 0,004 %).

Daraus entsteht die Erkenntnis, dass die Aktualisierungsvorgänge der Debian-Systeme sowie deren Zeitsynchronisation auf die ermittelten Informationen keinen erheblich erkennbaren Einfluss wirken.

[24] Die DNS-Anfragen werden jeweils mittels `dig -t A`, `dig -t AAAA` und `dig -t CNAME` ausgeführt, um IPv4-, IPv6-Adressen und Aliase zu ermitteln [215, 228]

[25] Zusätzlicher Filter `source.port is one of 80, 443`

[26] Angewendete Filter siehe Kapitel 4.3

4.1.3. Microsoft Azure cloud-init

Über 500'000 Verbindungen entstammen einer einzelnen IP-Adresse mit Scan-Ziel `st006` als alleiniges Ziel: `168.63.129.16`[27,28]. Somit handelt es sich um einen Ausreisser, der alleine einen signifikanten Einfluss auf die ermittelten Daten ausübt.

Diese IP-Adresse stammt von Microsoft und ist auf Scan-Ziel `st006` in dessen vordefinierter Debian-Umgebung als DNS-Nameserver hinterlegt [237]. Zudem ist sie dort in der Routing-Tabelle sowie u. a. in der Log-Datei `/var/log/cloud-init.log` vorzufinden. Diese Entdeckungen führen zur Anwendung „cloud-init", die bei Microsoft Azure mit Virtuellen Maschinen mit Linux-Betriebssystemen ausgeliefert wird [238, 239]. Nahezu 100 % von dessen Verbindungen stammen vom TCP-Quellport 32526[27,29]. Hierbei handelt es sich um eine bekannte IP-Adresse sowie einen bekannten Port, der für die Kommunikation von Azure-VMs mit Azure verwendet wird [240].

Aus diesen Gründen wird der derzeit aktiv verwende Filter nachfolgend so ergänzt, dass die Quell-IP-Adresse `168.63.129.16` zusätzlich ausgeschlossen wird.

4.1.4. Probe Attribution (RFC 9511)

Die Detektion der Out-of-Band Probe Attribution gemäss RFC 9511 erfolgt über achttausend mal. Durch zusätzliche Anwendung des Filters `rule.name: ProbeAttribution_OutOfBand` sind die Quell-IP-Adressen in der Visualisierung bzw. Tabelle „Notices - Source IP Addresses" ersichtlich.

Mittels „Discover"-Übersicht in OpenSearch Dashboards kann mit den aktiv angewandten Filter durch Klick oben auf „Save" die aktuelle Suche abgespeichert werden[30]. Nun kann oben unter „Reporting" ein Daten-Export im CSV-Format generiert werden. Die heruntergeladene Datei wird mittels nachfolgendem Befehl nach zu prüfenden URIs gefiltert.

```
1 $ grep -Eo "http[^,]*probing\.txt" On_demand_report_....csv | sort | uniq
```

Quelltext 4.1: Filtern eines Reports aus OpenSearch Dashboards nach URIs mit `probing.txt` zur manuellen Überprüfung

Daraus ergeben sich 41 eindeutige URIs. Der Sensor mit Hedgehog Linux hat diese Adressen bereits selber aufgerufen, jedoch nur den erhaltenen Inhalt geprüft (ohne Ausführung von allfälligem Javascript). Daher wird zur manuellen Prüfung die Anwendung `curl` oder der Tor Browser im sichersten Modus verwendet.

Keine URI entspricht einer korrekt detektierten Ouf-of-Band Probe Attribution. Sechzehn URIs sind der Domäne `showfreevids.com` zu zuordnen und treten mit diversen Subdomänen sowie demselben HTML-/CSS-Code auf. Andere URIs zeigen auf u. a. Informationen zu geparkten Domänen, Login-Masken oder sind nicht mehr zum Prüfungszeitpunkt erreichbar.

Die Prüfung des Strings „contact" im Inhalt der URI per Zeek-Skript scheint nicht auszureichen (siehe Kapitel 3.1.3). RFC 9511 empfiehlt Felder wie „Contact" oder „Description", aber erzwingt keine [15, 241]. Es wird einzig das Format der `probing.txt`-Textdatei gemäss RFC 9116 [96] vorgegeben [15]. Die Antwort der HTTP-Anfrage mit Zeek wird direkt in einem String gespeichert, womit der MIME-Typ nicht ausgelesen werden kann [242]. Eine Validierung der Antwort mittels ABNF-Regeln aus RFC 9116 [96] kann aus Zeitgründen nicht im Zeek-Skript implementiert werden.

[27] Zusätzlicher Filter `source.ip: 168.63.129.16`
[28] Ermittelt mit Visualisierung „Connections - Destination - Sum of Source Bytes" in OpenSearch Dashboards, siehe Tabelle D.1 in Kapitel D
[29] Ermittelt unter „Discover" in OpenSearch Dashboards mit Betrachtung des Felds `source.port`
[30] Der für diese Suche gewählter Name lautet hier „Scanner Detection Probe Attribution Out-of-Band"

Eine mögliche Erklärung einer Fehldetektion dieser Art zeigt die Webserver-Anwendung „nginx“ [243]. Diese ermöglicht in dessen Konfiguration das Umschreiben von angefragten URIs, womit beispielsweise eine Anfrage zu `/.well-known/probing.txt` nach `/hello.txt` umgeleitet werden kann [244]. Enthält hier `hello.txt` den String „contact“, würde dies zusammen mit dem HTTP-Response-Code [168] `200` die Detektion einer Out-of-Band Probe Attribution auslösen (gemäss aktuellem Stand des Zeek-Skripts, siehe Kapitel 3.1.3). Der nachfolgende Quelltext beweist dieses Verhalten auf einem lokal aufgebauten Debian-Server mit frisch installierter Anwendung nginx.

```
$ grep -v '.*#' /etc/nginx/sites-enabled/default | grep -v '^$'
server {
        listen 80 default_server;
        listen [::]:80 default_server;
        root /var/www/html;
        index index.html index.htm index.nginx-debian.html;
        server_name _;
        location / {
                rewrite ^/(.*)$ /hello.txt break;
        }
}
$ ls -al /var/www/html/
total 16
drwxr-xr-x 2 root root 4096 Jan 17 12:26 .
drwxr-xr-x 3 root root 4096 Jan 17 12:24 ..
-rw-r--r-- 1 root root    6 Jan 17 12:26 hello.txt
-rw-r--r-- 1 root root  615 Jan 17 12:24 index.nginx-debian.html
$ sudo systemctl restart nginx
$ curl http://localhost/.well-known/probing.txt -I
HTTP/1.1 200 OK
Server: nginx/1.22.1
Date: Fri, 17 Jan 2025 12:38:34 GMT
Content-Type: text/plain
Content-Length: 6
Last-Modified: Fri, 17 Jan 2025 12:26:47 GMT
Connection: keep-alive
ETag: "678a4c87-6"
Accept-Ranges: bytes
```

Quelltext 4.2: Beispiel einer Konfiguration und Prüfung der URI-Umschreibung bei der Webserver-Anwendung „nginx“ [244]

Aus diesem Grund wird für die nachfolgenden Analysen der aktive Filter so erweitert, dass Detektionen entsprechend `rule.name: ProbeAttribution_OutOfBand` ausgeschlossen werden.

Zur In-Band Probe Attribution (`rule.name: ProbeAttribution_InBand`) erscheint keine Detektion. Somit wird mit der aufgebauten Analyse-Umgebung keine Probe Attribution gemäss RFC 9511 festgestellt.

4.1.5. ZMap

Das IPv4-Identifikations-Feld dient der Fragmentierung zugehöriger Netzwerkpakete [83]. Pakete, die nicht zu fragmentieren sind, können bei diesem Feld einen beliebigen Wert beinhalten [83]. Wie in Kapitel 2.1.4 beschrieben, verwendet ZMap dieses Feld zu dessen Identifikation mit einem statischen Wert von `54321` [14, 27].

Bei einer zufälligen Wahl eines 16-Bit-Werts kann dieser theoretisch mit einer Wahrscheinlichkeit von $1/(2^{16})$ beziehungsweise $1/65536$ dem Wert `54321` entsprechen. Bezogen auf 6,5 Millionen Verbindungen würden somit theoretisch abgerundet 99 davon den Wert `54321` im IPv4-Identifikations-Feld enthalten, unabhängig von ZMap.

Effektiv wird dieser Wert 1'583'986 mal detektiert, was einen signifikant grösseren Anteil an den 6,5 Millionen Verbindungen bildet. Aus diesem Grund wird nachfolgend davon ausgegangen, das sämtliche ZMap-Detektionen effektiv von ZMap stammen.

4.1.6. Mehrfachdetektionen

Jede Detektion einer Scan-Quelle wird einzeln notiert (siehe Kapitel 3.1.3). Stellt dieselbe Quelle eine Anfrage und triggert dabei mehrere Detektionsmethoden, wird jede Detektion einzeln geloggt (z.B. zwei Detektionen bei `ZMap` und `KnownIP_good`, wenn von einer bekannten IP-Adresse ein ZMap-Scan detektiert wird[31]).

Die Tabelle „Scanner Detection Top 20 Source IP - By Method" in OpenSearch Dashboards[32] zeigt dies für die aktivsten Scan-Quellen. Hierbei gibt es auch IP-Adressen, zu denen die Detektionen `ZMap` und `KnownIP_bad` geloggt werden (siehe IP-Adresse `193.68.89.3` in Abbildung 4.2). In diesem Fall wird mit der Anwendung von ZMap ausgewiesen, dass ein Scan ausgeführt wird, die Scan-Quelle jedoch in Verbindung mit Malware oder sonstigen bösartigen Aktivitäten liegt.

Angenommen eine Scan-Quelle führt ständig mittels ZMap Scans aus: Ist diese seit Aufzeichnungsbeginn in der Tabelle mit bekannten, als bösartig eingestuften Quellen, würden sich die Detektionen gegenseitig aufheben. Wird eine Scan-Quelle im späteren Verlauf zur besagten Tabelle eingetragen, wäre die Anzahl an ZMap-Detektionen höher als z.B. welche mit `KnownIP_bad`. Dieser Fall ist in der nachfolgenden Abbildung visualisiert.

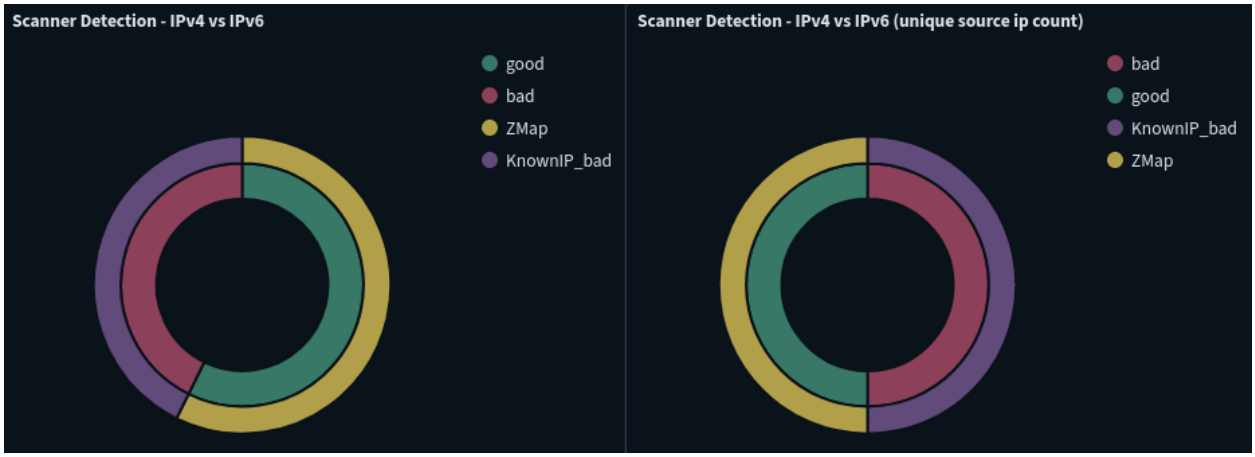


Abbildung 4.1.: Ausschnitt aus dem Dashboard „Scanner Detection" mit Vergleich zwischen Detektionen pro Anfrage einer IP-Adresse (selbe Adresse mehrmals gezählt) gegenüber pro IP-Adresse (selbe Adresse einmalig pro Detektionstyp gezählt), gefiltert mit Quell-IP-Adresse `193.68.89.3`, die im späteren Verlauf zur Tabelle mit bekannten, als bösartig eingestuften Quellen hinzugefügt wird

[31] Eine bekannte Quell-IP-Adresse bedeutet in dieser Arbeit, dass diese in den Tabellen des Zeek-Skripts gemäss Kapitel 3.1.3 eingetragen ist

[32] Siehe Tabelle D.1 in Kapitel D

Bei einzelnen Quell-IP-Adressen steigt mit der Anzahl an Verbindungen die Wahrscheinlichkeit, dass ein Netzwerkpaket die Detektion von ZMap gemäss vorherigem Kapitel auslöst. Abbildung 4.2 zeigt eine mögliche ZMap-Fehldetektion bei der IP-Adresse `89.248.163.4`.

Werden in diesem Fall die IP-Adressen einmalig pro Detektionstyp gezählt (siehe Abbildung 4.1), fällt eine ZMap-Fehldetektion höher ins Gewicht.

Scanner Detection Top 20 Source IP - By Method

source.ip: Desce...	rule.name: Desc...	Count
89.248.163.25	KnownIP_good	51,577
193.68.89.3	ZMap	22,775
193.68.89.3	KnownIP_bad	15,663
89.248.163.4	KnownIP_good	36,637
89.248.163.4	ZMap	1

Abbildung 4.2.: Ausschnitt aus Tabelle „Scanner Detection Top 20 Source IP - By Method“ , im Zeitraum von 23. Dezember 2024 bis 22. Januar 2025

Solche, mehrfach detektierten Scan-Quellen erfahren je nach Visualisierung eine höhere Gewichtung. Das Beispiel in Abbildung 4.1 zeigt, dass bei einer einmaligen Zählung die Quell-Adresse einmalig pro Detektionsmethode auftritt (selbe Adresse befindet sich je ein mal unter `ZMap` und `KnownIP_bad`). Bei einer Aufteilung gemäss der Intention „good“ und „bad“ hebt sich eine Quelle, zu der beide Werte ermittelt werden, gegenseitig auf (siehe Abbildung 4.1 auf der rechten Seite).

Eine weitere Perspektive bieten die Visualisierungen „Scanner Detection Scanner Names - By good and bad“ sowie „Scanner Detection Scanner Names - By good and bad (unique source ip count)“ in OpenSearch Dashboards bzw. Tabelle D.1. Diese zeigen pro Intention ein eigenes Diagramm, bei welchen je nach Visualisierung die Quell-IP-Adresse einmalig oder pro Scan gezählt wird. Auch hier kann eine Quell-IP-Adresse mehrfach vorkommen, wenn zum Beispiel bei der ZMap-Detektion die Quelle mit Namen „unknown“ und dem zugehörigen Namen in der Tabelle bekannter IP-Adressen geloggt wird.

Eine Aufteilung der Visualisierungen in Detektionsmethoden zeigt pro Methode ein eigenes Diagramm, in welchem einzeln betrachtet keine mehrfache Detektion vorkommen kann[33].

Wie zu Beginn des Kapitels erwähnt, zeigt die Tabelle „Scanner Detection Top 20 Source IP - By Method“ die aktivsten Scan-Quellen, inklusive Mehrfachdetektionen (siehe Abbildung 4.2). Die Anzahl der Scan-Detektionen pro IP-Adresse kann der Tabelle „Notices - Source IP Addresses“ entnommen werden, wobei die Top 20 Quell-Adressen im betrachteten Zeitraum im Durchschnitt gerundet zwanzigtausend Anfragen durchführen. 20’000 Scans ergeben bei gerundet 3,3 Millionen einen Anteil von 0,61 %.

Dieselbe Tabelle zeigt unter der IP-Adresse `89.248.163.200` die meisten Mehrfachdetektionen mit derselben Intention. Hierbei werden 8716 Detektionen der Methode `KnownIP_good` und 7149 der Methode `KnownFQDN_good` zugeordnet.

Jede Detektion basiert jedoch auf einem anderen Anhaltspunkt, womit jede Scan-Ausweisung (z.B. per Netzwerkpaketinhalt, DNS-Eintrag oder IP-Adresse) in dieser Arbeit als eigenständige Scan-Detektion geloggt wird. Ausnahme bilden IP-Adresse und -Subnetz. Nun existieren in der Tabelle bekannter Scanner-IP-Adressen Einträge, die auch in der Tabelle bekannter Scanner-IP-Subnetze vorkommen. Das Zeek-Skript loggt somit bei einer entsprechenden IP-Adresse zwei Detektionen. Dieses Verhalten wird angepasst, jedoch erst nach dem auszuwertenden Zeitraum (siehe Zeilen 27, 28 und 32 in Quelltext D.3 in Kapitel D.3)[34]. Daher gilt es dies für diese Auswertung wie nachfolgend zu korrigieren.

[33] Siehe Visualisierungen „Scanner Detection good - By Method“ und „Scanner Detection bad - By Method“ in OpenSearch Dashboards bzw. Tabelle D.1

[34] Die Anpassung führt dazu, dass bei einer erfolgreichen Detektion aufgrund der IP-Adresse keine Subnetz-Detektion mehr durchgeführt wird. Eine Zugehörigkeit zu einem bekannten Subnetz wird demnach nur noch geprüft, falls die direkte Detektion der IP-Adresse nicht erfolgreich ist. Dadurch wird eine doppelte Detektion aufgrund der IP-Adresse ausgeschlossen

Es wird eine Textdatei erstellt, die IP-Adressmuster entsprechend bekannter Subnetze enthält. Anhand dieser wird die Tabelle mit bekannten IP-Adressen durchsucht und zur nachfolgenden Auswertung verwendet.

```
# get known ip addresses that match patterns from known subnets
while read line; do
grep $line zeek/knownscannersip.table >> knownscannersip_subnetsmatch.table
done < <(cat knownsubnetpatterns.txt)
# get scanner names with matching patterns that have been double counted
   because of their ip address
awk '{print $1}' knownscannersip_subnetsmatch.table | sort | uniq
abuseiocid_1115753
abuseiocid_1143693
abuseiocid_1143694
abuseiocid_1143695
abuseiocid_1157698
abuseiocid_377535
censys
recyber
ripeatlas_probe
shadowwhisperer_malware_hackers
shadowwhisperer_scanner
```

Quelltext 4.3: Analyse der Mehrfach-Vorkommen bezüglich bekannter IP-Adressen und IP-Subnetze

Anhand der Ausgabe des vorherigen Quelltexts wird zur Korrektur in OpenSearch Dashboards ein Filter in der Sprache „Query DSL“ angelegt, der Quelltext 4.4 entspricht. Die Ergebnisse dieses Filters werden dann entsprechend ausgeschlossen.

```
{"query": { "bool": { "must_not": [{
"regexp": {
  "zeek.notice.msg": ".*abuseiocid_1115753
      .*subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*abuseiocid_1143693
      .*subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*abuseiocid_1143694
      .*subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*abuseiocid_1143695
      .*subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*abuseiocid_1157698
      .*subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*abuseiocid_377535
      .*subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*censys.*subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*recyber.*subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*ripeatlas_probe.*
      subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*
      shadowwhisperer_malware_hackers.*
      subnet.*"
}},{
"regexp": {
  "zeek.notice.msg": ".*
      shadowwhisperer_scanner.*subnet.*"
}}],
"minimum_should_match": 1
}}}
```

Quelltext 4.4: Filter in OpenSearch Dashboards anhand Quelltext 4.3 zur Korrektur zu vieler Detektionen (IP-Adresse in Tabellen bekannter IP-Adressen und Subnetze gleichzeitig)

4.2. IPv6

Die IPv6-Adressen der Scan-Ziele erfahren Kommunikationen von lokalen oder aus demselben Subnetz stammenden IPv6-Adressen [7, 245]. Diese werden dem aktiv verwendeten Filter angefügt, sodass diese Quell-IPv6-Adressen ausgeschlossen werden. Die Angabe gesamter Netzbereiche wie z.B. dem Subnetz für Link-Local Unicast-Adressen `fe80::/10` [7] oder für Unique-Local-Adressen `fc00::/7` [245] ist hierbei möglich [246].

Der Zeek-Zeitstempel-Filter (`zeek.ts`) aus dem Anfang von Kapitel 4 wird in diesem Abschnitt nicht angewendet. Zusätzlich wird hier der Filter um `network.type: ipv6` erweitert, um lediglich IPv6-Netzwerkverkehr zu betrachten.

Zusätzlich wird der angewendete Filter zu den Ziel-IP-Adressen modifiziert. Hierbei werden anstelle dem Einschliessen der Scan-Ziel-IP-Adressen die lokalen IPv6-Subnetze aus dem Beginn dieses Kapitels sowie IPv6-Link-Local-Multicast-Adressen ausgeschlossen[35]. Das Resultat zeigt IPv6-Netzwerkpakete, die nicht direkt an die IPv6-Adresse der Scan-Ziele gesendet, aber dennoch gesehen und aufgezeichnet werden. Im Dashboard „Scanner Detection“ sind nun zwei weitere IPv6-Zieladressen ersichtlich, zu welchen Scans detektiert werden.

Die erste dieser IPv6-Zieladressen lautet `2002:b061:c0f7::b061:c0f7`.
Diese Adresse wird für IPv6-Kommunikationen über IPv4-Netzwerke verwendet („6to4“, mit zugewiesenem Adressbereich `2002::/16`) [246, 247]. Sämtliche Scan-Detektionen zu dieser Adresse werden Shodan mit der Quell-Adresse `2604:a880:4:1d0::294:3000` zugeschrieben. Die nachfolgende Abbildung zeigt ein Beispiel eines einzelnen, mittels Arkime extrahierten, Netzwerkpakets. Die IPv4-Adresse `192.88.99.1` entspricht der Standard-Anycast-Adresse eines 6to4-Relay-Routers [248].

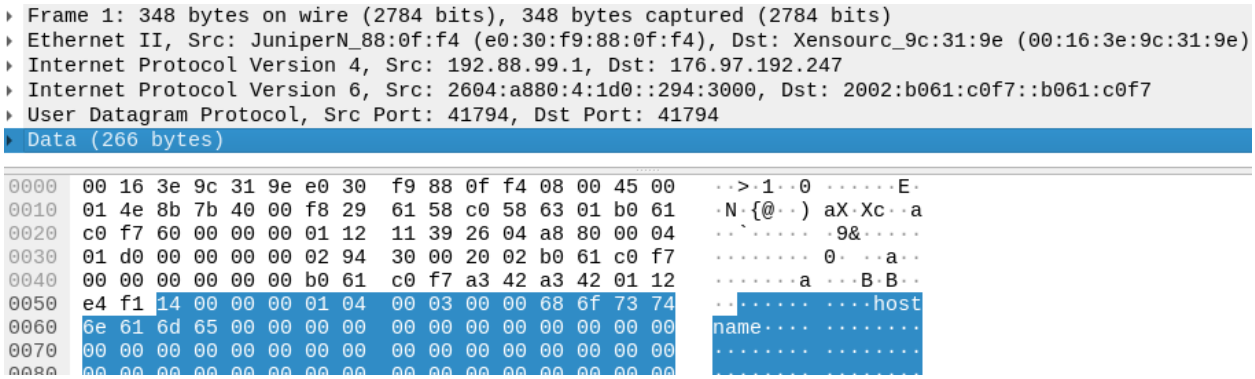


Abbildung 4.3.: Detektierter Scan von Shodan an Scan-Ziel `st002`, übermittelt mit 6to4

Die Abbildung 4.3 zeigt ebenfalls, dass die IPv4-Adresse des Scan-Ziels `st002` als Zieladresse hinterlegt ist. Die Präfix-Zuweisung aus RFC 3056 zeigt, dass die IPv4-Adresse in der 6to4-IPv6-Adresse enthalten ist (`2002:V4ADDR::/48`)[36] [247]. Daraus bildet sich die Erkenntnis, dass diese 6to4-Kommunikationen von Shodan prüfen, ob `st002` als 6to4-Router fungiert [247].

Die Zweite der zuvor erwähnten Zieladressen lautet `2a06:f901:4001:100:1:0:fd9e:f283`.
Diese IPv6-Adresse gehört zum selben Subnetz des Scan-Ziels `st002` . Hierfür detektierte Scans stammen aus dem IPv6-Subnetz `2a06:4880::/32` von Driftnet bzw. `internet-measurement.com`[37].
Die tiefere Betrachtung einer zugehörigen Verbindung mittels Arkime-Extraktion zeigt, dass die MAC-Adresse nicht zum Scan-Ziel `st002` gehört. Somit handelt es sich hierbei um Scans einer Kompo-

[35] Konkret sind dies folgende IPv6-Subnetze: `fe80::/10` [7], `fc00::/7` [245] und `ff02::/16` [7]

[36] Die Umrechnung der IPv4-Adresse des Scan-Ziels `st002` `176.97.192.247` in einen Hexadezimal-Wert ergibt `b061c0f7`. Dieser Wert ist in der hier abgebildeten IPv6-Zieladresse enthalten

[37] Der zugehörige Eintrag in der RIPE-Datenbank verweist auf `internet-measurement.com` zur Beschreibung der Scans und `driftnet.io` zur Dateneinsicht [178, 249] [250]

nente im selben Subnetz, die von st002 aufgezeichnet werden, aber nicht zu st002 gehören. Somit wird diese Ziel-Adresse in diesem Abschnitt aus den Ziel-IP-Adressen ausgeschlossen.

Der modifizierte Ziel-IP-Adressen-Filter wird nun wieder auf den vorherigen Stand gebracht, bei dem die IP-Adressen zu den Scan-Zielen gehören. Zusätzlich wird dieser Filter um den Adressbereich 2002:b061:c0f7::/48 (6to4-Adresse zu st002) ergänzt [247].

Im Zeitraum zwischen dem 27. Dezember 2024 bis und mit dem 16. Januar 2025 treten alleine mit IPv6 sechzehn Verbindungen auf, wovon vier von detektierten Scan-Quellen stammen.

Eine einzelne Scan-Detektion stammt von der RIPE Atlas Probe #1007148 [251] zum Scan-Ziel st003 . Bei den restlichen drei der erkannten Scans zum Scan-Ziel st006 wird anhand den Daten von Collins [18] die Scan-Quelle „mpgde“ vermerkt. Hierbei verweist deren Autorschaft als Quelle auf die Webseite der Max-Planck-Gesellschaft [252]. Abfragen bei der RIPE-Datenbank zu den detektierten „mpgde“-IPv6-Adressen zeigen zur Technischen Universität München [253]. Derselben Universität entstammen die Arbeiten von Trapickin und Gasser aus Kapitel 2.1.

Seit dem 17. Januar 2025 existieren DNS-Einträge zu den IPv6-Adressen der Scan-Ziele[38]. Seitdem werden neben den Shodan-6to4-Scans weitere IPv6-Scans alleine von der Scan-Quelle „mpgde“ detektiert. Zusätzlich zum Scan-Ziel st006 wird nun auch die IPv6-Adresse von st008 von demselben IPv6-Subnetz aus angefragt.

Total werden im gesamten, betrachteten Zeitraum vom 27. Dezember 2024 bis und mit dem 9. Februar 2025 494 IPv6-Verbindungen und 279 detektierte IPv6-Scans geloggt. Sämtliche Scan-Detektionen hierzu finden im Zeitraum vom 8. bis und mit dem 22. Januar 2025 statt. Detektionen mit Intention „bad“ werden keine ermittelt. Zusammengefasst werden zu IPv6 folgende Scans detektiert:

- 261 Scans von Shodan zu Scan-Ziel st002
  auf diversen Ports mittels 6to4
- 17 Scans von der Scan-Quelle „mpgde“ zu Scan-Zielen st006 und st008
  in Form von ICMP-Echo-Request-Paketen
- 1 Scan von der RIPE Atlas Probe #1007148 [251] zu Scan-Ziel st003
  in Form einer DNS-Anfrage auf UDP-Port 53

Bezüglich dem Auftreten der Scan-Ziel-IPv6-Adressen in Hitlisten von Gasser et al. aus Kapitel 2.1.2 sind lediglich einzelne, zugehörige Netzwerkbereiche vorzufinden [39, 254, 255].

Die nachfolgende Tabelle zeigt den Vergleich der Anteile von IPv4- und IPv6-Netzwerkpaketen[39]:

Tabelle 4.1.: Vergleich zwischen aufgezeichneten Daten zu IPv4 und IPv6

Internet Protokoll	IPv4	IPv6	Total
Verbindungen	7’010’853 (99,993 %)	494 (0,007 %)	7’011’347
Scan-Detektionen	3’605’596 (99,9923 %)	279 (0,0077 %)	3’605’875
Einmalig gezählte Quell-IP-Adressen detektierter Scan-Quellen	19’400 (99,9794 %)	4 (0,0206 %)	19’383

[38] Die hierbei verwendete Domäne `pinelair.com` befindet sich seit dem 20. Februar 2025 nicht mehr im Besitz der Autorschaft

[39] Der hier angewendete Filter entspricht dem aus Kapitel 4.3 mit folgenden Ausnahmen: Keine `zeek.ts`-Filterung und kein Ausschluss von Out-of-Band-Probe-Attribution-Detektionen. Für IPv4-Pakete wird der Filter `network.type: ipv4` angefügt, für IPv6 `network.type: ipv6`

4.3. Allgemeine Verbindungsmerkmale

Aufgrund der Ausführungen in Kapitel 4.1 und 4.2 ergibt sich zusammengefasst folgender Filter zur Auswertung der Daten im Dashboard „Scanner Detection“:

- Zeitraum: 27. Dezember 2024 0 Uhr bis 10. Februar 2025 0 Uhr
  Ausgeklammert: `zeek.ts is not between 2025-01-13T00:00:00Z → 2025-01-17T09:00:00Z`
  Siehe Beginn von Kapitel 4
- Quell-IP-Adressen enthalten keine
  - IP-Adressen der Scan-Ziele oder der Autorschaft
    Siehe Kapitel 3.5 und Tabelle C.1
  - Gateway-IPv4-Adresse `92.246.131.1` von Scan-Ziel `st007`
    Siehe Kapitel 4.1.1
  - Azure-IP-Adresse `168.63.129.16` von Scan-Ziel `st006`
    Siehe Kapitel 4.1.3
  - IPv6-Adressen aus den Subnetzen der Scan-Ziele
    Siehe Kapitel 4.2
  - Lokal verwendete IPv6-Adressen der Bereiche `fe80::/10` [7] oder `fc00::/7` [245]
    Siehe Kapitel 4.2
- Ziel-IP-Adressen gehören zu den Scan-Zielen
  Siehe Kapitel 3.5
  Inklusive 6to4-Adressbereich zum Scan-Ziel `st002` `2002:b061:c0f7::/48`, siehe Kapitel 4.2
- Ausschluss der Detektionsmethode für Out-of-Band Probe Attribution gemäss RFC 9511 (`rule.name: ProbeAttribution_OutOfBand`)
  Siehe Kapitel 4.1.4
- Ausschluss von IP-Subnetz-Detektionen bei Scan-Quellen, die gleichzeitig in den Tabellen bekannter IP-Adressen und -Subnetze vorkommen (Filter entsprechend Quelltext 4.4)
  Siehe Kapitel 4.1.6
- Zeek-Notices bzw. Scan-Detektionen des Zeek-Skripts (`rule.category: ScannerDetection`)
  Wird nicht zur Betrachtung allgemeiner Verbindungen angewendet, also bei nicht Scan-Detektions-relevanten Aussagen deaktiviert und bei Scan-Detektion-Betrachtungen aktiviert
  Siehe Kapitel 3.1.3

Sofern es nicht anders aufgeführt ist, werden nachfolgend stets diese Filter entsprechend angewendet.

Zeek loggt beobachtete Verbindungen in einer eigenen Log-Datei [164, 256]. Diese fliessen während der Verarbeitung in die Scan-Detektion mit ein. Im beobachteten Zeitraum werden von Zeek **6’479’989 Verbindungen**[40] aufgeführt. Die Anzahl an **Zeek-Notices der Kategorie „ScannerDetection“** beträgt währenddessen **3’300’675**[41].

Total betragen die aufgezeichneten Quell-Pakete zu den Scan-Zielen aufgerundet 400 Megabytes[42]. Davon beanspruchen die IPv4-Pakete zu den Scan-Zielen jeweils bei `st006` 15,96 %, `st005` 13,75 %, `st002` 13,13 %, `st003` 10,81 % und `st007` 10,63 %. Die restlichen IPv4-Ziel-Adressen machen je einen Anteil von 5 bis 8,5 % aus[42]. Bezüglich IPv6 wird allein die 6to4-Verbindung zur IP `2002:b061:c0f7::b061:c0f7` mit einem höheren Anteil als 0 % (0,008 %) aufgeführt[42].

[40] Ermittelt mit Visualisierung „Connections - Log Count“ in OpenSearch Dashboards
[41] Ermittelt mit Visualisierung „Notices - Log Count“ in OpenSearch Dashboards mit zusätzlichem Filter `rule.category: ScannerDetection`
[42] Ermittelt mit Visualisierung „Connections - Destination - Sum of Source Bytes“ in OpenSearch Dashboards, siehe Tabelle D.1 in Kapitel D

Die Tabellen und Visualisierungen in den Abbildungen 4.4, 4.5 und 4.6 zeigen die Anteile angewendeter Detektionsmethoden. Hierbei gilt es die Ausführungen zu Mehrfachdetektionen in Kapitel 4.1.6 zu beachten.

In Abbildung 4.5 wird jede Scan-Detektion gezählt. Die Visualisierung in Abbildung 4.6 zählt stattdessen jede Quell-IP-Adresse pro Detektionsmethode einmalig.

Weitere Aspekte zur Intention der Scan-Quellen werden in Kapitel 4.7 erläutert.

Notices - Notice Type

Notice Category	Notice Subcategory	Count
ScannerDetection	ZMap	1,583,986
ScannerDetection	KnownIP_good	1,100,660
ScannerDetection	KnownDomain_good	398,700
ScannerDetection	KnownIP_bad	172,472
ScannerDetection	KnownFQDN_good	44,810
ScannerDetection	KnownDomain_bad	43
ScannerDetection	KnownFQDN_bad	4

Abbildung 4.4.: Tabelle „Notices - Notice Type“ in OpenSearch Dashboards

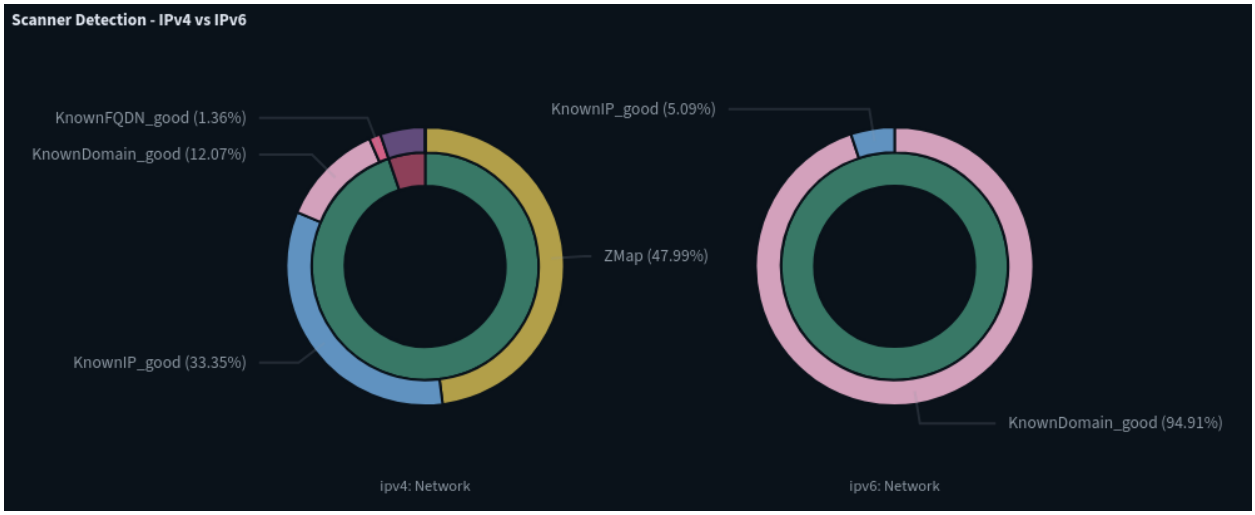


Abbildung 4.5.: Visualisierung „Scanner Detection - IPv4 vs IPv6“ in OpenSearch Dashboards (Modifizierte Darstellung ohne Inhaltsanpassung)

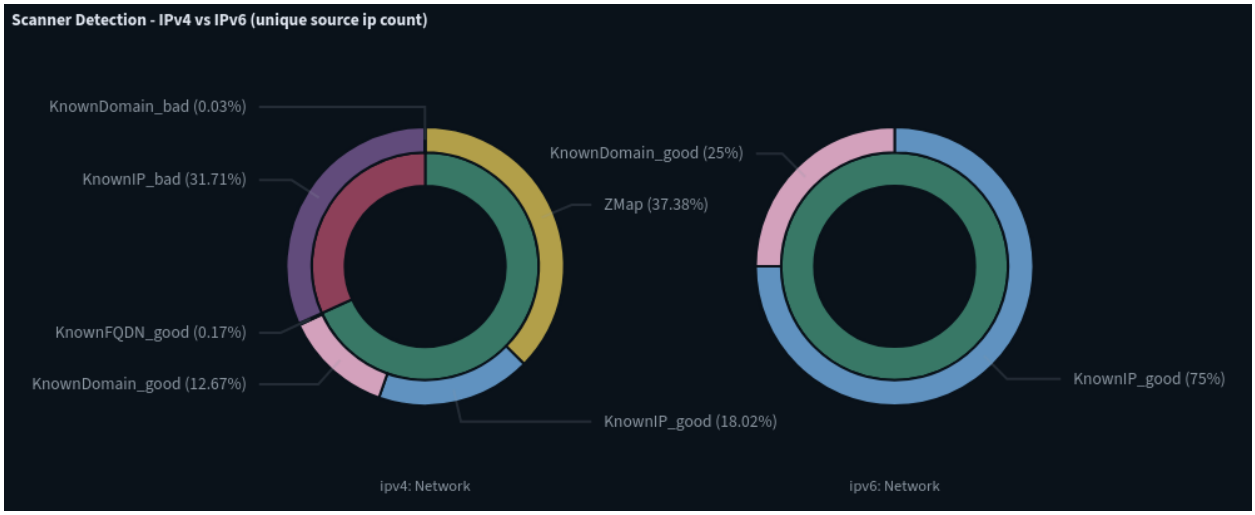


Abbildung 4.6.: Visualisierung „Scanner Detection - IPv4 vs IPv6 (unique source ip count)“ in OpenSearch Dashboards (Modifizierte Darstellung ohne Inhaltsanpassung)

4.4. Kontaktierte Ports

Im Bezug zu Port-Angaben werden diese von Zeek auch bei ICMP verwendet [257]. Der ICMP-„Message Type“ ist dabei im Quell-Port angegeben, während der ICMP-„Message Code“ im Ziel-Port abgebildet wird [257].

Die modifizierte Visualisierung „Notice - Destination Port“ in Abbildung 4.7 zeigt die meist vertretenen Ziel-Ports ermittelter Scan-Detektionen. Bei „Port 0“ handelt es sich hauptsächlich um ICMPv4-„Echo Request“-Pakete). Nachfolgend werden die zehn meist kontaktierten Ports detaillierter betrachtet.

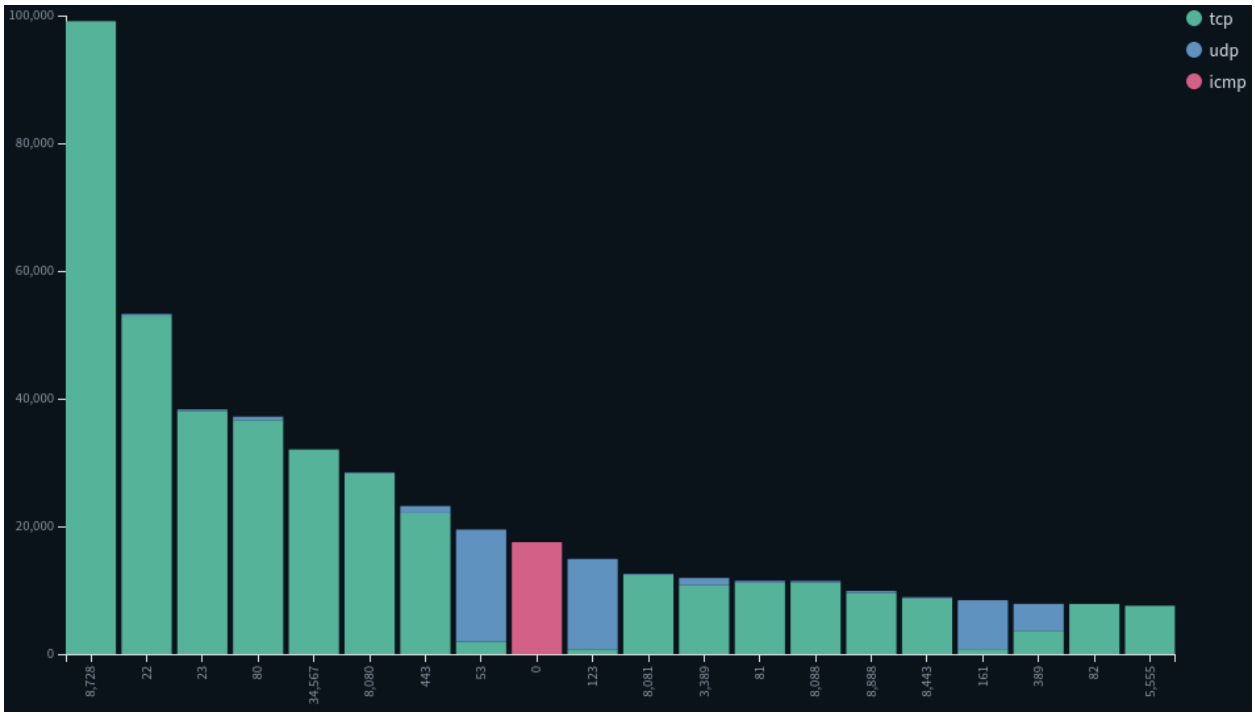


Abbildung 4.7.: Meist aufgerufene Ports bei detektierten Scans (Modifizierte Visualisierung „Notice - Destination Port“ mit zusätzlicher Einteilung entsprechend dem zugehörigen Protokoll)

In der „Discover“-Übersicht in OpenSearch Dashboards können nach Anwendung der Filter inklusive Ziel-Port und Protokoll (z.B. `destination.port: 80` und `protocol: tcp`) weitere Felder betrachtet werden. Das Feld `zeek.conn.conn_state_description` zeigt die von Zeek beobachteten Verbindungsstatus. Ein möglicher Status lautet „Connection attempt seen, no reply“.

Weitere Betrachtungen der Ziel-Ports mit Fokus auf den Anteil entsprechender Scanner ergeben nachfolgende Erkenntnisse in Tabelle 4.2 ab der nächsten Seite[43,44].

[43] Die Visualisierung „Destination Ports Top 20 Scanner portions“ (siehe Tabelle D.1) ist in Abbildung E.2 im Anhang dieser Thesis einsehbar. Zusätzlich werden zur Ermittlung weitere Filter für Port-spezifische Aussagen angewendet (beispielsweise `destination.port: 8728` oder Äquivalent in Arkime). Aus Zeit- und Ressourcengründen werden Netzwerkpakete zu den Ports stichprobenartig untersucht. Es könnten sich Daten unter den Aufzeichnungen befinden, die den nachfolgenden Bemerkungen nicht entsprechen

[44] Berechnet wird der Anteil an Scan-Detektionen anhand der Anzahl an Scan-Detektionen zum spezifischen Port und der Anzahl Detektionen bzw. Zeek-Notices der Kategorie „ScannerDetection“ (3’300’675) aus Kapitel 4.3

Tabelle 4.2.: Analyse der Top 10 Ziel-Ports detektierter Scans

Port	Anteil an Scan-Detektionen	Detektierte Scanner (Top 3)	Bemerkungen
8728 100 % TCP	3,00 %	1 ZMap 2 Shodan 3 bufferoverrun [196]	Ausschliesslich TCP-SYN-Segmente ● Bei IANA registrierter Dienst: nicht registriert [236] ● Keine Detektionen mit IPv6 ● Kein Scan-Ziel erfährt zu diesem Port signifikant mehr Anfragen 👎 0,01 % Scan-Detektionen mit Intention „bad“
22 99,45 % TCP 0,55 % UDP	1,61 %	1 Hosts aus ShadowWhisperer Malware-Hackers-Liste [93] 2 ZMap 3 Hosts aus ShadowWhisperer Scanner-Liste [93]	Neben TCP-SYN-Segmenten auch TCP-Inhalte mit SSH-Protokoll UDP-Datagramme mit String `nxp-scan` in Inhalt entdeckt Weiteres zu SSH siehe „JA4SSH“ in Kapitel 4.6 ● Bei IANA registrierter Dienst: SSH [236] ● Keine Detektionen mit IPv6 ● Scan-Ziele `st002` 🇳🇬, `st003` 🇩🇪, `st005` 🇭🇰 und `st007` 🇧🇷 erfahren zu diesem Port das dreifache an Kommunikationen gegenüber den restlichen Scan-Zielen 👎 61,85 % Scan-Detektionen mit Intention „bad“
23 99,34 % TCP 0,66 % UDP	1,16 %	1 Hosts aus ShadowWhisperer Malware-Hackers-Liste [93] 2 ThreatFox-IoC-Eintrag zu Botnet, Malware „BianLian“ (ID „1190455“) [258] 3 ZMap	Bzgl. TCP ausschliesslich SYN-Segmente UDP-Datagramme mit String `nxp-scan` in Inhalt entdeckt ● Bei IANA registrierter Dienst: Telnet [236] ● Eine einzelne Detektion mit IPv6 Shodan zu Scan-Ziel `st002` 🇳🇬 mittels 6to4, siehe Kapitel 4.2 ● Scan-Ziel `st004` 🇺🇸 erfährt das fünffache an Verbindungen zu diesem Port gegenüber den restlichen Zielen (Ausnahme `st008` 🇷🇺: Erfährt 1,47 % der restlichen Ziele) 👎 69,11 % Scan-Detektionen mit Intention „bad“

Analyse der Top 10 Ziel-Ports detektierter Scans Fortsetzung

Port	Anteil an Scan-Detektionen	Detektierte Scanner (Top 3)	Bemerkungen
80 98,53 % TCP 1,47 % UDP	1,13 %	1 ZMap 2 Hosts aus ShadowWhisperer Scanner-Liste [93] 3 Hosts aus ShadowWhisperer Malware-Hackers-Liste [93]	TCP-Verkehr beinhaltet neben SYN- auch ACK-Segmente und HTTP-Daten mit „GET“- und „HEAD“-Requests UDP-Datagramme enthalten u. a. HTTP-Anfragen oder Shell-Befehle zum Download und Ausführung von Shell-Skripts (siehe Abbildung E.1 im Anhang) Weiteres zu HTTP siehe „JA4H“ in Kapitel 4.6 ● Bei IANA registrierter Dienst: HTTP [236] ● Eine einzelne Detektion mit IPv6 Shodan zu Scan-Ziel `st002` mittels 6to4, siehe Kapitel 4.2 ● Scan-Ziel `st007` erfährt mit 4537 am meisten Anfragen, `st008` mit 2957 am wenigsten 👎 3,95 % Scan-Detektionen mit Intention „bad“
34567 100 % TCP	0,97 %	1 ZMap 2 Hosts aus ShadowWhisperer Malware-Hackers-Liste [93] 3 ThreatFox-IoC-Eintrag zu Botnet, Malware „AsyncRAT“ (ID „159805“) [259]	Ausschliesslich TCP-SYN-Segmente ● Bei IANA registrierter Dienst: „dhanalakshmi.org EDI Service“ [236] ● Keine Detektionen mit IPv6 ● Kein Scan-Ziel erfährt zu diesem Port signifikant mehr Netzwerkverkehr 👎 7,53 % Scan-Detektionen mit Intention „bad“
8080 99,43 % TCP 0,57 % UDP	0,86 %	1 ZMap 2 Hosts aus ShadowWhisperer Scanner-Liste [93] 3 Shadowserver [21]	Bzgl. TCP ausschliesslich SYN-Segmente UDP-Inhalte u.a. zum Protokoll SIP [260] und derzeit nicht decodierbaren Zeichenfolgen ● Bei IANA registrierter Dienst: HTTP Alternative [236] ● Eine einzelne Detektion mit IPv6 Shodan zu Scan-Ziel `st002` mittels 6to4, siehe Kapitel 4.2 ● Scan-Ziel `st005` erfährt das 1,25-fache an Kommunikationen zu diesem Port gegenüber den restlichen Scan-Zielen 👎 2,57 % Scan-Detektionen mit Intention „bad“

Analyse der Top 10 Ziel-Ports detektierter Scans Fortsetzung

Port	Anteil an Scan-Detektionen	Detektierte Scanner (Top 3)	Bemerkungen
443 95,22 % TCP 4,78 % UDP	0,70 %	1 ZMap 2 Hosts aus ShadowWhisperer Scanner-Liste [93] 3 Stretchoid [191]	Bzgl. TCP ausschliesslich SYN-Segmente Unter den UDP-Datagrammen sind Übertragungen zum DTLS- [261] und QUIC-Protokoll [262] ersichtlich ● Bei IANA registrierter Dienst: HTTPS [236] ● Zwei Detektionen mit IPv6 Shodan zu Scan-Ziel `st002` mittels 6to4, siehe Kapitel 4.2 ● Scan-Ziel `st004` erfährt mit 2799 am meisten Anfragen, `st010` mit 1725 am wenigsten 👎 2,61 % Scan-Detektionen mit Intention „bad“
53 9,96 % TCP 90,04 % UDP	0,59 %	1 ZMap 2 Hosts aus ShadowWhisperer Scanner-Liste [93] 3 Stretchoid [191]	Bzgl. TCP ausschliesslich SYN-Segmente UDP-Datagramme enthalten hauptsächlich DNS-Anfragen der Typen „*“, „A“ und „TXT“ zu `sl`, `version.bind`, `collectd.org`, `google.com` oder anderen ● Bei IANA registrierter Dienst: DNS [236] ● Zwei Detektionen mit IPv6 Shodan zu Scan-Ziel `st002` mittels 6to4 (siehe Kapitel 4.2) und eine IPv6-Detektion von RIPE Atlas Probe #1007148 [251] zu `st003` (DNS-AAAA-Abfrage zu `www.baidu.com`) ● Scan-Ziele `st004`, `st005`, `st006` und `st010` erfahren im Durchschnitt 70 % der Verbindungen im Vergleich zu den restlichen Scan-Zielen 👎 0,75 % Scan-Detektionen mit Intention „bad“

Analyse der Top 10 Ziel-Ports detektierter Scans Fortsetzung

Port	Anteil an Scan-Detektionen	Detektierte Scanner (Top 3)	Bemerkungen
0 99,98 % ICMP 0,02 % TCP	0,53 %	1 ZMap 2 ANT Lab [189] 3 Hosts aus ShadowWhisperer Scanner-Liste [93]	ICMP-Inhalte entsprechen derzeit nicht decodierbaren, ggf. zufälligen Zeichenfolgen TCP ausschliesslich SYN-Segmente ● ICMPv4-„Echo Request“-Pakete gemäss Beginn dieses Kapitels ● IPv6 ist hier ausgeschlossen (es wird hier lediglich ICMPv4 betrachtet) ● Scan-Ziel `st002` 🇳🇬 erfährt mit 2753 am meisten Anfragen, `st001` 🇨🇭 mit 1353 am wenigsten 👎 0,14 % Scan-Detektionen mit Intention „bad“
123 5,10 % TCP 94,90 % UDP	0,45 %	1 ZMap 2 Hosts aus ShadowWhisperer Scanner-Liste [93] 3 Shadowserver [21]	Bzgl. TCP ausschliesslich SYN-Segmente UDP-Datagramme enthalten Daten zum NTP-Protokoll [263] ● Bei IANA registrierter Dienst: NTP [236] ● Eine einzelne Detektion mit IPv6 Shodan zu Scan-Ziel `st002` 🇳🇬 mittels 6to4, siehe Kapitel 4.2 ● Scan-Ziel `st010` 🇬🇧 erfährt mit 2105 am meisten Verbindungen, `st008` 🇷🇺 mit 1248 am wenigsten 👎 2,13 % Scan-Detektionen mit Intention „bad“

4.5. Meist kontaktierte Ports zwischen einzelnen Scan-Zielen

Signifikante Unterschiede bei kontaktierten Ports, die einzelne Scan-Ziele im Vergleich zur Gesamtübersicht aufweisen, sind diesem Kapitel zu entnehmen.

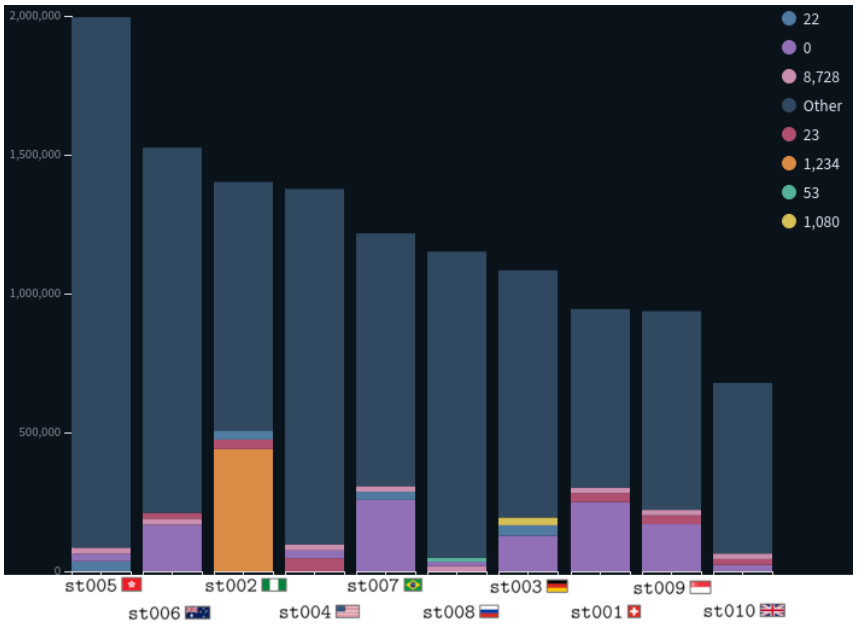


Abbildung 4.8.: Top 3 allgemeine Ziel-Ports pro Scan-Ziel, Visualisierung „Destination Top 10 IP and Top 3 Ports“ ohne Filter `rule.category: ScannerDetection`

Die Visualisierung „Destination Top 10 IP and Top 3 Ports“ (siehe Tabelle D.1 in Kapitel D) in Abbildung 4.8 zeigt die 3 meist vorkommenden Ziel-Ports pro IPv4-Adresse eines Scan-Ziels. Hierbei werden nicht nur Scan-Detektionen sondern allgemeine Verbindungen betrachtet.

Zu folgenden Ziel-Ports sind signifikante Unterschiede bezüglich deren Vorkommen ersichtlich[45]:

- Port 1234 bei Scan-Ziel `st002`
- Port 1080 bei Scan-Ziel `st003`

Port 1234

Bei IANA registrierter Dienst: „Infoseek Search Agent“ [236]

Dieser Port tritt bei den Verbindungen zu Scan-Ziel `st002` und `st006` auf[46].
`st002` erfährt hierbei über die 300-fache Menge an Verbindungen[46].

Ermittlungen in Arkime zeigen, dass die meisten Anfragen aus AS AS16509 „AMAZON-02“ [234] mit 244’057 Sitzungen stammen. Zum zweit häufigsten Quell-AS werden 192 Sitzungen von Arkime aufgezeichnet. Mit dem TCP-Protokoll sind lediglich SYN-Segmente ersichtlich. Zu UDP werden 28 Sitzungen aufgezeichnet, dessen Datagramme hauptsächlich nicht decodierbare, ggf. zufällige Zeichenfolgen beinhalten, aber auch Inhalte zum Protokoll SIP [260].

Das Dashboard „Scanner Detection“ zeigt mit zusätzlichem Filter `destination.port: 1234` folgendes zu dessen Scan-Detektionen: 15,77 % der Detektionen sind mit Intention „bad“ vermerkt, wobei signifikant mehr Tor Exit Nodes unter den Scan-Quellen genannt werden.

Port 1080

Bei IANA registrierter Dienst: „Socks“ [236]

Scan-Ziel `st003` wird unter diesem Port über 12 mal häufiger kontaktiert als andere Scan-Ziele[47].

Hierbei stammen gemäss Arkime die meisten Verbindungen von AS AS16276 „OVH SAS“ [264] (über 6000). Bei den TCP-Segmenten handelt es sich auch hier ausschliesslich aus SYN-Segmenten. Aufgezeichnete UDP-Datagramme beinhalten wieder SIP-relevante Inhalte und ggf. zufällige Zeichenfolgen.

Mit zusätzlicher Anwendung des Filters `destination.port: 1080` weist das „Scanner Detection“-Dashboard 2,51 % der detektierten Scans mit Intention „bad“ aus.

[45] Der Fokus liegt hierbei auf Ports, die in den häufigsten Ports detektierter Scans aus Kapitel 4.4 nicht vorkommen
[46] Ermittelt in Arkime mit Filter `port.dst == 1234` und Funktion „Export Unique Dst IP with counts“
[47] Ermittelt in Arkime mit Filter `port.dst == 1080` und Funktion „Export Unique Dst IP with counts“

4.6. JA4+

Mittels Malcolm ermittelte JA4+-Werte erfahren gemäss Kapitel 3.2 eine manuelle Auswertung.

In OpenSearch Dashboards wird die „Discover“-Übersicht geöffnet und die Filter entsprechend Kapitel 4.3 angewendet (ohne Einschränkung auf detektierte Scans)[48]. Ermittelte JA4+-Werte sind je nach Methode in entsprechenden Feldern vermerkt und können entsprechend eingesehen werden[49]. Zu folgenden JA4+-Methoden sind Werte vorzufinden [106]:

- `http.ja4h`
  JA4H, HTTP Client Finterprinting
  Anzahl Einträge: 2463 (12 einzigartig)
- `ssh.ja4ssh`
  JA4SSH, SSH Traffic Fingerprinting
  Anzahl Einträge: 6 (2 einzigartig)
- `tcp.ja4l`
  JA4L, Client to Server Latency Measurment
  Anzahl Einträge: 357’309 (357’309 einzigartig)
- `tcp.ja4t`
  JA4T, TCP Client Fingerprinting
  Anzahl Einträge: 6’474’774 (96’351 einzigartig)
- `tcp.ja4ts`
  JA4TS, TCP Server Response Fingerprinting
  Anzahl Einträge: 42 (9 einzigartig)
- `tls.ja4`
  JA4, TLS Client Fingerprinting
  Anzahl Einträge: 4087 (25 einzigartig)

Eine Übersicht der JA4+-Implementationen ist Tabelle 2.1 in Kapitel 2.2 zu entnehmen. Nachfolgend wird jede JA4+-Fingerprinting-Methode einzeln betrachtet. Dazu wird in der „Discover“-Übersicht jeweils der Filter `FELDNAME: exists` hinzugefügt (Anstelle von `FELDNAME` wird der zuvor vermerkte JA4+-Feldname verwendet). Durch Analysieren oder Hinzufügen einzelner Felder als Spalten werden weitere Zusammenhänge ersichtlich.

Bei Bedarf einer tieferen Auseinandersetzung mit einzelnen Verbindungen wird nach diesen in Arkime gesucht und allfällig verfügbare PCAP-Dateien extrahiert. Diese enthalten den entsprechenden Netzwerkverkehr inklusive Paketinhalte und können in Anwendungen wie Wireshark betrachtet werden.

Für eine semi-automatisierte Durchsuchung der JA4+-Datenbank von FoxIO wird in der „Discover“-Übersicht jeweils zusammen mit dem Filter `FELDNAME: exists` das entsprechende Feld als Spalte hinzugefügt. Durch Klick oben auf „Save“ wird die aktuelle Suche abgespeichert und mittels „Reporting“ ein CSV-Report exportiert.

Ein Beispiel mit `http.ja4h` ist in Abbildung 4.9 ersichtlich.

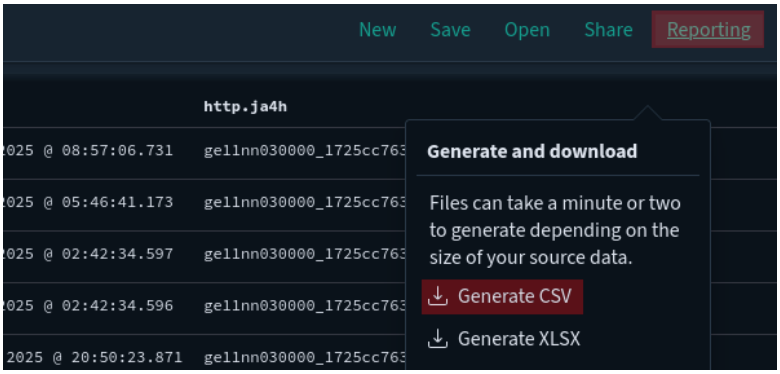


Abbildung 4.9.: Exportieren einer gespeicherten Suche in der „Discover“-Übersicht in OpenSearch Dashboards

[48] Der Filter `rule.category: ScannerDetection` wird in diesem Kapitel nicht angewendet

[49] Unter „Dashboards Management“ → „Index patterns“ → „arkime_sessions3-*“ sind sämtliche genutzten Felder ersichtlich und können dort mittels `ja4` gefiltert werden. Daraufhin wird nach jedem Feld in der gefilterten „Discover“-Übersicht gesucht, ob es über Werte verfügt (zum Beispiel mittels Filter `tcp.ja4t: exists`). Einige Felder wie `tcp.ja4t` und `zeek.conn.ja4t` verfügen über dieselben Daten, sind jedoch aufgrund deren Bezeichnung identifizierbar

Auf diesen Weg gestartete Report-Prozesse haben ein Limit von 10'000 Zeilen [265]. Dieses Limit wird jeweils von der Anzahl Einträge mit `tcp.ja4l` und `tcp.ja4t` überschritten. In diesem Fall verläuft der Export unter dem Menüpunkt „Reporting" mit einem erhöhten Limit nicht erfolgreich [265]. Stattdessen wird eine temporäre „Data Table"-Visualisierung erstellt, in welcher die Zeilen anhand beispielsweise `tcp.ja4t` mit einer Grösse von 40'000 Elementen aufgeteilt werden. Daraufhin wird per Klick auf den Download-Button links oben im „Raw"-Format eine CSV-Datei exportiert. Teile des gegebenen Zeitrahmens werden einzeln bearbeitet, sodass jeweils weniger als 40 Tausend Elemente aufgelistet sind.

Nun wird mittels folgendem Befehl die JA4+-Datenbank von FoxIO in Form einer JSON-Datei [110] beispielsweise nach JA4H-Werten durchsucht:

```
while read value; do
jq ".[] | select(.ja4h_fingerprint==\"${value}\")" ja4+_db.json >>
   ja4h_matches.json
done < <(awk -F'"' 'NR > 1 {print $4}' ja4h.csv | sort | uniq)
```

Quelltext 4.5: Beispiel: Durchsuchung der JA4+-Datenbank von FoxIO [110] nach JA4H-Fingerprints [266, 267]

Für den Daten-Export zu den Feldern `tcp.ja4l` und `tcp.ja4t` erfährt die CSV-Datei ein anderer Aufbau. Daher wird die letzte Zeile aus Quelltext 4.5 zu folgender angepasst:

```
while read value; do
jq ".[] | select(.ja4t_fingerprint==\"${value}\")" ja4+_db.json >>
   ja4t_matches.json
done < <(awk -F',' 'NR > 1 {print $1}' ja4t-datatable-*.csv | sed 's/"//g' |
   sort | uniq)
```

Quelltext 4.6: Anpassung des Quelltexts 4.5 zu dem Format der CSV-Exports von `tcp.ja4t` (Ausführung für `tcp.ja4t` dauert über 24 Stunden)

Diese Datenbank ist öffentlich einsehbar und erlaubt Beiträge von autorisierten Benutzerinnen und Benutzer [268]. Mit gegebener Zeit und Mittel handelt es sich bei der Datenbank von FoxIO um die einzige ermittelte Quelle, die anhand JA4+-Fingerprints durchsucht werden kann [106]. Censys bietet dies nur für JA4S- und JA4X-Fingerprints an, zu welchen hier keine Werte ermittelt werden [269].

```
jq '.[]' ja4+_db.json \
| grep -Eo "ja4[^_]*_fingerprint" \
| sort | uniq
ja4_fingerprint
ja4h_fingerprint
ja4s_fingerprint
ja4t_fingerprint
ja4tscan_fingerprint
ja4ts_fingerprint
ja4x_fingerprint
```

Quelltext 4.7: JA4+-Fingerprint-Typen in der JA4+-Datenbank von FoxIO in Form einer JSON-Datei [110]

Quelltext 4.7 zeigt die in der Datenbank abgespeicherten JA4+-Fingerprint-Typen.

Einzig zu Fingerprints dieses Typs JA4T ergibt die Suche in der FoxIO-JA4+-Datenbank [110] Treffer.

Nachfolgend wird auf die ermittelten Ergebnisse zu einzelnen JA4+-Typen eingegangen.

JA4L

JA4L basiert auf Zeitstempel, woraus alleine aus einem entsprechenden Wert keine Rückschlüsse auf die Intention möglich sind [270]. Des Weiteren befinden sich keine JA4L-Fingerprints in der JA4+-Datenbank von FoxIO, weshalb diese nicht weiter betrachtet werden.

JA4H

Anhand von JA4H wird ein Fingerprint des HTTP-Clients angelegt. Ermittlungen in der „Discover“-Übersicht in OpenSearch Dashboards zusammen mit dem Filter `http.ja4h: exists` führen zu folgenden Aussagen:

- Die Anteile angewendeter Methoden in den HTTP-Anfragen lauten wie folgt: 93,65 % „GET“, 4,43 % „POST“, 1,73 % „CONNECT“ und 0,19 % „HEAD“
- Aufgezeichnete HTTP-GET-Anfragen enthalten als Wert für „Host“ [271] diverse Domänen, die u. a. zu pornografischen Webseiten oder Glücksspielen führen
  - Der meist hinterlegte Wert ist `freedomhouse.org`, der zu einer Menschenrechtsorganisation gehört [272]
  - Andere Werte unter „Host“ zeigen direkt auf diverse IP-Adressen
  - Spezifische Pfade zu u. a. Javascript-, PHP- oder Streaming-Dateien (Endungen „.m3u8“, „.ts“ [273]) sind ebenfalls in den Anfragen ersichtlich
- HTTP-POST-Anfragen enthalten unter anderem
  - Login-Versuche zu `/wp-login.php`
  - Formularfelder mit Wert `androxgh0st`, die einer Python-Malware zugeordnet werden können [274]
  - User-Agents wie „Python WinRM client“ oder „Hello, World“

JA4

Unter dem Netzwerkverkehr mit ermitteltem JA4-Fingerprint sind ausschliesslich UDP-Datagramme mit DTLS und QUIC-Protokollanwendungen vorzufinden.

JA4TS

Hierunter befinden sich TCP-Verbindungen, die keine TCP-SYN-Segmente beinhalten. Das einzige Netzwerkpaket mit JA4TS-Fingerprint und TCP-Payload in Arkime wird mit IPv6 von `2a04:4e42:400::644` mit Quell-Port 80 zu `st010` 🇬🇧 gesendet. Hierbei wird unter anderem eine PGP-Signatur übertragen.

Der Whois-Eintrag zur IPv6-Adresse verweist auf die Organisation Fastly. Diese ist in Kapitel 4.1.2 im Zusammenhang mit Debian-Betriebssystem-Aktualisierungen bereits erwähnt. Die IPv6-Adresse gehört dem dort erwähnten AS von Fastly an [235].

JA4T

Gemäss Analyse anhand Quelltext 4.6 mit der FoxIO-JA4+-Datenbank gibt es Übereinstimmungen zu folgenden JA4T-Fingerprints [110]:

- `1024_00_00_00`
  Applikation **masscan**
  Anzahl Einträge: 51'032
- `1024_2_1460_00`
  Applikation **Nmap**
  Anzahl Einträge: 182
- `1025_2_1460_00`, `512_00_00_00`
  Applikation „Modified **Nmap**“
  Anzahl Einträge: 1275
- `29200_2-4-8-1-3_1460_6`
  Gerät „Nest Thermostat V2“
  Anzahl Einträge: 2598
- `64240_2-1-3-1-1-4_1460_8`
  Betriebssystem „Windows 10“
  Anzahl Einträge: 8711
- `64240_2-4-8-1-3_1460_7`
  Betriebssystem „WSL Ubuntu 22.04“
  Anzahl Einträge: 127'484
- `65535_00_00_00`
  Applikation „**ZMap**“
  Anzahl Einträge: 105'144
- `65535_2-1-3-1-1-8-4-0-0_1460_6`
  Betriebssystem „Mac OSX/iPhone“
  Anzahl Einträge: 6
- `65535_2_1460_00`
  Applikation „Modified **ZMap**“
  Anzahl Einträge: 658
- `65535_2-4-8-1-3_1460_8`
  Betriebssystem „Ubuntu 22.04“
  Anzahl Einträge: 120

Die Anzahl Einträge wird in der „Discover“-Übersicht unter Anwendung der Filter und des Zeitraums aus Kapitel 4.3 ermittelt (exklusive `rule.category: ScannerDetection`). Zusätzlich wird jeweils eine Suche nach dem JA4T-Fingerprint entsprechend `tcp.ja4t: FINGERPRINT` durchgeführt.

Die Betrachtung des Felds `event.provider` zeigt, dass sämtliche JA4T-Werte von Arkime stammen. Aus diesem Grund wird in der nachfolgenden Auswertung die Gesamtanzahl entsprechender Einträge zur Anzahl der bekannten Scanner (Intention „good“) hinzugefügt[50]. Es werden lediglich die JA4T-Einträge für masscan, Nmap und ZMap inklusive „Modified“-Varianten gezählt. Ein JA4T-Fingerprint beruht auf den Merkmalen eines TCP-Segments ohne Berücksichtigung des IP-Headers [275]. Bisherige ZMap-Detektionen entstehen anhand des IPv4-Identifikations-Felds, das nicht in jeder ZMap-Variante verwendet wird (siehe Kapitel 2.1.4) [27]. Daher werden ZMap-Detektionen mit JA4T zusätzlich gezählt.

Hierbei handelt es sich total um 158'291 identifizierte JA4T-Fingerprints detektierter Scan-Anwendungen.

[50] Die Zeek-Notices, in welchen die Intention jeweils vermerkt wird, stammen von Zeek. Zu den Top 5 Quell-IP-Adressen pro aufgeführtem JA4T-Fingerprint existieren zusammen weniger als zehn Zeek-Notices, jedoch über hunderttausend Verbindungs-Einträge. Somit werden zu diesen Verbindungen keine Zeek-Notices ausgelöst und bilden manuell ermittelte Scan-Detektionen

JA4SSH

Zwei JA4SSH-Fingerprints von derselben IP-Adresse (`185.60.136.12`) werden aufgelistet: `c12s0_c1s0_c1s0` und `c12s0_c4s0_c4s0`. Diese sind bei jeweils drei Verbindungen zu den Scan-Zielen `st004` , `st005` und `st008` hinterlegt.

Eine Analyse der SSH-Kommunikationen anhand entsprechender PCAP-Extrakte zeigt folgendes Verhalten[51]:

- Sekunde 0:
  Senden eines TCP-SYN-Segments [10] an das Scan-Ziel
  TCP-Segmente werden von der ufw-Firewall auf dem Scan-Ziel verworfen
- Sekunde 1:
  Wiederholung des TCP-SYN-Segments
- Sekunde 2:
  Wiederholung des TCP-SYN-Segments
- Sekunde 3:
  Senden mehrerer Pakete nacheinander
  - TCP-ACK-Segment [10]
  - SSH-Paket mit Identifikationsstring zu „Protocol Version Exchange“ entsprechend des SSH-Transport-Layer-Protokolls [276]
    - Dieser trägt hier den Wert `SSH-2.0-Go`
    - Das Paket wird vier mal hintereinander gesendet
  - TCP-Segment mit Flags RST und ACK [10]

Der Wert `SSH-2.0-Go` entstammt einer Programmbibliothek zur Programmiersprache Go [277]. Bei einem SSH-Verbindungsaufbau zu einer Adresse ohne aktivem Serverdienst werden lediglich wiederholt TCP-SYN-Segmente bis zur Zeitüberschreitung gesendet. Das beobachtete, wiederholte Verhalten entspricht demnach nicht dem zu Erwartenden.

Recherchen zur Sender-IP-Adresse führen einzig zum zugehörigen RIPE-Datenbankeintrag [278]. Dieser verweist auf eine Organisation im Iran, die Informatikdienstleistungen inklusive Server-Hosting anbietet [279].

[51] Die beobachteten Zeitabstände sind mit einer Toleranz von Millisekunden jeweils auf die Sekunde genau

4.7. Intention der Scan-Quellen

Die Detektion Out-of-Band Probe Attribution gemäss RFC 9511 vermerkt eine zugehörige Quelle mit Intention „good“ (👍), wird jedoch aufgrund der Ergebnisse in Kapitel 4.1.4 ausgeklammert.

Auf Mehrfachdetektionen, bei denen ein Scan beide Intentionen erfüllt, wird in Kapitel 4.1.6 eingegangen.

Verbindungen aus Tor Exit Nodes werden hier mit der Intention „bad“ (👎) versehen (siehe Kapitel 3.1.3), jedoch sind derartige Verbindungen nicht immer bösartig. Beispielsweise kann eine Benutzerin oder ein Benutzer Tor benutzen wenn keine ungefilterte Verbindung zum Internet besteht [280]. Ein Scan-Ziel kann theoretisch von allen mit Internetzugang direkt mittels IP-Adresse oder über einen veralteten DNS- oder Suchmaschinen-Eintrag kontaktiert werden. Wurde unter dessen IP-Adresse früher ein legitimer Dienst angeboten, ist ein bösartiger Scan unwahrscheinlicher. Diese Aussage gilt jedoch nur für Ports, unter welchen entsprechende Dienste angeboten werden (üblicherweise sind das bei einem Webserver TCP-/UDP-Ports 80 sowie 443 für HTTP(S) und QUIC [236]).

Detektionen von Tor Exit Nodes sind mittels dem Filter `zeek.notice.msg: *ExitNode*` einsehbar. Hierzu werden 842 Detektionen von 34 IPv4-Adressen aufgeführt.

Abbildung 4.10 zeigt die Menge an detektierten Scans, aufgeteilt nach Intention und Detektion-Typ. Zusätzlich wird die Anzahl an JA4T-Fingerprints identifizierter Scan-Anwendungen aus Kapitel 4.6 miteinbezogen. Ausser diesen Fingerprints und der ZMap-Detektion basieren sämtliche ermittelten Intentionen auf den angereicherten Tabellen aus Kapitel 3.1.3.

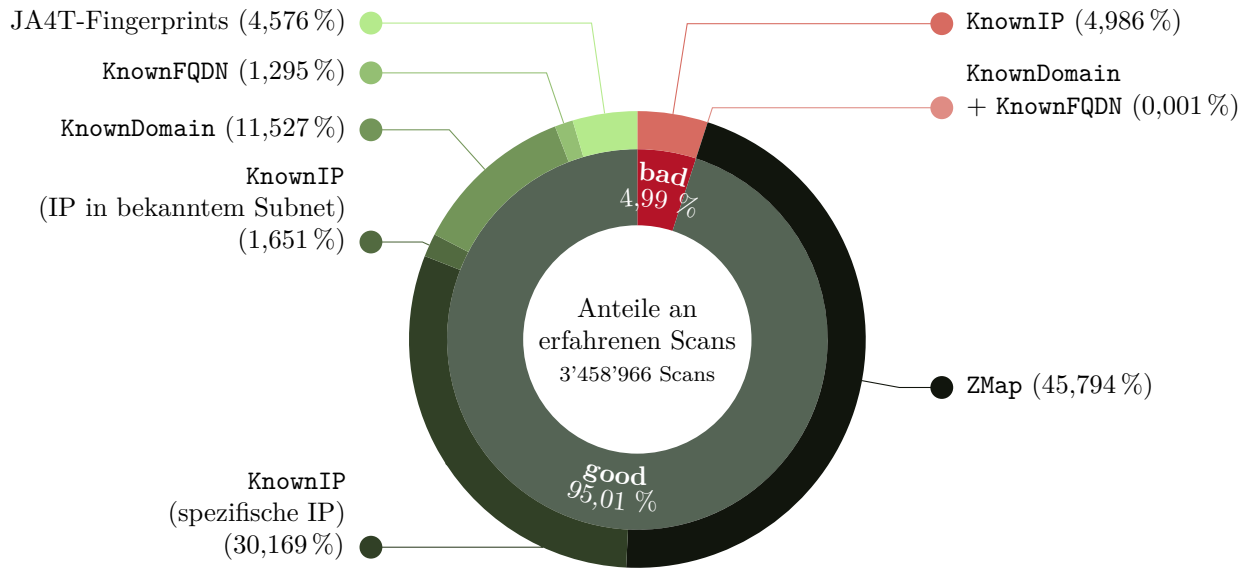


Abbildung 4.10.: Anteile an detektierten Intentionen, aufgeteilt nach Detektion-Typ des Zeek-Skripts aus Kapitel 3.1.3 plus JA4T-Fingerprints aus Kapitel 4.6
(Aufteilung der Zählung zu `KnownIP` mittels Filter `zeek.notice.msg: *subnet*`)

Diese sowie die nachfolgenden Abbildungen entsprechen bis auf die JA4T-Fingerprints den Visualisierungen in OpenSearch Dashboards (siehe Tabelle D.1). Sie werden zur Übersicht sowie manuellen Erweiterung neu erstellt.

Scan-Quellen mit einer positiv vermerkten Intention werden in der nächsten Abbildung aufgeführt. Aus Übersichtsgründen ist der Anteil an ZMap-Detektionen ausgeklammert. Die Institutionen mit der höchsten Anzahl an Scans sind mit entsprechenden Anteilen namentlich erwähnt.

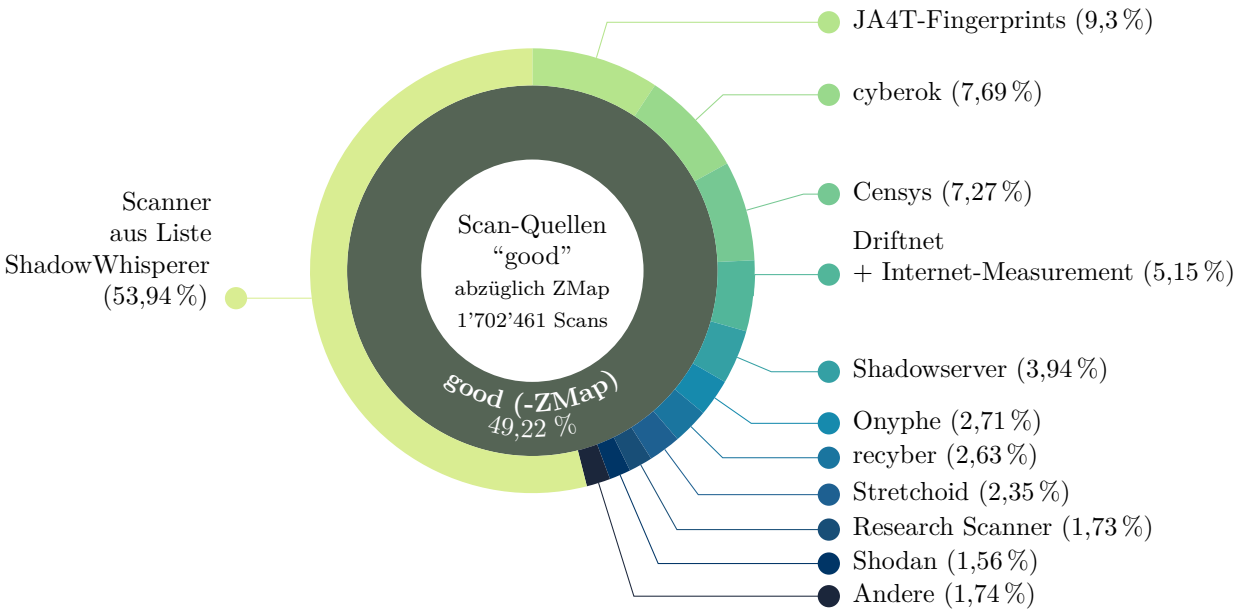


Abbildung 4.11.: Anteile detektierter Scan-Quellen mit Intention „good“ (Abzüglich ZMap-Detektionen) [20, 21, 93, 95, 178, 183, 185, 190, 191, 249]

Abbildung 4.12 zeigt erfahrene Scans mit negativ notierter Intention, aufgeteilt nach Bezeichnungen der Scan-Quellen. Weitere Informationen zu einer einzelnen „Abuse-IoC-ID“ sind anhand folgendem URI-Muster einsehbar: `https://threatfox.abuse.ch/ioc/ID` [281]. Für die ID `1281855` entspricht dies der URI `https://threatfox.abuse.ch/ioc/1281855` [281]. Merkmale einzelner IoCs sind in der Abbildung angefügt.

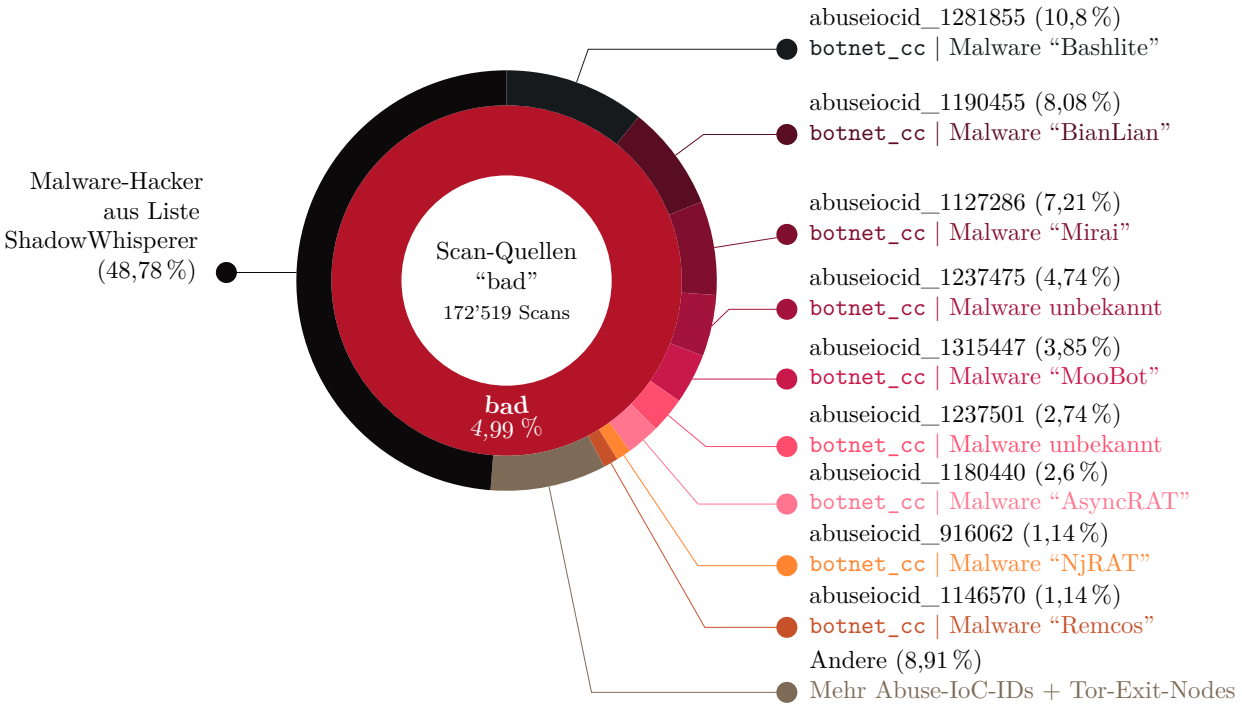


Abbildung 4.12.: Anteile detektierter Scan-Quellen mit Intention „bad“ und allfälligen IoC-Informationen [93, 258, 282–289]

Die zur Detektion verwendeten Scanner- und Malware-Listen von ShadowWhisperer [93] vereinen IP-Adressen bekannter Scan-Quellen, führen jedoch nur die Adressen ohne Namen. Stattdessen wird in der Tabelle zu Einträgen aus diesen Listen die Bezeichnung `shadowwhisperer_scanner`, `shadowwhisperer_malware_hackers` oder `shadowwhisperer_malware_hosting` vermerkt. Quelltext 4.8 zeigt die am meisten vorkommenden Bezeichnungen zu Einträgen in den Tabellen bekannter Adressen aus Kapitel 3.1.3.

```
for file in \
zeek/knownscanners*.table; do
echo "####### $file"
awk 'NR > 1 {print $1}' $file | sort \
| uniq -c | sort -n -r | head -n 10
done
####### zeek/knownscannersfqdn.table
1 stretchoid
1 shodan
1 shadowserver
1 research-scanner
1 recyber
1 onyphe
1 netsecscan
1 ncsc
1 leakix
1 internet-measurement
```

```
####### zeek/knownscannersip.table
52186 ripeatlas_probe
16445 shadowwhisperer_malware_hackers
7647 shadowwhisperer_scanner
1536 censys
1280 netsystems
1052 alphastrike
764 u_mich
665 shadowwhisperer_malware_hosting
376 rapid7
352 stretchoid
####### zeek/knownscannerssubnet.table
21 internet-measurement
13 censys
3 driftnet
2 recyber
2 mpgde
1 u_stanford
1 criminalip
```

Quelltext 4.8: Top 10 Bezeichnungen der Einträge in den Tabellen aus Kapitel 3.1.3

Tabelle 4.3 zeigt die mittels Zeek-Skript ermittelten Anteile detektierter Intentionen pro Scan-Ziel.

Tabelle 4.3.: Anteile an detektierten Intentionen pro Scan-Ziel ohne JA4+-Fingerprints

Scan-Ziel	Anzahl Scan-Detektionen ↑ Höchster Wert ↓ Niedrigster Wert	Anteil an Scan-Intentionen	
		Intention „good“ 👍 ↑ Höchster Wert ↓ Niedrigster Wert	Intention „bad“ 👎 ↑ Höchster Wert ↓ Niedrigster Wert
st001 🇨🇭	221’603 ↓	96,10 %	3,90 %
st002 🇮🇪	261’527	94,15 %	5,85 %
st003 🇩🇪	247’737	89,43 % ↓	10,57 % ↑
st004 🇺🇸	511’256	95,39 %	4,61 %
st005 🇭🇰	522’655 ↑	93,49 %	6,51 %
st006 🇦🇺	510’013	97,46 % ↑	2,54 % ↓
st007 🇧🇷	285’838	90,57 %	9,43 %
st008 🇷🇺	258’771	97,35 %	2,65 %
st009 🇸🇬	251’160	96,72 %	3,28 %
st010 🇬🇧	230’115	95,75 %	4,25 %
Durchschnitt	**330’067,5**	**94,64 %**	**5,36 %**

4.8. Standortangaben

Gemäss den Ausführungen in Kapitel 3.5.2 werden die Standortangaben anhand Arkime anstelle der Daten aus Zeek betrachtet. Dies hat zur Folge, dass nicht nur die mittels Zeek detektierten Scans, sondern sämtliche Verbindungen betrachtet werden[52]. Aufgrund des Aufbaus und der Nutzung der Scan-Ziele wird jede Verbindung als Scan betrachtet, auch wenn sie inklusive deren Intention mit Arkime nicht direkt auslesbar ist.

Folgender Filter wird in diesem Abschnitt angewendet:

- Zeitraum entsprechend Kapitel 4.3 ohne Ausklammerungen
- Quell- und Ziel-IP-Adressen entsprechend Kapitel 4.3
- Daten ausschliesslich aus Arkime mittels `event.provider == arkime`
- Standortangabe der Quelle muss existieren (`country.src == EXISTS!`)
  96 % der aufgezeichneten Arkime-Einträge vor Anwendung der anderen Filter erfüllen dieses Kriterium

Die Weboberfläche von Arkime bietet unter der Funktion „Fetch Viz Data“ einen Zeitstrahl mit der Menge an Sitzungen sowie eine Weltkarte an (siehe Abbildung 3.7 in Kapitel 3.1.4). Diese Karte erlaubt es, die Standortangaben von Quell- und Ziel-IP-Adressen einzeln oder kombiniert zu betrachten. In der vergrösserten Version werden zusätzlich die zehn meist vertretenen Länder aufgelistet[53].

Wie in Kapitel 3.5.2 gezeigt, existiert nicht für jede IP-Adresse eine Standortangabe. Entsprechende Einträge werden ausgefiltert und nicht auf der Weltkarte repräsentiert. Abbildung 4.13 zeigt die von Arkime ermittelten Ursprungsländer erfahrener Verbindungen auf der gesamten Analyse-Umgebung. Je dunkler ein Land markiert ist, desto mehr Kommunikationen werden von dort aus gesendet. Die Länderbezeichnungen sind in Form von Codes mit zwei Buchstaben („alpha-2“) des ISO-Standards 3166 [224, 225] aufgelistet.

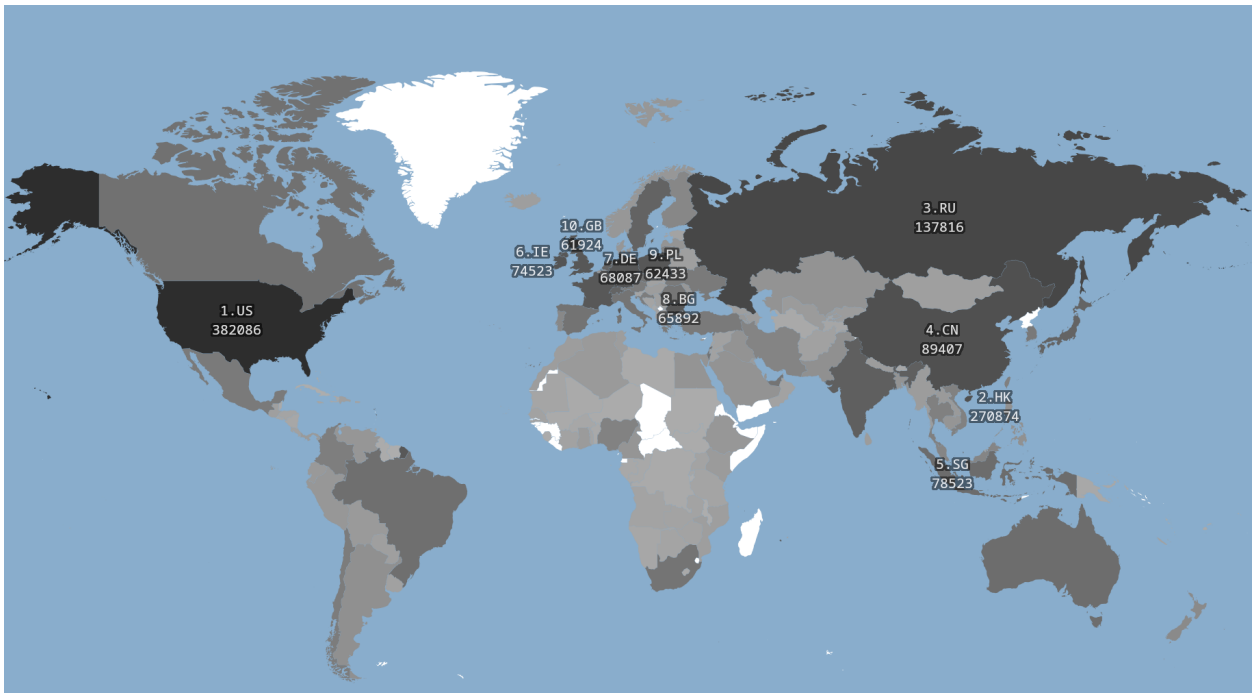


Abbildung 4.13.: Ursprungsländer erfahrener Scans auf der gesamten Analyse-Umgebung (Top 10 Werte eingetragen)

[52] Die Korrelation der Scan-Detektionen in Zeek mit den Einträgen in Arkime kann in der gegebenen Zeit nicht durchgeführt werden

[53] Ähnliche Visualisierungen mit denselben Daten und Filter inklusive `event.provider` sind auch in OpenSearch Dashboards möglich. Arkime bietet hierzu eine bereits vorhandene Übersicht

Nachfolgend werden die Top 5 Ursprungsländer der Kommunikationen zu den einzelnen Scan-Zielen gezeigt. Diese werden in Arkime ermittelt unter Anwendung von „Views“, die zusätzlich entsprechende Adress-Filter hinterlegt haben (zu jedem Scan-Ziel gibt es eine View). Nach Öffnen des Kontextmenüs der Spalte „Src IP / Country“ wird mit der Option „Export Unique Src Country with count“ eine entsprechende Rangliste ausgegeben. Abbildung 4.14 zeigt die Top 5 Ursprungsländer pro Scan-Ziel. Die abgebildeten Mengen beziehen sich rein auf die Anteile eines Ursprungslands innerhalb der Top 5. Die Menge an Verbindungen zwischen den Scan-Zielen wird hier nicht verglichen, ist jedoch in Tabelle 4.3 aus Kapitel 4.7 einsehbar.

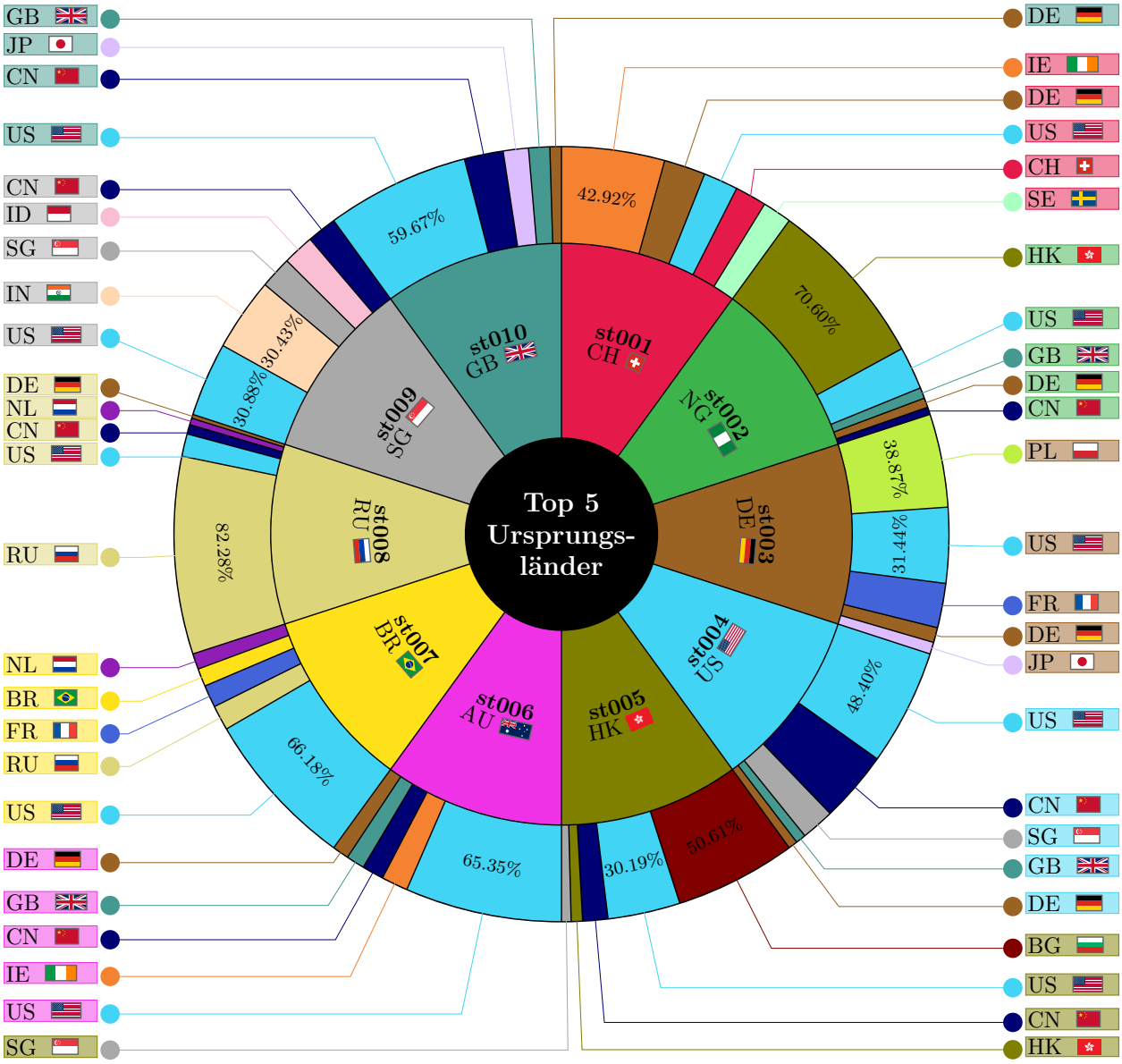


Abbildung 4.14.: Top 5 Ursprungsländer erfahrener Scans pro Scan-Ziel (Prozentual pro Scan-Ziel, Menge an Scans pro Scan-Ziel ist nicht abgebildet)

4.9. Öffentliche Informationen zu Scan-Zielen

Dieser Abschnitt zeigt einen Einblick in die öffentlich einsehbaren Informationen kontaktierter Scan-Ziele. Es werden nur IP-Adressen auf den Plattformen überprüft, zu denen ein Scan-Eintrag im Dashboard ersichtlich ist. Andernfalls kann nicht ausgeschlossen werden, dass ein zukünftiger Scan durch eine entsprechende Suchanfrage angestossen wird. Ist zu der Scan-Quelle eindeutig eine Suchmaschine zuweisbar, wie im Falle von Censys oder Shodan, erfährt diese zur Ermittlung einsehbarer Daten entsprechende Anfragen.

Mittels Suche nach `zeek.notice.msg: *shodan*` im Dashboard erscheinen die von **Shodan** gescannten Scan-Ziele unter den Ziel-IP-Adressen. Hierbei ist das Scan-Ziel in Russland (`st008` 🇷🇺) als einziges nicht aufgeführt. Es werden lediglich die IPv4-Adressen der Scan-Ziele bei Shodan angefragt. Es sind keine Informationen zu den Scan-Zielen auf Shodan einsehbar.

Die Betrachtung von **Censys** [20] (Suche nach `zeek.notice.msg: *censys*` im Dashboard) zeigt bei `st008` 🇷🇺 zwei Scan-Detektionen, wobei die restlichen Scan-Ziele hierzu zwischen zwölf- und fünfzehntausend Einträge haben. Censys bietet im Vergleich zu Shodan Ergebnisse zu jeder gescannten Scan-Ziel-IPv4-Adresse (siehe Abbildungen 4.15 und 4.16). Hierbei werden keine öffentlich erreichbaren Dienste angezeigt, jedoch zutreffende Routing- und Standort-Informationen [20].

ONYPHE untersucht ebenfalls die IPv4-Adressen sämtlicher Scan-Ziele (Suche nach `zeek.notice.msg: *onyphe*` im Dashboard). Die Suchmaschine von ONYPHE [290] erlaubt es zum Beispiel dessen Daten mit der Anfrage `category:datascan ip:8.217.233.46` nach Informationen zu spezifischen IP-Adressen zu durchsuchen [188]. Zu keiner IPv4-Adresse der Scan-Ziele erscheint hierbei ein Resultat [290]. Vor-Analysen während der Aufzeichnung im Januar 2025 zeigten bei ONYPHE Resultate zu den Scan-Zielen `st007` 🇧🇷 und `st009` 🇸🇬 [291, 292]. Diese sind zum aktuellen Zeitpunkt einen Monat später nicht mehr auf der Plattform einsehbar [290].

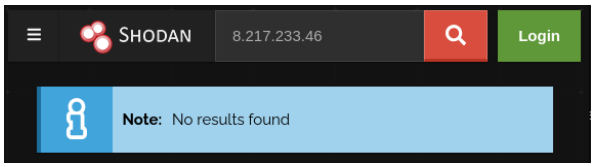


Abbildung 4.15.: Shodan: Suchresultat zu "8.217.233.46" [293]

Abbildung 4.16.: Censys: Suchresultat zu "8.217.233.46" [294]

Die Scanner von `internet-measurement.com` [178] prüfen wie ONYPHE die IPv4-Adressen sämtlicher Scan-Ziele (Suche nach `zeek.notice.msg: *internet-measurement*` im Dashboard). Die zugehörige Suchmaschine Driftnet [249] ermöglicht Anfragen zu spezifischen IP-Adressen mittels beispielsweise `ip:8.217.233.46`. Die Webseite `internet-measurement.com` [178] verweist zwar auf Driftnet, jedoch werden auch Scans von Driftnet selber detektiert (Suche nach `zeek.notice.msg: *driftnet*` im Dashboard, scannt ebenfalls die IPv4-Adressen aller Scan-Ziele).

Driftnet zeigt Ergebnisse zu den IPv4-Adressen der Scan-Ziele `st001` , `st003` , `st004` und `st005` . Bei letzterem handelt es sich um einen spezifischen DNS-Eintrag, der auf die IPv4-Adresse von `st005` verweist (`fuzesen.mapeilin.com`) [249]. Bei den restlichen Ergebnissen handelt es sich um DNS-Einträge von Hosting- oder Internet-Dienstleistungsunternehmen spezifisch zur entsprechenden IP-Adresse [249]. Der Whois-Eintrag zur Domäne `mapeilin.com` verweist auf eine Registrierung durch Cloudflare als Registrar [295]. Unter den Kontaktangaben wird China als Land mit der zugehörigen Provinz aufgeführt, weitere Informationen sind jedoch nicht einsehbar [295, 296].

LeakIX [297] (Suche nach `zeek.notice.msg: *leakix*` im Dashboard) prüft ebenfalls sämtliche IPv4-Adressen der Scan-Ziele. Dessen Suchmaschine [297] wird zu diesen Adressen mit einer Abfrage entsprechend `ip:"8.217.233.46"` in den Kategorien „Leaks“ und „Services“ geprüft [297]. Unter „Leaks“ gibt es Treffer für die IPv4-Adressen von `st002` , `st005` , `st007` und `st008` [298–301].

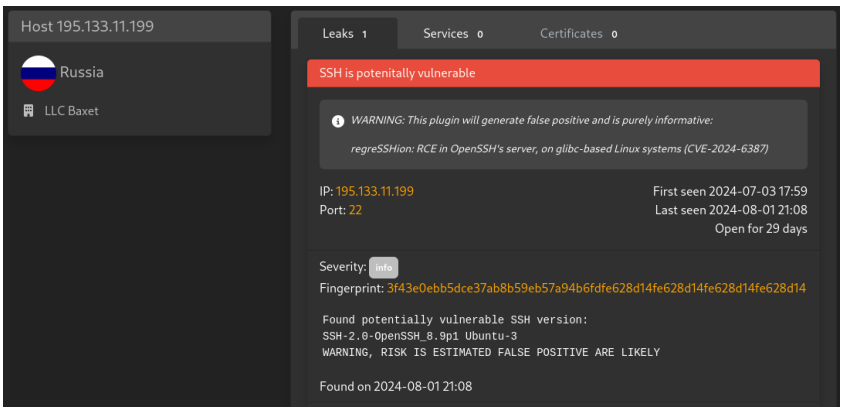


Abbildung 4.17.: LeakIX: Suchresultat zu " ip:"195.133.11.199" [301]

Die Standortangaben treffen zu, bis auf die von `st007` : Hier wird die Niederlande angegeben[54]. Die Ergebnisse weisen alle veraltete SSH-Serverdienst-Versionen aus (siehe Abbildung 4.17). Hierbei aufgeführte Zeitstempel („Last seen“) befinden sich vor dem Aufbau entsprechender Scan-Ziele.

Google scannt mit dessen Googlebot [193] (Suche nach `zeek.notice.msg: *googlebot*` im Dashboard) die IPv4-Adressen der Scan-Ziele `st002` , `st003` , `st008` und `st010` . Anfragen entsprechend `site:195.201.150.203` auf Google führen zu keinerlei Ergebnissen.

Die Scan-Ziele erlauben gemäss Kapitel 3.1.1 ausser Ping-Anfragen keine eingehenden Verbindungen und haben somit zum Internet keine Ports geöffnet. Ein Scan-Ziel könnte theoretisch im Zeitraum zwischen der Bereitstellung des Servers bis zu dessen Konfiguration inklusive Firewall-Einstellungen gescannt werden.

Zuvor betriebene Konfigurationen ehemaliger Benutzerinnen und Benutzer unter derselben IPv4-Adresse können vereinzelt in Suchmaschinen aufgefunden werden.

[54] Korrekt wäre Brasilien, Zeek gibt hierzu in Kapitel 3.9 Russland an

4.10. Persönliche Interpretation / Diskussion

Dieser Abschnitt enthält Hypothesen und Deutungen des Autors. Sie können von den Fakten abweichen und dienen der Diskussion ermittelter Ergebnisse. Eine persönliche Meinung des Autors zur allgemeinen Thesis ist Kapitel 5.3 „Rückblick“ zu entnehmen. Der sachliche Ausblick sowie das Fazit befinden sich in Kapitel 5.

4.10.1. IPv6

Im Vergleich zu IPv4 werden kaum IPv6-Verbindungen aufgezeichnet. Nach den Ausführungen in Kapitel 2.1.2 könnten folgende Unternehmungen zu weiterem IPv6-Netzwerkverkehr auf den Scan-Zielen führen:

- Scan-Ziele kommunizieren aktiv mittels IPv6 im Internet indem sie beispielsweise selber Scans durchführen
  Dies führt zu einer Erhöhung der Wahrscheinlichkeit, dass die IPv6-Adresse in einer Hitlist auftaucht, ändert jedoch auch das definierte Verhalten eines Scan-Ziels
- DNS-Einträge bei Domänen anlegen, die mit einer höheren Wahrscheinlichkeit bereits gescannt werden
  Die hier verwendete Domäne `pinelair.com` existiert erst seit dem 17. Januar 2025 [302]
- DNS-Einträge wie `www` oder `mail` könnten zu einer höheren Scan-Wahrscheinlichkeit führen, gewichten jedoch einzelne Scan-Ziele höher als andere (sofern alle bei derselben Domäne eingetragen werden)

Die Anzahl an eingetroffenen Verbindungen mit IPv6 nach der Erstellung von DNS-Einträgen zu den IPv6-Adressen der Scan-Ziele am 17. Januar 2025 ist nicht gestiegen. Einzig die Scans mittels 6to4 fallen auf (siehe Kapitel 4.2). Diese basieren jedoch auf der IPv4-Adresse, weshalb die Aussage getroffen werden kann, dass sie nicht mit den DNS-Einträgen in Verbindung stehen.

Kapitel 3.3 zur Wahl der Hosting-Dienstleistungsunternehmen zeigt, dass IPv6 auch zum Zeitpunkt dieser Arbeit keine Selbstverständlichkeit ist.

4.10.2. Kontaktierte Ports und JA4+-Fingerprints

Die Auswertung in Kapitel 4.4 zeigt unter den meist kontaktierten Ports viele Einträge, die erfahrungsgemäss mit der Bereitstellung von Webserver-Diensten im Zusammenhang stehen (80, 8080, 443, 8081, 8088, 8888, 8443).

Ebenso wird gezeigt, dass unter Ports 22 und 23 (SSH und Telnet) je über 60 % der detektierten Scans mit bösartigen Absichten aufgezeichnet wird. Die Vermutung liegt nahe, dass Angreifende sich über die meist dahinterliegen Dienste Zugang zu Infrastrukturen verschaffen wollen (beispielsweise über zu simple Login-Daten oder nicht geschlossene Software-Schwachstellen).

Der weitaus meist kontaktierte Ziel-Port gemäss Analyse ist jedoch 8728. Recherchen ergeben, dass dieser Port für die API-Schnittstelle eines Routers der Firma MikroTik verwendet wird [303]. Verwundbare MikroTik Router können mit Malware infiziert und Teil eines Botnets werden [304–306]. Diese, in der Standardausführung offene Schnittstelle ermöglicht mit entsprechenden Zugangsdaten die Analyse und Konfiguration eines solchen Routers [303]. In der Analyse-Umgebung stammen über 99,9 % der Scan-Detektionen zu diesem Port von ZMap. 8728 ist aufgrund dieser API ebenfalls im Dokument von Durumeric et al. aus 2024 einer der meist kontaktierten Ports [27]. 99,5 % der zugehörigen Scans stammen dort von ZMap [27].

Aufgrund der Firewall-Konfiguration eines Scan-Ziels werden lediglich Ping-Anfragen aus dem Internet zugelassen und beantwortet (siehe Kapitel 3.1.1). Daraus ergibt sich, dass JA4+-Fingerprints auf Basis

von TCP-Handshakes oder Anwendungen wie SSH nicht möglich sein sollten. Ausnahme bildet JA4T, das TCP-SYN-Segmente verwendet [275]. Die Anzahl an Einträgen für die jeweiligen JA4+-Methoden in Kapitel 4.6 unterstützt diese Aussage. Dennoch sind auch zu ausgeschlossenen Methoden Einträge vorhanden, wenn auch verhältnismässig wenig (JA4, JA4H, JA4SSH und JA4TS).

Zu JA4SSH werden in Kapitel 4.6 Fingerprints von einer einzelnen Scan-Quelle aufgeführt. Diese Quelle sendet jedoch unabhängig von einer Antwort auf dessen TCP-SYN-Segment ein ACK-Segment sowie SSH-Daten. Dieses SSH-Paket führt zu besagtem JA4SSH-Fingerprint. Aufgrund der Quell-IP-Adresse oder den Paketinhalten wird kein IoC festgestellt. Somit ist die Absicht des Scans nicht erkennbar, kann jedoch gut- oder bösartig ausfallen.

JA4T trägt zur Detektion von Scans hier einen signifikanten Anteil bei, da anhand dessen Fingerprints weitere Scan-Anwendungen identifiziert werden können. Diese Detektionsmethode ist zudem nicht von bekannten IP-Adressen, Domänen oder FQDNs abhängig. Andererseits ist es für eine erfolgreiche Detektion mit JA4T ebenfalls unverzichtbar, eine Datenbank mit bekannten Fingerprints zu konsultieren.

In Kapitel 4.6 mittels JA4T ermittelte Betriebssysteme ermöglichen keine Ermittlung der Scan-Absicht. Kommunikationen des dort beobachteten Thermostats [307] könnten u. a. aus folgenden Gründen auftreten:

- Das Gerät ist kompromittiert und es werden Versuche zur weiteren Ausbreitung von dessen installierter Schadsoftware unternommen
- Es liegt ein Konfigurationsfehler vor, der zur Kontaktaufnahme zu einem der Scan-Ziele führt

Die Ports in Abbildung 4.7 aus Kapitel 4.4 treten in der Arbeit von Durumeric et al. aus 2024 ebenfalls auf, bis auf folgende Ausnahmen[55]:

- **34567**: Recherchen ergeben, dass Geräte des Typs „Network Video Recorder (NVR)“ der Firma Xiongmai unter diesem Port verwundbar sind [308–310]. Im Vergleich zu anderen Top-Ports dieser Arbeit erfährt Port 34567 7,53 % der Scan-Detektionen mit Intention „bad“ . Dies liegt über dem Durchschnitt entsprechender Scan-Detektionen von 5,36 % (siehe Tabelle 4.3)
- **53**: In Tabelle 4.2 aufgeführte DNS-Anfragen können bei einer modifizierten Quell-Adresse zu einer DDoS-Attacke des Typs „DNS Amplification“ beitragen, sofern hinter dem Port ein entsprechend antwortender DNS-Serverdienst liegen würde [311, 312]
- **123**: Im Rahmen dieser Arbeit werden keine Indizien für eine DDoS-Attacke des Typs „NTP Amplification“ festgestellt, können jedoch nicht ausgeschlossen werden [313, 314]
- **161**: Das Dashboard „SNMP“ von Malcolm in OpenSearch Dashboards zeigt über fünftausend SNMP-Anfragen mit dem „Community String“ `public` [56]. Erfahrungsgemäss handelt es sich hierbei um eine Standardangabe unter welcher Informationen über Netzwerkkomponenten mittels SNMP ausgelesen werden können
- **389**: Zum zugehörigen Protokoll LDAP sind im gleichnamigen Malcolm-Dashboard in OpenSearch Dashboards über viertausend Anfragen aufgeführt [57]. Es wird versucht auf Informationen in allfällig verfügbaren Verzeichnisdiensten zu zugreifen

Aussagen zur Intention der Anfragen sind in Kapitel 4.10.5 aufgeführt.

[55] Weitere Informationen zu den Top-Ports dieser Arbeit sind Tabelle 4.2 in Kapitel 4.4 zu entnehmen
[56] Zusätzlich angewendeter Filter `destination.port: 161`
[57] Zusätzlich angewendeter Filter `destination.port: 389`

4.10.3. Standortangaben

In Kapitel 3.5.2 „Geografische Merkmale“ werden Datenbanken mit Standortangaben zu IP-Adressen aufgeführt, die keine 100-prozentige Genauigkeit versprechen.

Wilhoit ermittelt 2013 mittels Skripts im Webbrowser den Standort einer Quelle [24]. Dazu werden u. a. WLAN-Access-Points in der physischen Nähe ausgewertet [24]. Durumeric et al. weist als verwendete Datenbank „GeoIP“ von MaxMind aus [14, 40]. Datenbanken desselben Herstellers werden auch von Malcolm verwendet, jedoch in ungenaueren, kostenfreien Varianten („GeoLite2“ [222]) [107, 221, 223]. Heo und Shin weisen ebenfalls „GeoLite2“ als Quelle aus [22].

MaxMind beschreibt IP-Geolokalisierung als grundsätzlich ungenau [222]. Richter und Berger verwenden zur Lokalisierung eine proprietäre Datenbank [25]. Serbanescu et al. erwähnen „GeoIP“ als Quelle für die Standortangabe, jedoch nicht, ob es sich hierbei um die Datenbank von MaxMind oder einer anderen Quelle handelt.

Die in Kapitel 3.5.2 aufgezeigten Unterschiede zwischen den Daten von Arkime und Zeek sowie den tatsächlichen Standorten der Scan-Ziele weisen auf eine bestimmte Ungenauigkeit hin. Dies obwohl „GeoLite2“-Datenbanken für beide Komponenten verwendet werden, aber anscheinend nicht derselben Version entsprechen [107, 221, 223]. Wird eine Verbindung mittels Tor aufgebaut, können vom verwendeten Tor Exit Node keine Rückschlüsse auf die Quell-IP-Adresse gezogen werden [103]. Dann ist lediglich die Adresse des Tor Exit Nodes ersichtlich [102, 104, 105].

Diese Erkenntnisse stellen die Angaben bei einer IP-Geolokalisierung allgemein in Frage.

4.10.4. Vergleich mit Arbeit von Heo und Shin

Wie in der Einleitung in Kapitel 1 beschrieben, kommen Heo und Shin 2018 zum Ergebnis, dass sich annähernd 5 % der Scan-Quellen öffentlich ausweisen [22]. Gerundet 700 von 3,78 Millionen Scan-IP-Adressen werden hierbei als „verantwortungsvoll“ deklariert und 2,65 Billionen[58] Scans identifiziert [22]. Heo und Shin haben dazu Log-Informationen aus 31 Tagen zweier Firewalls ihres Campus ausgewertet, die produktiv verwendet werden [22]. Besagte Firewalls sind beide zusammen in einem Cluster mit Aktiv-Aktiv-Konfiguration an einem 16-Bit und zwei 20-Bit IPv4-Netzwerksegmenten angebunden (73’728 zugewiesene IP-Adressen)[59] [22].

Die Scan-Ziele dieser Thesis erfahren keinen produktiven Netzwerkverkehr, wobei jede Anfrage als Scan gewertet wird. Mit 10 Scan-Zielen sind 10 IPv4-Zieladressen vertreten. Dessen Daten werden im Zeitraum von 40,625 Tagen betrachtet[60].

Die Ansätze zur Detektion von Scan-Quellen unterscheiden sich zwischen dieser und der Arbeit von Heo und Shin [22]. Daher ist kein direkter Vergleich zwischen sämtlichen ermittelten Daten möglich. Dennoch wird nachfolgend ein Versuch unternommen:

Tabelle 4.4.: Persönliche Interpretation: Vergleich mit Arbeit von Heo und Shin [22]

Beschreibung	Heo und Shin	Diese Arbeit
Zeitraum	31 Tage Juni/Juli 2016	40,625 Tage 27.12.24 bis 9.2.25 minus 4,375 Tage
IPv4-Zieladressen	73’728	10 [61]
IPv4-Quelladressen	3,78 Millionen	181’541 [62]
IPv4-Quelladressen mit ausgewiesenen bzw. als „good“ deklarierten Scans	700	13’135 [63]
Scans Eine Zeek-Verbindung wird hier als Scan gewertet, nicht die Scan-Detektionen	2,65 Billionen ≈ 85,484 Millionen pro Tag	6’479’500 [64] ≈ 159’495 pro Tag
Scans pro Ziel-IP-Adressen im Durchschnitt	≈ 35’943 ≈ 1159 pro Tag	647’950 ≈ 15’949 pro Tag

Aufgrund der verhältnismässig hohen Anzahl an Ziel-IP-Adressen bei Heo und Shin fällt der Durchschnittswert in der letzten Tabellenzeile kleiner aus. Es ist vorstellbar, dass nicht jede dieser Adressen gleich oft kontaktiert wird.

Pro Tag erfahrene Scans (zweitletzte Tabellenzeile) fallen bei den Campus-Firewalls von Heo und Shin über das 500-fache höher aus. Hierbei handelt es sich um aktiv und produktiv verwendete Komponenten gegenüber den Scan-Zielen dieser Arbeit, die ohne grosse Interaktionen mit anderen

[58] Annahme: Mit einer Billion ist 10^9 gemeint [315]

[59] Die Arbeit zählt die Netzwerk- und Broadcast-Adresse pro Netzwerksegment mit:
$2^{32-16} - 2 + 2 * (2^{32-20} - 2) = 2^{16} - 2 + 2 * (2^{12} - 2) = 65534 + 2 * 4094 = 73722$

[60] 27. Dezember 2024 bis und mit 9. Februar 2025 abzüglich dem 13. bis und mit 17. Januar 2025 um 9 Uhr, siehe Kapitel 4

[61] Aufgrund der geringen Anzahl an IPv6-Scan-Detektionen (siehe Kapitel 4.2) werden diese hier ausgeklammert

[62] Ermittelt in OpenSearch Dashboards durch Erstellung einer „Metric“-Visualisierung mit folgenden Eigenschaften: `Aggregation: Unique Count` und `Field: source.ip` ohne Filter `rule.category: ScannerDetection`, mit zusätzlichem Filter `network.type: ipv4`

[63] Ermittelt in Dashboard „Scanner Detection“ in Visualisierung „Notices - Unique Source IP Count“ mit zusätzlichen Filter `zeek.notice.msg: *good*` und `network.type: ipv4`

[64] Ermittelt mit Visualisierung „Connections - Log Count“ in OpenSearch Dashboards mit zusätzlichem Filter `network.type: ipv4`

Komponenten im Internet platziert werden. Ausnahme bilden hierbei die automatisierten Debian-Aktualisierungsvorgänge, deren Zeitsynchronisation mittels NTP, zugehörige DNS-Anfragen sowie den WireGuard-Tunnel zum VPN-Server.

Eine Vermutung daraus lautet, dass die Anzahl an Scans zwischen den genannten Zeiträumen gestiegen ist.

Das Dokument von Durumeric et al. zeigt einen allgemeinen Wachstum von Scans im Internet zwischen 2016 und 2024 (siehe Abbildung 4.18) [27].

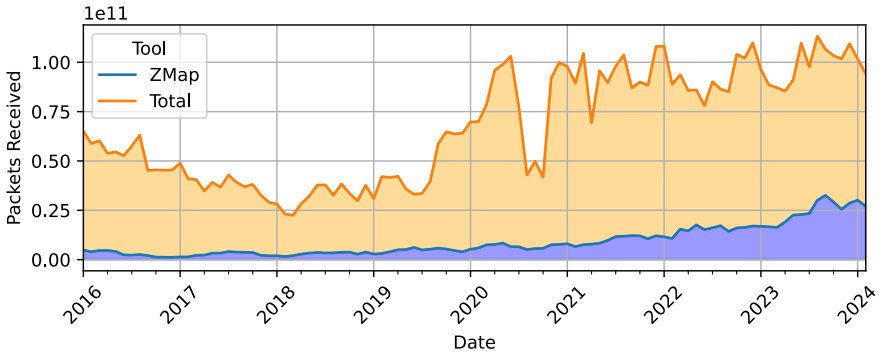


Abbildung 4.18.: ZMap zugeschriebener Scan-Netzwerkverkehr („Figure 1“ aus Durumeric et al. [27])

Kapitel 4.3 und Tabelle 4.3 in Kapitel 4.7 zeigen auch, dass die Scan-Ziele eine unterschiedliche Menge an Kommunikationen erfahren. Mögliche Faktoren hierzu können der Standort sein, aber auch das gewählte Hosting-Dienstleistungsunternehmen oder auch zuvor aufgeschaltete Dienste unter derselben IP-Adresse.

Die Arbeit von Heo und Shin erwähnt ZMap nicht, wobei diese Anwendung in dieser Thesis einen grossen Anteil detektierter Scans ausmacht [22]. ZMap existiert seit 2013, wobei dessen Wachstum seitdem zugenommen hat (siehe Abbildung 4.18) [27].

Shodan und Shadowserver haben bei Heo und Shin zwei Drittel der Scan-Quellen ausgemacht, während ihre Anteile hier geringer ausfallen (1,56 % und 3,94 %, siehe Abbildung 4.11 in Kapitel 4.7) [22].

Der hier meist kontaktierte Port 8728 (siehe Kapitel 4.4) wird von Heo und Shin nicht erwähnt [22]. Dies liegt mit einer hohen Wahrscheinlichkeit daran, dass dieser Port im Zusammenhang mit MikroTik Routern verwendet wird, deren Schwachstellen-Ausnutzung ab 2018 an Verbreitung zunahm (nach dem Analyse-Zeitraum von Heo und Shin) [305].

4.10.5. Intention der Scan-Quellen und Aussagekraft

Der grösste Teil hier detektierter Scans beruht auf vorhandenen Daten zu bekannten Scan-Quellen aus Kapitel 3.1.3 sowie der Detektion von ZMap. Somit leistet die Qualität der Informationen in den Tabellen des Zeek-Skripts einen signifikanten Beitrag. Regelmässiges Prüfen der Zeek-Logs und allfällige Erweiterungen der Daten bzw. Tabellen erhöhen die Qualität. Automatisiert angereicherte Daten sind auf langfristige Sicht regelmässig zu bereinigen, da beispielsweise IP-Adressen ihre Eigentümerin bzw. Eigentümer sowie ihren Verwendungszweck ändern können[65].

Bezüglich der Detektion einer Out-of-Band Probe Attribution gemäss RFC 9511 hat diese im aktuellen Stand nur nach weiteren, manuellen Analysen eine Aussagekraft. Wie in Kapitel 4.1.4 gezeigt, entspricht aktuell jede solcher Detektionen einer Fehldetektion. Unter anderem werden neben der Domäne `showfreevids.com` ein Onlineshop und eine Webmail-Login-Maske hinter den detektierten URIs festgestellt. Von keiner dieser Seiten sind Scans zu erwarten, was sie eher als bösartige oder kompromittierte Quelle einstufen lassen könnte.

Durumeric et al. erwähnen, dass im ersten Quartal von 2024 35 % des Internet-weiten Scan-Verkehrs von ZMap stammt [27]. Im Rahmen dieser Arbeit erfährt die ZMap-Detektion einen Scan-Anteil von 45,794 % (siehe Abbildung 4.10 in Kapitel 4.7). Bei Betrachtung der Abbildung 4.18 in Kapitel 4.10.2 könnte eine mögliche Aussage sein, dass der Anteil von ZMap an Internet-weiten Scans weiter zugenommen hat.

Scan-Detektionen mit negativer Intention („bad“) werden hauptsächlich anhand bekannter IP-Adressen ausgelöst. Detektionen desselben Typs mittels JA4+-Fingerprints treten in dieser Arbeit nicht auf (mit allfälliger Ausnahme des Thermostats in Kapitel 4.10.2). Das Betreiben von Diensten auf den Scan-Zielen würde weitere Kommunikationen mit sich bringen und die Detektionswahrscheinlichkeit unter der Anwendung von JA4+ erhöhen. In der aktuellen Variante der Analyse-Umgebung werden lediglich Ping-Anfragen akzeptiert und andere Protokolle auf der Firewall blockiert (siehe Kapitel 3.1.1).

Andere, mitgelieferte Dashboards von Malcolm können unabhängig des entwickelten Zeek-Skripts weitere Informationen aufzeigen. Die gesammelten Daten könnten bei zusätzlichen, aktuell manuellen Analysen zusätzliche Erkenntnisse mit sich bringen, die im Rahmen dieser Arbeit u. a. aus Zeitgründen unentdeckt bleiben.

Einzelne Betrachtungen kontaktierter Ports führen zur Vermutung, dass detektierte Scans mit positiver Intention („good“) zu Aufklärungszwecken verwendet werden. Scan-Quellen mit negativ detektierten Intentionen könnten ebenfalls diesen Zwecken dienen, mit dem Unterschied, dass diese für allfällige Angriffe verwendet werden könnten.

Aufgrund, dass die Autorschaft dieser Arbeit aus einer Person besteht, werden die erhobenen Informationen von keiner weiteren Person kontrolliert. Dies erhöht die Wahrscheinlichkeit für Fehler bei der Detektion oder Interpretation, die trotz sorgfältiger Auseinandersetzung nicht komplett auszuschliessen sind.

Beispiele wie der Inhalt `nxp-scan` in UDP-Datagrammen zum Ziel-Port 22 [66] zeigen, dass es Scan-Quellen gibt, die eine Scan-Durchführung innerhalb eines Netzwerkpakets ausweisen. Institutionen wie Shodan oder Censys, die sich auf ihren Webseiten ausweisen, zeigen ebenfalls den Willen, Scans im Internet ethisch durchzuführen. Eine Standardisierung zur Identifizierung solcher Scans würde eine entsprechende Detektion vereinfachen. Dies wäre vorstellbar durch die Adaption der Vorschläge gemäss RFC 9511 „Attribution of Internet Probes“ [15].

[65] Dies ist nach dem Analyse-Zeitraum dieser Arbeit implementiert und dokumentiert (siehe Kapitel C.2.4). Auf eine Implementation innerhalb des Analyse-Zeitraums wird verzichtet, um währenddessen das Analyse-Verhalten nicht zu verändern

[66] Siehe Tabelle 4.2 in Kapitel 4.4

5. Abschluss

In diesem Kapitel befinden sich abschliessende Aussagen dieser Arbeit. Sie gehen auf die Fragen und Zielsetzungen aus dem ersten Kapitel („Einleitung“) ein. Die persönliche Interpretation sowie Diskussion befindet sich in Kapitel 4.10.

5.1. Zusammenfassung

Die im Rahmen dieser Arbeit nachvollziehbar aufgebaute und erweiterbare Analyse-Umgebung wertet den aktuellen Netzwerkverkehr aus, den sie von möglichst weltweit platzierten Scan-Zielen erfährt.

Auswertungen erfolgen anhand Kommunikationsmerkmalen wie IP-Adressen oder JA4+-Fingerprints. Scans weisen sich unter Anwendung bestimmter Applikationen wie ZMap oder bekannter Adressen (mittels IP oder DNS) aus. Ein sich ausweisender Scan wird mit der Intention „good“ detektiert und entsprechend vermerkt. Stammt ein Scan von einer Adresse, die in Verbindung mit Malware oder sonstigen bösartigen Aktivitäten steht, wird diese mit der Intention „bad“ detektiert. Somit entsteht eine automatisierte Klassierung entsprechend der Scan-Intentionen.

Zu Beginn dieser Arbeit wird die Hypothese aufgestellt, dass unter 10 % der Scan-Quellen sich mit ihren Absichten öffentlich erkenntlich zeigen. Diese Hypothese baut unter anderem auf der Arbeit von Heo und Shin auf, in der sich ungefähr 5 % des Scan-Verkehrs öffentlich ausweist [22]. Heo und Shin haben jedoch produktiv verwendete Firewalls an einem Standort ausgewertet und die Scan-Detektionen ohne die Betrachtung von ZMap oder den Vergleich mit bekannten IP-Adressen durchgeführt. Die Scan-Ziele dieser Arbeit werden nicht für produktiven Netzwerkverkehr verwendet. Dies verhindert einen direkten Vergleich zwischen der Arbeit von Heo und Shin und dieser. Der Anteil von Detektionen zu ZMap und bekannten IP-Adressen mit positiv vermerkten Intentionen fällt zu über 77 % aus (siehe `ZMap` und `KnownIP` in Abbildung 4.10).

Das Ergebnis dieser Arbeit zeigt, dass innerhalb des betrachteten Zeitrahmens (ca. 40 Tage um und mit Januar 2025) im Durchschnitt ungefähr 95 % der Scan-Detektionen mit Intention „good“ erkannt werden. Die restlichen 5 % entsprechen hierbei Scan-Detektionen mit Intention „bad“. Je nach Scan-Ziel und dessen Platzierung erfahren Detektionen mit Intention „bad“ Anteile zwischen ungefähr 2,5 bis 10,5 % (siehe Tabelle 4.3). Die Detektionen stützen sich hierbei zum Grossteil auf öffentlich bekannte Daten wie IP-Adressen, Domänen oder FQDNs zu diversen Institutionen.

Scan-Ziele dieser Arbeit beantworten im Internet lediglich Ping-Anfragen, aber keine TCP- oder UDP-Kommunikationen. Dies schränkt die Interaktionen zwischen Scan-Quellen und Scan-Zielen ein, was Detektionen aufgrund von Paketinhalten in dieser Arbeit ausschliesst. Öffentlich verfügbare Dienste wie beispielsweise Webserver-Dienste könnten einen Einfluss auf die erfahrenen Scan-Detektionen ausüben, wobei diese bereits unabhängig von der Verfügbarkeit geprüft werden (siehe kontaktierte Ports in Kapitel 4.4).

Verbindungen mit IPv6 erreichen unabhängig von der Intention einen Anteil von 0,007 % (siehe Tabelle 4.1). Entsprechend Kapitel 2.1.2 klammern viele Arbeiten IPv6 aus, wozu hier lediglich Scans mit Intention „good“ ermittelt werden.

Daraus ergibt sich folgende Erwartungshaltung beim Anschluss eines Zugangspunkts am Internet: Über die öffentliche IPv4-Adresse sind Scans diverser Institutionen zu erwarten, von denen im Durchschnitt 5 % mit der Intention „bad“ ausfallen. Der Anteil an IPv6-Verbindungen fällt im Vergleich marginal aus. Hierbei stellt sich die Frage, ob der IPv6-Anteil mit der eigenen Nutzung beziehungsweise Vorkommen im Internet-Netzwerkverkehr steigen würde.

5.2. Fazit und Ausblick

Das zum Zeitpunkt dieser Arbeit aktuelle Verhalten von Scans im Internet zu Scan-Zielen mit offener Interaktion unter ICMP (Ping-Anfragen) wird hiermit abgebildet. Zusätzlich werden die aktuellen Anteile beobachteter Scan-Institutionen und -Anwendungen aufgezeigt.

In der Praxis bieten öffentlich erreichbare Komponenten Dienste unter entsprechenden Ports an, die von diversen Institutionen weiter geprüft werden können. Diese Arbeit bestätigt, dass unabhängig davon auch nicht ansprechbare Ports geprüft werden. Daraus gilt es weiterhin zu beachten, dass öffentlich angebotene Dienste auf das Nötigste eingeschränkt, möglichst abgesichert und aktuell zu halten sind.

Weiterhin zeigt das Einsehen öffentlicher Informationen zu den IP-Adressen der Scan-Ziele in Kapitel 4.9, dass diese zuvor von anderen Eigentümerinnen und Eigentümer genutzt wurden und gar noch aktive DNS-Einträge dazu vorhanden sein können.

Scan-Institutionen weisen ihre Adressen meist unter der eigenen Webseite öffentlich aus, unter welchen diese Scans durchführen. Weitere Analysen bereits erfahrener Kommunikationen bringen zuvor unbekannte Scan-Institutionen hervor, die erst nach entsprechender Eintragung in der Analyse-Umgebung detektiert werden können. Somit wird empfohlen, die Logs der Analyse-Umgebung regelmässig nach zuvor unbekannten Quellen zu prüfen. Eine Scan-Anwendung wie ZMap oder masscan kann anhand bestimmter Kommunikationsmerkmale oder JA4+-Fingerprints detektiert werden. Der RFC 9511 Artikel bietet Vorschläge zu einer Identifikation von Scan-Quellen, ohne dass die zugehörige Institution zuvor bekannt sein muss [15]. Eine Anwendung eines solchen Vorschlags konnte im Rahmen dieser Arbeit nicht detektiert werden. Die zugehörige Implementation zur Detektion der „Out-of-Band Probe Attribution“ gemäss RFC 9511 erfährt derzeit Fehl-Detektionen. Diese könnten durch eine Validierung mittels ABNF-Regeln aus RFC 9116 minimiert werden [15, 96].

Die Analyse-Umgebung könnte durch eine automatisierte Auswertung ermittelter JA4+-Fingerprints oder Protokoll-spezifischer Verhalten erweitert werden. Hierbei sind Detektionen für beispielsweise DDoS-Amplification-Angriffe mittels DNS oder NTP vorstellbar [311, 313]. Eine weitere mögliche Erweiterung wären weiterführende, automatisierte Scans der Scan-Quellen, um beispielsweise Command & Control Server zu identifizieren. Dazu erstellte Scan-Anfragen könnten entsprechend RFC 9511 ausfallen, um dessen Verwendung im Internet zu unterstützen. Scans, bei denen viele Anfragen von einzelnen Quellen in einer kurzen Zeit entstehen, könnten als unethisch markiert werden.

Weiterhin erlaubt die Analyse-Umgebung den Aufbau von Scan-Zielen, die sich am selben Standort und Dienstleistungsunternehmen befinden, jedoch unter verschiedenen Adressen erreichbar sind und sich unterschiedlich verhalten. Dies würde einen Vergleich erfahrener Kommunikationen zu bestimmten Verhaltensmustern oder Konfigurationen ermöglichen. Dies kann den Betrieb eines öffentlich verfügbaren Serverdienstes, Erstellen eines zugehörigen DNS-Eintrags, Platzierung eines Scan-Ziels im Tor Netzwerk oder die ausschliessliche Verwendung von IPv6 beinhalten. Malcolm bietet bereits Auswertungsmöglichkeiten zu einer Auswahl an Protokollen, die hierzu zugezogen werden können. Dadurch, dass Scan-Ziele ihren Netzwerkverkehr ohne diesen selbst aufzuzeichnen zum zentralen Sensor replizieren und anhand Ansible konfiguriert werden, sind sie nach Bedarf auf- und abbaubar. Je nach Implementation gilt es zusätzliche Sicherheitsmassnahmen an den Scan-Zielen vorzunehmen.

Bei allfälligen Erweiterungen der Analyse-Umgebung um Funktionalitäten oder Scan-Ziele gilt es entsprechende Ressourcenansprüche zu beachten. Je nach verfügbaren Ressourcen können auch von den Scan-Zielen ausgehende Kommunikationen repliziert und analysiert werden. Gegebenenfalls können Optimierungen des hierzu erstellten Zeek-Skripts zur Scan-Detektion die Ressourcenansprüche verringern.

Für den weiterführenden Betrieb der Analyse-Umgebung wird empfohlen, die Malcolm-Komponenten und dessen Dienste zu überwachen. Netzwerk-Informationen der ERSPAN-Interfaces pro Scan-Ziel und VPN-Statistiken auf dem Sensor/VPN-Server mit Hedgehog Linux bieten Hinweise auf die Verfügbarkeit und Funktionalität einzelner Scan-Ziele.

Zur Aktualisierung des Betriebssystems und der Systemzeit sowie für DNS-Anfragen kommunizieren die Scan-Ziele aktuell mit öffentlichen Servern im Internet. Die zugehörigen Ressourcen könnten lokal aufgebaut und über den etablierten VPN-Tunnel bezogen werden. Somit würde die Kommunikation zwischen Scan-Zielen und fremden Infrastrukturen minimiert werden.

Betrachtungen über einen längeren Zeitraum könnten zu weiteren Erkenntnissen oder beispielsweise Scan-Detektionen mit IPv6 führen. Eventuell treten zukünftig auch Detektionen zu Scans entsprechend den Vorschlägen aus RFC 9511 in Erscheinung.

5.3. Rückblick

Dieser Abschnitt bildet den persönlichen Rückblick des Autors.

Die Durchführung dieser Arbeit war für mich sehr lehrreich und ermöglichte das Anwenden und Festigen des erarbeiteten Wissens aus der zugehörigen Cyber-Security-Ausbildung. Gewonnene Erfahrungen besonders mit den Produkten Zeek, Arkime und OpenSearch Dashboards können in zukünftigen Aufgaben produktiv eingesetzt werden.

Ansprüche an die gegebenen Ressourcen und Limitierungen habe ich zu Beginn unterschätzt, was zu weiteren Optimierungsmassnahmen, Ersatzlösungen oder Erweiterungen führte. Die verhältnismässig geringe Verbreitung oder Unterstützung von IPv6 bei den Hosting-Dienstleistungsunternehmen sowie beobachteten Verbindungen hat mich trotz eigener Erfahrungen überrascht und zeigt, dass IPv4 immer noch das weitaus präferierte Internet-Protokoll zu sein scheint.

Mit grossem Wert auf Nachvollziehbarkeit und dessen weiterführendem Betrieb konnte ich eine erweiterbare Analyse-Umgebung aufbauen, die Verbindungen von und zu den Scan-Zielen zentral aufzeichnet und auswertet. Zur Detektion erstellte Tabellen mit bekannten Informationen zu Scan-Institutionen enthalten bei einer automatischen Anreicherung zugehörige Quellenverweise, die gegebenenfalls zu weiteren Informationen über die Scan-Quellen führen. Ohne diese Tabellen würde die Anzahl an Detektionen wesentlich geringer ausfallen.

Zu viel erzeugte Detektionen durch IP-Adressen, die gleichzeitig in Tabellen mit bekannten IP-Adressen und IP-Subnetzen vorkommen, konnte ich durch entsprechende Filter nachträglich abfangen und im zugehörigen Zeek-Skript korrigieren.

Ermittelte Ergebnisse aus den Auswertungen betragen einen umfangreichen Anteil dieser Arbeit, können jedoch aus Zeit- und Übersichtsgründen nicht vollständig abgebildet werden. Umso mehr freut es mich, dass ich die Analyse-Umgebung inklusive aller Daten dem Themensponsor übergeben kann.

Der hier eingeflossene Aufwand überschreitet die Empfehlung von 360 Stunden um über 20 %, zeigt jedoch auch meine Begeisterung am Thema sowie der Arbeit. Trotzdem gibt es Punkte, die aus Zeitgründen nicht weiter verfolgt werden konnten und ich gerne angeschaut hätte. Die Durchführung als Einzelarbeit bringt mehr Flexibilität und Kontrolle mit sich, schliesst jedoch die Infragestellung einzelner Implementationen, Massnahmen oder Schlussfolgerungen durch eine weitere Person aus (mit Ausnahme von Reviews durch die Experten).

Ich bedanke mich bei den Experten dieser Arbeit Hansjürg Wenger, Bruce Nikkel und Rolf Lanz sowie Donatello Gallucci für den Austausch und die Unterstützung während dieser Arbeit. Weiterer Dank gilt meinem Betrieb für die Unterstützung während der gesamten Ausbildung zum MAS Cyber Security und den Mitarbeitenden der Berner Fachhochschule, die zu meiner Ausbildung und der hier verwendeten LaTeX-Vorlage [316] beigetragen haben.

Verzeichnisse

Literaturverzeichnis

[1] S and V Design, *Fingerprint scanning on circuit board. secure system concept with a fingerprint. Cyber security technology concept abstract background futuristic Hi-tech style. Vector and Illustration*, 24. März 2024. besucht am 7. Aug. 2024. Adresse: https://www.shutterstock.com/image-vector/fingerprint-scanning-on-circuit-board-secure-2441137059.

[2] M. Peischl und A. Habegger, *BFH-CI – Corporate Design LaTeX Templates for Bern University of Applied Sciences*, 27. Juni 2024. besucht am 9. Dez. 2024. Adresse: https://gitlab.ti.bfh.ch/bfh-latex/bfh-ci.

[3] Font Awesome und M. Krüger, *The fontawesome5 package*, 2. Mai 2022. besucht am 30. Nov. 2024. Adresse: http://mirrors.ibiblio.org/CTAN/fonts/fontawesome5/doc/fontawesome5.pdf.

[4] W. Haager, *worldflags – Drawing flags with TikZ*, 12. Nov. 2023. besucht am 10. Dez. 2024. Adresse: https://www.ctan.org/pkg/worldflags.

[5] M. Rohsopht, *Flag of Hong Kong*, 6. Sep. 2021. besucht am 8. Jan. 2025. Adresse: https://commons.wikimedia.org/wiki/File:Flag_of_Hong_Kong.svg.

[6] *Internet Protocol*, RFC 791, Sep. 1981. DOI: 10.17487/RFC0791. Adresse: https://www.rfc-editor.org/info/rfc791.

[7] D. S. E. Deering und B. Hinden, *IP Version 6 Addressing Architecture*, RFC 4291, Feb. 2006. DOI: 10.17487/RFC4291. Adresse: https://www.rfc-editor.org/info/rfc4291.

[8] *Internet Control Message Protocol*, RFC 792, Sep. 1981. DOI: 10.17487/RFC0792. Adresse: https://www.rfc-editor.org/info/rfc792.

[9] M. Gupta und A. Conta, *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*, RFC 4443, März 2006. DOI: 10.17487/RFC4443. Adresse: https://www.rfc-editor.org/info/rfc4443.

[10] W. Eddy, *Transmission Control Protocol (TCP)*, RFC 9293, Aug. 2022. DOI: 10.17487/RFC9293. Adresse: https://www.rfc-editor.org/info/rfc9293.

[11] *User Datagram Protocol*, RFC 768, Aug. 1980. DOI: 10.17487/RFC0768. Adresse: https://www.rfc-editor.org/info/rfc768.

[12] T. A. Ahanger, "Port Scan - A Security Concern," *International Journal of Engineering and Innovative Technology (IJEIT)*, Jg. 13, Nr. 10, S. 241–246, 2014, ISSN: 2277-3754. Adresse: https://www.ijeit.com/archive/30/volume-3issue-10april-2014.html.

[13] A. V. Serbanescu, S. Obermeier und D.-Y. Yu, "ICS threat analysis using a large-scale honeynet," in *Proceedings of the 3rd International Symposium for ICS & SCADA Cyber Security Research*, Ser. ICS-CSR '15, Ingolstadt, Germany: BCS Learning & Development Ltd., 2015, S. 20–30, ISBN: 9781780173177. DOI: 10.14236/ewic/ICS2015.3. Adresse: https://doi.org/10.14236/ewic/ICS2015.3.

[14] Z. Durumeric, M. Bailey und J. A. Halderman, "An internet-wide view of internet-wide scanning," in *Proceedings of the 23rd USENIX Conference on Security Symposium*, Ser. SEC'14, San Diego, CA: USENIX Association, 2014, S. 65–78, ISBN: 9781931971157.

[15] Éric Vyncke, B. Donnet und J. Iurman, *Attribution of Internet Probes*, RFC 9511, Nov. 2023. DOI: 10.17487/RFC9511. Adresse: https://www.rfc-editor.org/info/rfc9511.

[16] R. Trapickin, "Who Is Scanning the Internet?" In *Proceedings of the Seminars Future Internet (FI) and Innovative Internet Technologies and Mobile Communications (IITM), Summer Semester 2015*, G. Carle, D. Raumer und L. Schwaighofer, Hrsg., Ser. Network Architectures and Services (NET), Bd. NET-2015-09-1, Munich, Germany: Chair for Network Architectures und Services, Department of Computer Science, Technische Universität München, März 2015, S. 81–88. DOI:

10.2313/NET-2015-09-1_11. Adresse: http://www.net.in.tum.de/fileadmin/TUM/NET/NET-2015-09-1/NET-2015-09-1_11.pdf.

[17] National Cyber Security Centre, *NCSC Scanning information*, 1. Nov. 2022. besucht am 14. Sep. 2024. Adresse: https://www.ncsc.gov.uk/pdfs/information/ncsc-scanning-information.pdf.

[18] M. Collins, *Acknowledged Scanners*, 17. Dez. 2023. besucht am 18. Sep. 2024. Adresse: https://gitlab.com/mcollins_at_isi/acknowledged_scanners.

[19] Shodan, *Shodan Search Engine - Search Engine for the Internet of Things*. besucht am 16. Okt. 2024. Adresse: https://www.shodan.io/.

[20] Censys, *Censys Search*. besucht am 16. Okt. 2024. Adresse: https://search.censys.io/.

[21] The Shadowserver Foundation, *Shadowserver - Lighting the way to a more secure Internet*. besucht am 16. Okt. 2024. Adresse: https://www.shadowserver.org/.

[22] H. Heo und S. Shin, "Who is knocking on the telnet port: A large-scale empirical study of network scanning," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, S. 625–636.

[23] X. Wang, "On the feasibility of real-time cyber attack attribution on the Internet," in *MILCOM 2016 - 2016 IEEE Military Communications Conference*, 2016, S. 289–294. DOI: 10.1109/MILCOM.2016.7795341.

[24] K. Wilhoit, "The SCADA That Didn't Cry Wolf: Who's Really Attacking Your ICS Equipment? (Part 2)," 27. Aug. 2013. Adresse: https://documents.trendmicro.com/assets/white_papers/wp-the-scada-that-didnt-cry-wolf.pdf.

[25] P. Richter und A. Berger, "Scanning the Scanners: Sensing the Internet from a Massively Distributed Network Telescope," in *Proceedings of the Internet Measurement Conference*, Ser. IMC '19, Amsterdam, Netherlands: Association for Computing Machinery, 2019, S. 144–157, ISBN: 9781450369480. DOI: 10.1145/3355369.3355595. Adresse: https://doi.org/10.1145/3355369.3355595.

[26] Berner Fachhochschule, *Cyber Lab*. besucht am 16. Okt. 2024. Adresse: https://www.bfh.ch/de/forschung/alle-dienstleistungen/cyber-lab/.

[27] Z. Durumeric, D. Adrian, P. Stephens, E. Wustrow und J. A. Halderman, *Ten Years of ZMap*, 2024. arXiv: 2406.15585 [cs.CR]. Adresse: https://arxiv.org/abs/2406.15585.

[28] G. Wan et al., "On the Origin of Scanning: The Impact of Location on Internet-Wide Scans," in *Proceedings of the ACM Internet Measurement Conference*, Ser. IMC '20, Virtual Event, USA: Association for Computing Machinery, 2020, S. 662–679, ISBN: 9781450381383. DOI: 10.1145/3419394.3424214. Adresse: https://doi.org/10.1145/3419394.3424214.

[29] P. Richter, O. Gasser und A. Berger, "Illuminating Large-Scale IPv6 Scanning in the Internet," in *ACM Internet Measurement Conference 2022*, Okt. 2022. DOI: 10.1145/3517745.3561452.

[30] A. Tundis, W. Mazurczyk und M. Mühlhäuser, "A review of network vulnerabilities scanning tools: types, capabilities and functioning," in *Proceedings of the 13th International Conference on Availability, Reliability and Security*, Ser. ARES '18, Hamburg, Germany: Association for Computing Machinery, 2018, ISBN: 9781450364485. DOI: 10.1145/3230833.3233287. Adresse: https://doi.org/10.1145/3230833.3233287.

[31] C. Bennett, A. Abdou und P. Oorschot, "Empirical Scanning Analysis of Censys and Shodan," Feb. 2021, ISBN: 1891562673. DOI: 10.14722/madweb.2021.23009.

[32] M. u. Nisa und K. Kifayat, "Detection of Slow Port Scanning Attacks," in *2020 International Conference on Cyber Warfare and Security (ICCWS)*, 2020, S. 1–7. DOI: 10.1109/ICCWS48432.2020.9292389.

[33] E. Bou-Harb, M. Debbabi und C. Assi, "Cyber Scanning: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, Jg. 16, Nr. 3, S. 1496–1519, 2014. DOI: 10.1109/SURV.2013.102913.00020.

[34] A. Murdock, F. Li, P. Bramsen, Z. Durumeric und V. Paxson, "Target generation for internet-wide IPv6 scanning," in *Proceedings of the 2017 Internet Measurement Conference*, Ser. IMC '17,

London, United Kingdom: Association for Computing Machinery, 2017, S. 242–253, ISBN: 9781450351188. DOI: 10.1145/3131365.3131405. Adresse: https://doi.org/10.1145/3131365.3131405.

[35] O. Gasser, Q. Scheitle, S. Gebhard und G. Carle, *Scanning the IPv6 Internet: Towards a Comprehensive Hitlist*, 2016. arXiv: 1607.05179 [cs.NI]. Adresse: https://arxiv.org/abs/1607.05179.

[36] O. Gasser, "Evaluating Network Security Using Internet-wide Measurements," en, Diss., Technische Universität München, 2019, S. 212. Adresse: https://mediatum.ub.tum.de/1473343.

[37] The ZMap Team, *The ZMap Project*. besucht am 23. Okt. 2024. Adresse: https://zmap.io/.

[38] F. Gont und T. Chown, *Network Reconnaissance in IPv6 Networks*, RFC 7707, März 2016. DOI: 10.17487/RFC7707. Adresse: https://www.rfc-editor.org/info/rfc7707.

[39] O. Gasser et al., "Clusters in the Expanse: Understanding and Unbiasing IPv6 Hitlists," in *Proceedings of the 2018 Internet Measurement Conference*, Boston, MA, USA: ACM, 2018. DOI: 10.1145/3278532.3278564.

[40] MaxMind, Inc., *MaxMind GeoIP Databases*. besucht am 17. Jan. 2025. Adresse: https://www.maxmind.com/en/geoip-databases.

[41] ipgeolocation, *IP Geolocation API*. besucht am 23. Okt. 2024. Adresse: https://ipgeolocation.io/documentation/ip-geolocation-api.html.

[42] Geolocation.com, *Geolocate the Location of an IP Address - Geolocation*. besucht am 17. Jan. 2025. Adresse: https://www.geolocation.com/.

[43] proinity LLC, *IP Location Finder*. besucht am 23. Okt. 2024. Adresse: https://tools.keycdn.com/geo.

[44] G. F. Lyon, *Nmap Network Scanning: Official Nmap Project Guide to Network Discovery and Security Scanning*. Insecure.Com, LLC, 2008, ISBN: 9780979958717. Adresse: https://nmap.org/book.

[45] E. Bou-Harb, M. Debbabi und C. Assi, "A Statistical Approach for Fingerprinting Probing Activities," in *2013 International Conference on Availability, Reliability and Security*, 2013, S. 21–30. DOI: 10.1109/ARES.2013.9.

[46] E. Bou-Harb, "Approaches and Techniques for Fingerprinting and Attributing Probing Activities by Observing Network Telescopes," Submitted, Diss., Concordia University, 2015. Adresse: https://spectrum.library.concordia.ca/id/eprint/980132/.

[47] psatyavavk6, *What is Banner Grabbing?* 17. Aug. 2022. besucht am 23. Okt. 2024. Adresse: https://www.geeksforgeeks.org/what-is-banner-grabbing/.

[48] R. Bodenheim, J. Butts, S. Dunlap und B. Mullins, "Evaluation of the ability of the Shodan search engine to identify Internet-facing industrial control devices," *International Journal of Critical Infrastructure Protection*, Jg. 7, Nr. 2, S. 114–123, 2014, ISSN: 1874-5482. DOI: https://doi.org/10.1016/j.ijcip.2014.03.001. Adresse: https://www.sciencedirect.com/science/article/pii/S1874548214000213.

[49] R. D. Graham et al., *MASSCAN: Mass IP port scanner*, 16. Nov. 2023. besucht am 23. Okt. 2024. Adresse: https://github.com/robertdavidgraham/masscan.

[50] U. Haque, *Thingful (2014)*, 5. Juni 2024. besucht am 23. Okt. 2024. Adresse: https://haque.co.uk/work/thingful/.

[51] QOMPLX, *Punkspider*. besucht am 23. Okt. 2024. Adresse: https://punkspider.org/wtv.html.

[52] QOMPLX, *Punkspider - Contact*. besucht am 23. Okt. 2024. Adresse: https://punkspider.org/contact.html.

[53] KnownSec Hong Kong, *ZoomEye - About Us*. besucht am 23. Okt. 2024. Adresse: https://www.zoomeye.hk/about.

[54] Tenable, Inc, *Tenable Nessus*. besucht am 23. Okt. 2024. Adresse: https://www.tenable.com/products/nessus.

[55] OffSec Services Limited, *skipfish - Kali Linux Tools*, 23. Mai 2024. besucht am 23. Okt. 2024. Adresse: https://www.kali.org/tools/skipfish/.
[56] Acunetix, by Invicti, *Tenable Nessus*. besucht am 23. Okt. 2024. Adresse: https://www.acunetix.com/product/.
[57] IVRE, *IVRE - About IVRE*. besucht am 23. Okt. 2024. Adresse: https://ivre.rocks/#about.
[58] Vulners Inc, *Vulners.com - Vulnerability Scanner Dashboard*. besucht am 23. Okt. 2024. Adresse: https://vulners.com/scanner.
[59] Subgraph, *Vega Vulnerability Scanner*. besucht am 23. Okt. 2024. Adresse: https://subgraph.com/vega/.
[60] Greenbone AG, *OpenVAS - Open Vulnerability Assessment Scanner*. besucht am 23. Okt. 2024. Adresse: https://openvas.org/.
[61] G. Combs et al., *Wireshark README*, 12. Aug. 2024. besucht am 25. Okt. 2024. Adresse: https://gitlab.com/wireshark/wireshark/-/blob/master/README.md.
[62] C. Sullo und D. Lodge, *Nikto 2.5*. besucht am 23. Okt. 2024. Adresse: https://cirt.net/Nikto2.
[63] A. Keks, *Angry IP Scanner*. besucht am 23. Okt. 2024. Adresse: https://angryip.org/.
[64] Famatech Corp., *Advanced IP Scanner*. besucht am 23. Okt. 2024. Adresse: https://www.advanced-ip-scanner.com/.
[65] Rapid7, *Nexpose Vulnerability Scanner*. besucht am 23. Okt. 2024. Adresse: https://www.rapid7.com/products/nexpose/.
[66] A. Foster, T. Kelley, G. Willcox, B. Cook, C. Condon und J. Lee, *Metasploit Documentation - Getting started*, 21. Sep. 2023. besucht am 23. Okt. 2024. Adresse: https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html.
[67] H. Kathpal, *Notice of End of Life (EOL) for Qualys FreeScan*, 7. Mai 2021. besucht am 23. Okt. 2024. Adresse: https://notifications.qualys.com/product/2021/05/07/notice-of-end-of-life-eol-for-qualys-freescan.
[68] Qualys, Inc, *Qualys Community Edition*. besucht am 23. Okt. 2024. Adresse: https://www.qualys.com/community-edition/.
[69] Subgraph, *Vega GitHub Repository*. besucht am 23. Okt. 2024. Adresse: https://github.com/subgraph/Vega.
[70] J. M. Pittman, *Machine Learning and Port Scans: A Systematic Review*, 2023. arXiv: 2301.13581 [cs.CR]. Adresse: https://arxiv.org/abs/2301.13581.
[71] Internet Assigned Numbers Authority, *Number Resources*. besucht am 23. Okt. 2024. Adresse: https://www.iana.org/numbers.
[72] RIPE, *RIPE Database (Whois)*. besucht am 23. Okt. 2024. Adresse: https://stat.ripe.net/widget/whois.
[73] APNIC, *Searching the Whois database*. besucht am 23. Okt. 2024. Adresse: https://www.apnic.net/manage-ip/using-whois/searching/.
[74] Whois.com, *Whois Domain Lookup*. besucht am 23. Okt. 2024. Adresse: https://www.whois.com/whois/.
[75] K. Elliott, "The Who, What, Where, When, and Why of WHOIS: Privacy and Accuracy Concerns of the WHOIS Database," *SMU Science and Technology Law Review*, Jg. 12, Nr. 2, 2009. besucht am 17. Jan. 2025. Adresse: https://scholar.smu.edu/scitech/vol12/iss2/4/.
[76] united-domains GmbH, *Whois Domain Privacy - Anonyme Domain-Registrierung*. besucht am 17. Jan. 2025. Adresse: https://www.united-domains.de/whois-domain-privacy/.
[77] Hostpoint AG, *Whois Domain-Abfrage*. besucht am 17. Jan. 2025. Adresse: https://www.hostpoint.ch/domains/whois.html.
[78] Wikipedia contributors, *Domain privacy — Wikipedia, The Free Encyclopedia*, 2024. besucht am 17. Jan. 2025. Adresse: https://en.wikipedia.org/w/index.php?title=Domain_privacy&oldid=1250142564.

[79] D. Sangvikar, C. Navarrete, M. Tennis, Y. Jia, Y. Fu und S. Shibiraj, *Cobalt Strike Analysis and Tutorial: Identifying Beacon Team Servers in the Wild*, 3. Nov. 2022. besucht am 23. Okt. 2024. Adresse: https://unit42.paloaltonetworks.com/cobalt-strike-team-server/.

[80] RIPE, *RIPE Atlas*. besucht am 23. Okt. 2024. Adresse: https://atlas.ripe.net/.

[81] Censys, *Censys Internet Scanning Introduction*, 25. Juli 2024. besucht am 23. Okt. 2024. Adresse: https://support.censys.io/hc/en-us/articles/25692846962708-Censys-Internet-Scanning-Introduction.

[82] M. Patton, E. Gross, R. Chinn, S. Forbis, L. Walker und H. Chen, "Uninvited Connections: A Study of Vulnerable Devices on the Internet of Things (IoT)," in *2014 IEEE Joint Intelligence and Security Informatics Conference*, 2014, S. 232–235. DOI: 10.1109/JISIC.2014.43.

[83] D. J. D. Touch, *Updated Specification of the IPv4 ID Field*, RFC 6864, Feb. 2013. DOI: 10.17487/RFC6864. Adresse: https://www.rfc-editor.org/info/rfc6864.

[84] Z. Durumeric, D. Roethlisberger, P. Stephens, D. Adrian, A. Holland et al., *zmap/src/probe_modules/packet.c*, 25. Apr. 2024. besucht am 18. Jan. 2025. Adresse: https://github.com/zmap/zmap/blob/main/src/probe_modules/packet.c.

[85] Center for Applied Internet Data Analysis, *Archipelago (Ark) Measurement Infrastructure*, 7. Mai 2024. besucht am 23. Okt. 2024. Adresse: https://www.caida.org/projects/ark/.

[86] Regents of the University of California, *CAIDA Resource Catalog*. besucht am 23. Okt. 2024. Adresse: https://catalog.caida.org/.

[87] Stanford University, *Stanford Internet Research Data Repository*. besucht am 23. Okt. 2024. Adresse: https://scans.io/.

[88] openresolver.com, *Open recursive DNS resolver test*. besucht am 23. Okt. 2024. Adresse: https://www.openresolver.com/.

[89] N. J. Rubenking, *Are You a Zombie? How to Check for Open DNS Resolvers*, 8. Apr. 2013. besucht am 23. Okt. 2024. Adresse: https://www.pcmag.com/news/are-you-a-zombie-how-to-check-for-open-dns-resolvers.

[90] The Shadowserver Foundation, *MEDIUM: DNS Open Resolvers Report*, 8. Dez. 2023. besucht am 23. Okt. 2024. Adresse: https://www.shadowserver.org/what-we-do/network-reporting/dns-open-resolvers-report/.

[91] *Internet Census 2012*, 2012. besucht am 23. Okt. 2024. Adresse: https://internetcensus2012.github.io/InternetCensus2012/paper.html.

[92] J. Kirsch, C. Grothoff, M. Ermert, J. Appelbaum, L. Poitras und H. Moltke, *NSA/GCHQ: Das HACIENDA-Programm zur Kolonisierung des Internet*, 15. Aug. 2014. besucht am 23. Okt. 2024. Adresse: https://www.heise.de/hintergrund/NSA-GCHQ-Das-HACIENDA-Programm-2292574.html.

[93] ShadowWhisperer, *Various lists to be used with an IP blocker*. besucht am 30. Nov. 2024. Adresse: https://github.com/ShadowWhisperer/IPs.

[94] abuse.ch, *ThreatFox Export IOCs*. besucht am 30. Nov. 2024. Adresse: https://threatfox.abuse.ch/export/.

[95] ONYPHE, *About Us*. besucht am 8. Jan. 2025. Adresse: https://www.onyphe.io/about.

[96] E. Foudil und Y. Shafranovich, *A File Format to Aid in Security Vulnerability Disclosure*, RFC 9116, Apr. 2022. DOI: 10.17487/RFC9116. Adresse: https://www.rfc-editor.org/info/rfc9116.

[97] Internet Assigned Numbers Authority, *Well-Known URIs*, 22. Okt. 2024. besucht am 23. Okt. 2024. Adresse: https://www.iana.org/assignments/well-known-uris/well-known-uris.xhtml.

[98] S. Radesky und S. Kennelley, *security.txt: A Simple File with Big Value*, 20. Dez. 2023. besucht am 23. Okt. 2024. Adresse: https://www.cisa.gov/news-events/news/securitytxt-simple-file-big-value.

[99] B. Krebs, *Does Your Organization Have a Security.txt File?* 20. Sep. 2021. besucht am 23. Okt. 2024. Adresse: https://krebsonsecurity.com/2021/09/does-your-organization-have-a-security-txt-file/.

[100] K. Paine, O. Whitehouse, J. Sellwood und A. S, *Indicators of Compromise (IoCs) and Their Role in Attack Defence*, RFC 9424, Aug. 2023. DOI: 10.17487/RFC9424. Adresse: https://www.rfc-editor.org/info/rfc9424.

[101] Censys, *Opt Out of Data Collection*. besucht am 29. Nov. 2024. Adresse: https://support.censys.io/hc/en-us/articles/360043177092-Opt-Out-of-Data-Collection.

[102] Hacker Target Pty Ltd, *Tor Exit Nodes Located and Mapped*. besucht am 25. Okt. 2024. Adresse: https://hackertarget.com/tor-exit-node-visualization/.

[103] The Tor Project Inc., *Tor Project, About Tor*. besucht am 25. Okt. 2024. Adresse: https://support.torproject.org/about/.

[104] Enkidu-6, *Tor-Relay-Lists, Updated lists of Tor Relays*. besucht am 25. Okt. 2024. Adresse: https://enkidu-6.github.io/tor-relay-lists/.

[105] The Tor Project Inc., *Relay Search*. besucht am 25. Okt. 2024. Adresse: https://metrics.torproject.org/rs.html.

[106] J. Althouse, *JA4+ Network Fingerprinting*, 14. Okt. 2024. besucht am 25. Okt. 2024. Adresse: https://github.com/FoxIO-LLC/ja4/blob/main/README.md.

[107] Arkime, *Arkime FAQ*. besucht am 25. Okt. 2024. Adresse: https://arkime.com/faq.

[108] J. Althouse, T. Noel, D. Perry, G. Lipsky und D. Roethlisberger, *JA4+ Plugin for Wireshark*, 24. Sep. 2024. besucht am 25. Okt. 2024. Adresse: https://github.com/FoxIO-LLC/ja4/tree/main/wireshark.

[109] J. Althouse, T. Noel und G. Lipsky, *JA4TScan*, 24. Apr. 2024. besucht am 25. Okt. 2024. Adresse: https://github.com/FoxIO-LLC/ja4tscan/blob/main/README.md.

[110] FoxIO, *JA4+ Database*. besucht am 25. Okt. 2024. Adresse: https://ja4db.com/.

[111] J. Song, Y. Kim und Y. Won, "Operating System Fingerprint Recognition Using ICMP," in *Advances in Computer Science and Ubiquitous Computing*, J. J. Park, D.-S. Park, Y.-S. Jeong und Y. Pan, Hrsg., Singapore: Springer Singapore, 2020, S. 285–290, ISBN: 978-981-13-9341-9.

[112] P. Lalet, V. Ruello, D. Salmon, F. Monjalet et al., *IVRE*, 22. Feb. 2023. besucht am 26. Okt. 2024. Adresse: https://github.com/ivre/ivre/blob/master/README.md.

[113] S. Grover, M. Pierce et al., *Malcolm*, 10. Sep. 2024. besucht am 25. Okt. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/README.md.

[114] Idaho National Laboratory, *Malcolm*. besucht am 26. Okt. 2024. Adresse: https://inl.gov/national-security/ics-malcolm/.

[115] S. Grover und M. Pierce, *Malcolm*, 22. Okt. 2024. besucht am 25. Okt. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/README.md.

[116] S. Grover, *Custom Rules, Scripts and Plugins*, 11. Sep. 2024. besucht am 26. Okt. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/custom-rules.md.

[117] The Zeek Project, *Introduction to Scripting*. besucht am 26. Okt. 2024. Adresse: https://docs.zeek.org/en/master/scripting/index.html.

[118] Arkime, *Rules Format*. besucht am 26. Okt. 2024. Adresse: https://arkime.com/rulesformat.

[119] A. Wick, E. Rinne, T. Salusky und O. McGill, *Simple lua integration*, 11. Sep. 2024. besucht am 26. Okt. 2024. Adresse: https://github.com/arkime/arkime/tree/main/capture/plugins/lua.

[120] S. Grover, *Arkime*, 12. Sep. 2024. besucht am 26. Okt. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/arkime.md.

[121] S. Grover, M. Pierce, J. Arscott, B. Allen, C. Clauss, E. Schaller et al., *Malcolm*, 24. Okt. 2024. besucht am 26. Okt. 2024. Adresse: https://github.com/cisagov/Malcolm/tree/main.

[122] S. Grover, *Network Diagram*, 24. Juni 2020. besucht am 26. Okt. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/images/malcolm_network_diagram.odg.

[123] D. Perry, C. Craft, G. Harris, G. Combs und U. Lamping, *Packet capture library (libpcap)*, 26. Okt. 2020. besucht am 25. Okt. 2024. Adresse: https://wiki.wireshark.org/libpcap.

[124] A. Wick, E. Rinne, T. Salusky und O. McGill, *arkime/capture/reader-libpcap.c*, 11. Sep. 2024. besucht am 25. Okt. 2024. Adresse: https://github.com/arkime/arkime/blob/main/capture/reader-libpcap.c.

[125] L. F. Sikos, "Packet analysis for network forensics: A comprehensive," *Forensic Science International: Digital Investigation*, Jg. 32, S. 200 892, 2020, ISSN: 2666-2817. DOI: https://doi.org/10.1016/j.fsidi.2019.200892. besucht am 8. Juni 2024. Adresse: https://www.sciencedirect.com/science/article/pii/S1742287619302002.

[126] S. Grover, *Malcolm Components*, 18. Sep. 2023. besucht am 1. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/images/malcolm_components.png.

[127] J. A. Donenfeld, *WireGuard: Next Generation Kernel Network Tunnel*, 1. Juni 2020. besucht am 30. Okt. 2024. Adresse: https://www.wireguard.com/papers/wireguard.pdf.

[128] J. McGinty, B. Schwind, C. Maclennan, F. Affolter, R. Kawaguchi, M. Laitl et al., *innernet*, 13. Sep. 2024. besucht am 30. Okt. 2024. Adresse: https://github.com/tonarino/innernet/blob/main/README.md.

[129] C. Chee, N. Stuckenbrock, S.-H. Haase, M. Ehlers, B. Yüksel, A. Budantsov et al., *Awesome WireGuard*, 4. Sep. 2024. besucht am 30. Okt. 2024. Adresse: https://github.com/cedrickchee/awesome-wireguard/blob/main/README.md.

[130] ashirkar, *Understanding SPAN,RSPAN,and ERSPAN*, 3. Jan. 2019. besucht am 30. Okt. 2024. Adresse: https://community.cisco.com/t5/networking-knowledge-base/understanding-span-rspan-and-erspan/ta-p/3144951.

[131] M. Foschiano, K. Ghosh und M. Mehta, "Cisco Systems' Encapsulated Remote Switch Port Analyzer (ERSPAN)," Internet Engineering Task Force, Internet-Draft draft-foschiano-erspan-03, Feb. 2017, Work in Progress, 16 S. Adresse: https://datatracker.ietf.org/doc/draft-foschiano-erspan/03/.

[132] S. Hemminger et al., *tc(8) Linux manual page*, 8. Mai 2024. besucht am 1. Nov. 2024. Adresse: https://git.kernel.org/pub/scm/network/iproute2/iproute2.git/tree/man/man8/tc.8.

[133] S. Hemminger et al., *ip(8) Linux manual page*, 26. Feb. 2024. besucht am 1. Nov. 2024. Adresse: https://git.kernel.org/pub/scm/network/iproute2/iproute2.git/tree/man/man8/ip.8.

[134] xose, t richards, willsewell, bluca, jnguyen und MdRayhan, *iproute2 README*, 4. Nov. 2023. besucht am 1. Nov. 2024. Adresse: https://wiki.linuxfoundation.org/networking/iproute2.

[135] B. Siach, G. Espinasse, Q. Monnet und S. Hemminger, *iproute2 README*, 13. März 2024. besucht am 1. Nov. 2024. Adresse: https://git.kernel.org/pub/scm/network/iproute2/iproute2.git/tree/README.

[136] The kernel development community, *Family tc netlink specification*, 4. Nov. 2023. besucht am 1. Nov. 2024. Adresse: https://www.kernel.org/doc/html/latest/networking/netlink_spec/tc.html.

[137] D. Waiting, *Traffic Mirroring with Linux Tc*, 24. Mai 2020. besucht am 1. Nov. 2024. Adresse: https://medium.com/swlh/traffic-mirroring-with-linux-tc-df4d36116119.

[138] H. Liu, *An introduction to Linux virtual interfaces: Tunnels, ERSPAN and IP6ERSPAN*, 17. Mai 2019. besucht am 1. Nov. 2024. Adresse: https://developers.redhat.com/blog/2019/05/17/an-introduction-to-linux-virtual-interfaces-tunnels#erspan_and_ip6erspan.

[139] W. Tu und G. Rose, *ERSPAN in Linux, A short history and review*, 11. Juni 2018. besucht am 1. Nov. 2024. Adresse: http://oldvger.kernel.org/lpc_net2018_talks/erspan-linux-presentation.pdf.

[140] OpenSearch contributors, *Introduction to OpenSearch*. besucht am 2. Nov. 2024. Adresse: https://opensearch.org/docs/latest/getting-started/intro/.

[141] S. Grover, *Live analysis*, 12. Sep. 2024. besucht am 26. Okt. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/live-analysis.md.
[142] S. Grover und M. Pierce, *install.py*, 21. Okt. 2024. besucht am 2. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/scripts/install.py.
[143] S. Grover et al., *Malcolm Components*, 12. Sep. 2024. besucht am 2. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/hedgehog.md.
[144] P. Wise, T. Lange, I. Kelling, M. Ducorps et al., *Automating Linux Installations*, 8. Juni 2020. besucht am 9. Nov. 2024. Adresse: https://wiki.debian.org/AutomatedInstallation.
[145] Debian Installer team, *Debian GNU/Linux Installation Guide for 64-bit PC (amd64)*, 8. Mai 2023. besucht am 9. Nov. 2024. Adresse: https://www.debian.org/releases/stable/amd64/.
[146] R. Hertzog und R. Mas, *The Debian Administrator's Handbook*, 2022. besucht am 9. Nov. 2024. Adresse: https://debian-handbook.info/browse/stable/index.html.
[147] Ansible project contributors, *Introduction to Ansible*, 12. Nov. 2024. besucht am 13. Nov. 2024. Adresse: https://docs.ansible.com/ansible/latest/getting_started/introduction.html.
[148] J. Strandboge, W. Garcia-Fontes, A. B. Pérez, J. Conder et al., *ufw*. besucht am 15. Nov. 2024. Adresse: https://launchpad.net/ufw.
[149] S. Grover, *Hardening*, 22. Okt. 2024. besucht am 15. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/hardening.md.
[150] S. Grover, *Hardening*, 22. Okt. 2024. besucht am 15. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/hedgehog-hardening.md.
[151] P. Biondi, *Scapy Introduction*. besucht am 29. Nov. 2024. Adresse: https://scapy.readthedocs.io/en/latest/introduction.html.
[152] S. Grover, *Malcolm Components*, 11. Juni 2019. besucht am 2. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/images/logo/Malcolm.svg.
[153] S. Grover und M. Pierce, *End-to-end Malcolm and Hedgehog Linux ISO Installation*, 12. Sep. 2024. besucht am 2. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/malcolm-hedgehog-e2e-iso-install.md.
[154] S. Hemminger et al., *ip-link(8) Linux manual page*, 26. Feb. 2024. besucht am 6. Nov. 2024. Adresse: https://git.kernel.org/pub/scm/network/iproute2/iproute2.git/tree/man/man8/ip-link.8.in.
[155] Red Hat, Inc., *What is the maximum number of interface aliases supported in Red Hat Enterprise Linux?* 7. Aug. 2024. besucht am 30. Nov. 2024. Adresse: https://access.redhat.com/solutions/40500.
[156] V. Jacobson, C. Leres, S. McCanne et al., *tcpdump(1) man page*, 7. Sep. 2024. besucht am 6. Nov. 2024. Adresse: https://www.tcpdump.org/manpages/tcpdump.1.html.
[157] P. Sutter, S. Ladkani, S. Hemminger, V. Nogueira, L. Boccassi und A. Claudi, *tc-mirred(8) Linux manual page*, 8. Mai 2024. besucht am 1. Nov. 2024. Adresse: https://git.kernel.org/pub/scm/network/iproute2/iproute2.git/plain/man/man8/tc-mirred.8.
[158] E. Rocca, H. Wansing, P. Wise, F. Stupp, B. Potkin et al., *DebianInstaller - Preseed - EditIso*, 16. Nov. 2023. besucht am 9. Nov. 2024. Adresse: https://wiki.debian.org/DebianInstaller/Preseed/EditIso.
[159] *An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware*, RFC 826, Nov. 1982. DOI: 10.17487/RFC0826. Adresse: https://www.rfc-editor.org/info/rfc826.
[160] Canonical Ltd., *ufw (8) man page*, 18. Mai 2023. besucht am 15. Nov. 2024. Adresse: https://manpages.debian.org/bookworm/ufw/ufw.8.en.html.
[161] C. Hornig, *A Standard for the Transmission of IP Datagrams over Ethernet Networks*, RFC 894, Apr. 1984. DOI: 10.17487/RFC0894. Adresse: https://www.rfc-editor.org/info/rfc894.
[162] D. S. E. Deering und B. Hinden, *Internet Protocol, Version 6 (IPv6) Specification*, RFC 8200, Juli 2017. DOI: 10.17487/RFC8200. Adresse: https://www.rfc-editor.org/info/rfc8200.

[163] The Zeek Project, *Quick Start Guide*, 15. Nov. 2024. besucht am 20. Nov. 2024. Adresse: https://docs.zeek.org/en/master/quickstart.html.
[164] The Zeek Project, *Introduction to Scripting, The Basics*. besucht am 20. Nov. 2024. Adresse: https://docs.zeek.org/en/master/scripting/basics.html.
[165] The Zeek Project, *Script Reference*. besucht am 23. Nov. 2024. Adresse: https://docs.zeek.org/en/master/script-reference/index.html.
[166] D. Crawford, Jan et al., *ActiveHTTP*, Jan. 2017. besucht am 23. Nov. 2024. Adresse: https://community.zeek.org/t/activehttp/4628/6.
[167] B. Moss und J. Siwek, *ActiveHTTP*, Juni 2019. besucht am 23. Nov. 2024. Adresse: https://community.zeek.org/t/activehttp-module-error/5721.
[168] Internet Assigned Numbers Authority, *Hypertext Transfer Protocol (HTTP) Status Code Registry*, 13. Nov. 2024. besucht am 23. Nov. 2024. Adresse: https://www.iana.org/assignments/http-status-codes/http-status-codes.xhtml.
[169] Internet Assigned Numbers Authority, *Internet Protocol Version 6 (IPv6) Parameters*, 31. Okt. 2024. besucht am 23. Nov. 2024. Adresse: https://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xhtml.
[170] D. B. Johnson, J. Arkko und C. E. Perkins, *Mobility Support in IPv6*, RFC 6275, Juli 2011. DOI: 10.17487/RFC6275. Adresse: https://www.rfc-editor.org/info/rfc6275.
[171] The Zeek Project, *Input Framework*. besucht am 23. Nov. 2024. Adresse: https://docs.zeek.org/en/master/frameworks/input.html.
[172] RIPE, *RIPE Atlas Docs, Query Parameters for Probes*, 31. Mai 2024. besucht am 30. Nov. 2024. Adresse: https://atlas.ripe.net/docs/apis/rest-api-manual/probes/queryparameters.html.
[173] RIPE, *RIPE Atlas API Probe List*. besucht am 30. Nov. 2024. Adresse: https://atlas.ripe.net/api/v2/probes/.
[174] The Tor Project, *Tor Bulk Exit List exporting tool*. besucht am 6. Dez. 2024. Adresse: https://check.torproject.org/api/bulk.
[175] J. Elfström und P. Mortensen, *How to delete duplicate lines in a file without sorting it in Unix*, 28. Okt. 2021. besucht am 6. Dez. 2024. Adresse: https://stackoverflow.com/a/1444448.
[176] lind und J. Richards, *Save modifications in place with awk*, 7. Aug. 2020. besucht am 30. Nov. 2024. Adresse: https://stackoverflow.com/a/16531920.
[177] The Zeek Project, *capture_loss.log and reporter.log*, 13. Dez. 2024. besucht am 18. Dez. 2024. Adresse: https://docs.zeek.org/en/master/logs/capture-loss-and-reporter.html.
[178] Sharashka Inc., *internet-measurement.com*. besucht am 18. Dez. 2024. Adresse: https://www.internet-measurement.com/.
[179] Internet Census Group, *Internet Census Group*. besucht am 18. Dez. 2024. Adresse: https://www.internet-census.org.
[180] Whois.com, *Whois IP 167.94.138.138*, 29. März 2024. besucht am 18. Dez. 2024. Adresse: https://www.whois.com/whois/167.94.138.138.
[181] Whois.com, *Whois IP 199.45.155.100*, 29. März 2024. besucht am 18. Dez. 2024. Adresse: https://www.whois.com/whois/199.45.155.100.
[182] Whois.com, *Whois censys-scanner.com*, 30. Apr. 2024. besucht am 18. Dez. 2024. Adresse: https://www.whois.com/whois/censys-scanner.com.
[183] research-scanner.com, *Research Scanner*. besucht am 18. Dez. 2024. Adresse: http://research-scanner.com/.
[184] netsecscan.net, *scanning research*. besucht am 18. Dez. 2024. Adresse: http://netsecscan.net/.
[185] The Recyber Project, *The Recyber Project*. besucht am 18. Dez. 2024. Adresse: https://www.recyber.net/.
[186] CyberResilience.io, *CyberResilience*. besucht am 18. Dez. 2024. Adresse: https://cyberresilience.io/cyberresilience.

[187] ONYPHE, *Why we won't scan /0 for log4shell issue*, 14. Dez. 2021. besucht am 18. Dez. 2024. Adresse: https://www.onyphe.io/docs/write-ups/why-we-won-t-scan--0-for-log4shell-issue.
[188] ONYPHE, *ONYPHE Query Language (OQL)*. besucht am 18. Dez. 2024. Adresse: https://www.onyphe.io/docs/onyphe-query-language.
[189] ANT Lab, *about ANT*. besucht am 18. Dez. 2024. Adresse: https://ant.isi.edu/about/index.html.
[190] CyberOK, *Abuse (responsible scanning) Policy*, 10. Nov. 2024. besucht am 18. Dez. 2024. Adresse: https://www.cyberok.ru/docs/cyberok_scan_policy_en.pdf.
[191] Stretchoid, *Stretchoid Opt-Out*. besucht am 30. Dez. 2024. Adresse: https://stretchoid.com/.
[192] LeakIX, *About LeakIX*. besucht am 30. Dez. 2024. Adresse: https://leakix.net/about.
[193] Google for Developers, *Googlebot*, 26. Nov. 2024. besucht am 30. Dez. 2024. Adresse: https://developers.google.com/search/docs/crawling-indexing/googlebot?visit_id=638711619625940695-4045913395&rd=1.
[194] Department of Computer Science and Technology, University of Cambridge, *Cambridge Cybercrime Centre Internet scanner*. besucht am 30. Dez. 2024. Adresse: https://cccc-scanner.dtg.cl.cam.ac.uk/.
[195] BinaryEdge, *Who is BinaryEdge?* Besucht am 18. Dez. 2024. Adresse: https://www.binaryedge.io/about.html.
[196] S. Erb, *Quickly find certificates in IPv4 space*. besucht am 18. Dez. 2024. Adresse: https://tls.bufferover.run/.
[197] S. Grover und M. Pierce, *Malcolm Components*, 12. Sep. 2024. besucht am 2. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/components.md.
[198] S. Grover und M. Pierce, *OpenSearch Dashboards*, 25. Juni 2024. besucht am 6. Dez. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/dashboards.md.
[199] The Zeek Project, *base/init-bare.zeek, udp_inactivity_timeout*. besucht am 5. März 2025. Adresse: https://docs.zeek.org/en/master/scripts/base/init-bare.zeek.html#id-udp_inactivity_timeout.
[200] Philippe Biondi and the Scapy community, *Scapy Documentation*, 23. Nov. 2024. besucht am 23. Nov. 2024. Adresse: https://scapy.readthedocs.io/en/latest/index.html.
[201] O. Eggert, *IPv6 Packet Creation With Scapy Documentation*, 19. Jan. 2012. besucht am 23. Nov. 2024. Adresse: https://www.idsv6.de/Downloads/IPv6PacketCreationWithScapy.pdf.
[202] A. Atlasis, *IPv6 Extension Headers - New Features, and New Attack Vectors*, 3. Jan. 2013. besucht am 23. Nov. 2024. Adresse: https://troopers.de/wp-content/uploads/2013/01/TROOPERS13-IPv6_Extension_Headers_New_Features_and_New_Attack_Vectors-Antonios_Atlasis.pdf.
[203] Arkime, *Arkime Settings, Plugins - JA4+*. besucht am 25. Okt. 2024. Adresse: https://arkime.com/settings#ja4plus.
[204] J. Althouse, M. Pierce, P. Lalet, D. Roethlisberger et al., *JA4+ for Zeek*, 15. Okt. 2024. besucht am 6. Dez. 2024. Adresse: https://github.com/FoxIO-LLC/ja4/tree/main/zeek.
[205] Alibaba Cloud, *Elastic Compute Service*. besucht am 13. Dez. 2024. Adresse: https://www.alibabacloud.com/product/ecs.
[206] Amazon Web Services, Inc. or its affiliates, *Amazon EC2*. besucht am 10. Dez. 2024. Adresse: https://aws.amazon.com/ec2/.
[207] Hetzner Online GmbH, *Truly thrifty cloud hosting - Hetzner Online GmbH*. besucht am 10. Dez. 2024. Adresse: https://www.hetzner.com/cloud/.
[208] Microsoft, *Azure Virtual Machines*. besucht am 10. Dez. 2024. Adresse: https://azure.microsoft.com/en-us/products/virtual-machines/.
[209] justhost.ru, *Virtual hosting plans*. besucht am 14. Dez. 2024. Adresse: https://justhost.ru/en.

[210] Serverspace, *How to Configure IPv6 on a VPS*. besucht am 13. Dez. 2024. Adresse: https://serverspace.io/services/cloud-servers/.
[211] UltaHost, *VPS Hosting*. besucht am 10. Dez. 2024. Adresse: https://ultahost.com/vps-hosting.
[212] VPS.us, *Simple VPS Hosting Anywhere*. besucht am 11. Dez. 2024. Adresse: https://vps.us/kvm-vps/.
[213] Kamatera Inc., *VPSServer.com*. besucht am 18. Dez. 2024. Adresse: https://www.vpsserver.com/cloud-server/.
[214] HOSTAFRICA, *Linux VPS Hosting - HOSTAFRICA*. besucht am 11. Dez. 2024. Adresse: https://www.hostafrica.ng/servers/virtual-server/.
[215] V. Ksinant, C. Huitema, D. S. Thomson und M. Souissi, *DNS Extensions to Support IP Version 6*, RFC 3596, Okt. 2003. DOI: 10.17487/RFC3596. Adresse: https://www.rfc-editor.org/info/rfc3596.
[216] KeePassXC Team, *KeePassXC Password Manager*. besucht am 17. Jan. 2025. Adresse: https://keepassxc.org/.
[217] K. Adamu, V. Lalwani, Fezoj und Thushara, *Visualize data using JSON input in Metrics/Advanced*, 14. Okt. 2024. besucht am 27. Dez. 2024. Adresse: https://forum.opensearch.org/t/visualize-data-using-json-input-in-metrics-advanced/9189.
[218] OpenSearch contributors, *Execute Painless script*. besucht am 27. Dez. 2024. Adresse: https://opensearch.org/docs/latest/api-reference/script-apis/exec-script/.
[219] Elasticsearch B.V., *A Brief Painless Walkthrough*. besucht am 27. Dez. 2024. Adresse: https://www.elastic.co/guide/en/elasticsearch/painless/8.17/painless-walkthrough.html.
[220] The Zeek Project, *base/frameworks/notice/main.zeek, Notice::Info*. besucht am 3. Jan. 2025. Adresse: https://docs.zeek.org/en/master/scripts/base/frameworks/notice/main.zeek.html#type-Notice::Info.
[221] S. Grover, *standardize locations/sources for GeoIP database #394*, 4. Nov. 2024. besucht am 30. Dez. 2024. Adresse: https://github.com/cisagov/Malcolm/issues/394.
[222] MaxMind, Inc., *GeoLite Databases and Web Services*. besucht am 31. Jan. 2025. Adresse: https://dev.maxmind.com/geoip/geolite2-free-geolocation-data/.
[223] Elasticsearch B.V., *Geoip filter plugin, Supported Databases*, 22. Mai 2024. besucht am 31. Jan. 2025. Adresse: https://www.elastic.co/guide/en/logstash/current/plugins-filters-geoip.html#_supported_databases.
[224] International Organization for Standardization, *ISO 3166 Country Codes*. besucht am 3. Jan. 2025. Adresse: https://www.iso.org/iso-3166-country-codes.html.
[225] International Organization for Standardization, *Online Browsing Platform (OBP)*. besucht am 3. Jan. 2025. Adresse: https://www.iso.org/obp/ui/#search/code/.
[226] The Zeek Project, *Zeek FAQ*. besucht am 18. Dez. 2024. Adresse: https://zeek.org/faq/.
[227] P. Machata et al., *IP Sysctl, IP Variables*, 10. Dez. 2024. besucht am 10. Jan. 2025. Adresse: https://www.kernel.org/doc/html/latest/networking/ip-sysctl.html#ip-variables.
[228] *Domain names - concepts and facilities*, RFC 1034, Nov. 1987. DOI: 10.17487/RFC1034. Adresse: https://www.rfc-editor.org/info/rfc1034.
[229] WHOIS API, *DNS Chronicle API*. besucht am 22. Jan. 2025. Adresse: https://dns-history.whoisxmlapi.com/api/documentation/making-requests.
[230] Silent Push Inc., *Scan through passive DNS data*, 2. Mai 2023. besucht am 22. Jan. 2025. Adresse: https://help.silentpush.com/docs/scanning-through-passive-dns-data.
[231] Silent Push Inc., *Explore Indicator DNS Data*. besucht am 22. Jan. 2025. Adresse: https://explore.silentpush.com/explore.

[232] Silent Push Inc., *Search through passive DNS data (forward lookup)*, 16. Mai 2023. besucht am 22. Jan. 2025. Adresse: https://help.silentpush.com/v1/docs/perform-a-forward-lookup-of-passive-dns-data.
[233] Debian System Administration (DSA), *Debian mirrors backed by Fastly CDN*. besucht am 22. Jan. 2025. Adresse: http://deb.debian.org/.
[234] Hurricane Electric, *AS16509 Amazon.com, Inc.* besucht am 22. Jan. 2025. Adresse: https://bgp.he.net/AS16509.
[235] Hurricane Electric, *AS54113 Fastly, Inc.* besucht am 22. Jan. 2025. Adresse: https://bgp.he.net/AS54113.
[236] Internet Assigned Numbers Authority, *Service Name and Transport Protocol Port Number Registry*, 23. Jan. 2025. besucht am 12. Feb. 2025. Adresse: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml.
[237] Whois.com, *Whois IP 168.63.129.16*, 14. Dez. 2021. besucht am 10. Jan. 2025. Adresse: https://www.whois.com/whois/168.63.129.16.
[238] V. Kavuru et al., *cloud-init support for virtual machines in Azure*, 22. Aug. 2024. besucht am 10. Jan. 2025. Adresse: https://learn.microsoft.com/en-us/azure/virtual-machines/linux/using-cloud-init.
[239] M. McInnes et al., *Diving deeper into cloud-init*, 22. Aug. 2024. besucht am 10. Jan. 2025. Adresse: https://learn.microsoft.com/en-us/azure/virtual-machines/linux/cloud-init-deep-dive.
[240] P. Barbé und K. Schouteeten, *So I became a node: exploiting bootstrap tokens in Azure Kubernetes Service*, 23. Apr. 2024. besucht am 10. Jan. 2025. Adresse: https://www.synacktiv.com/publications/so-i-became-a-node-exploiting-bootstrap-tokens-in-azure-kubernetes-service.
[241] S. O. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, RFC 2119, März 1997. DOI: 10.17487/RFC2119. Adresse: https://www.rfc-editor.org/info/rfc2119.
[242] R. Sommer, D. Thayer, J. Siwek und vpax, *zeek/scripts/base/utils/active-http.zeek*. besucht am 10. Jan. 2025. Adresse: https://github.com/zeek/zeek/blob/master/scripts/base/utils/active-http.zeek#L15-L24.
[243] F5, Inc., *nginx*. besucht am 17. Jan. 2025. Adresse: https://nginx.org/.
[244] C. Kuenzler, *Nginx rewrite URL examples with and without redirect address*, 21. Feb. 2022. besucht am 17. Jan. 2025. Adresse: https://www.claudiokuenzler.com/blog/436/nginx-rewrite-url-examples-with-without-redirect-address.
[245] B. Haberman und B. Hinden, *Unique Local IPv6 Unicast Addresses*, RFC 4193, Okt. 2005. DOI: 10.17487/RFC4193. Adresse: https://www.rfc-editor.org/info/rfc4193.
[246] Internet Assigned Numbers Authority, *IANA IPv6 Special-Purpose Address Registry*, 22. Okt. 2024. besucht am 17. Jan. 2025. Adresse: https://www.iana.org/assignments/iana-ipv6-special-registry/iana-ipv6-special-registry.xhtml.
[247] B. E. Carpenter und K. Moore, *Connection of IPv6 Domains via IPv4 Clouds*, RFC 3056, Feb. 2001. DOI: 10.17487/RFC3056. Adresse: https://www.rfc-editor.org/info/rfc3056.
[248] Oracle Corporation and/or its affiliates, *6to4 Automatic Tunnels*, 2010. besucht am 29. Jan. 2025. Adresse: https://docs.oracle.com/cd/E19253-01/816-4554/ipv6-ref-50/index.html.
[249] Sharashka Inc., *Driftnet*. besucht am 19. Feb. 2025. Adresse: https://driftnet.io/.
[250] RIPE, *RIPE Database Lookup Result 2a06:4880::/32*. besucht am 29. Jan. 2025. Adresse: https://apps.db.ripe.net/db-web-ui/lookup?source=ripe&key=2a06:4880::%2F32&type=inet6num.
[251] RIPE Atlas, *Probe #1007148*. besucht am 18. Jan. 2025. Adresse: https://atlas.ripe.net/probes/1007148#tab-network.
[252] Max-Planck-Gesellschaft, *Max-Planck-Gesellschaft*. besucht am 29. Jan. 2025. Adresse: https://www.mpg.de.

[253] RIPE, *RIPE Database*. besucht am 29. Jan. 2025. Adresse: https://apps.db.ripe.net/db-web-ui/query.

[254] J. Zirngibl, L. Steger, P. Sattler, O. Gasser und G. Carle, "Rusty Clusters? Dusting an IPv6 Research Foundation," in *Proceedings of the 2022 Internet Measurement Conference*, Nice, France: ACM, 2022. DOI: 10.1145/3517745.3561440.

[255] L. Steger, L. Kuang, J. Zirngibl, G. Carle und O. Gasser, "Target Acquired? Evaluating Target Generation Algorithms for IPv6," in *Proceedings of the Network Traffic Measurement and Analysis Conference (TMA)*, Naples, Italy, Juni 2023.

[256] The Zeek Project, *conn.log*, 9. Jan. 2025. besucht am 10. Jan. 2025. Adresse: https://docs.zeek.org/en/master/logs/conn.html.

[257] The Zeek Project, *Types, Port*, 9. Jan. 2025. besucht am 10. Jan. 2025. Adresse: https://docs.zeek.org/en/master/script-reference/types.html#type-port.

[258] drb_ra, *ThreatFox database entry for ip:port 122.226.191.252:8443*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/1190455.

[259] abuse.ch, *ThreatFox database entry for ip:port 51.15.19.32:7707*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/159805.

[260] E. Schooler et al., *SIP: Session Initiation Protocol*, RFC 3261, Juli 2002. DOI: 10.17487/RFC3261. Adresse: https://www.rfc-editor.org/info/rfc3261.

[261] E. Rescorla, H. Tschofenig und N. Modadugu, *The Datagram Transport Layer Security (DTLS) Protocol Version 1.3*, RFC 9147, Apr. 2022. DOI: 10.17487/RFC9147. Adresse: https://www.rfc-editor.org/info/rfc9147.

[262] J. Iyengar und M. Thomson, *QUIC: A UDP-Based Multiplexed and Secure Transport*, RFC 9000, Mai 2021. DOI: 10.17487/RFC9000. Adresse: https://www.rfc-editor.org/info/rfc9000.

[263] J. Martin, J. Burbank, W. Kasch und P. D. L. Mills, *Network Time Protocol Version 4: Protocol and Algorithms Specification*, RFC 5905, Juni 2010. DOI: 10.17487/RFC5905. Adresse: https://www.rfc-editor.org/info/rfc5905.

[264] Hurricane Electric, *AS16276 OVH SAS*. besucht am 19. Feb. 2025. Adresse: https://bgp.he.net/AS16276.

[265] OpenSearch contributors, *Reporting using OpenSearch Dashboards*. besucht am 1. Feb. 2025. Adresse: https://opensearch.org/docs/latest/reporting/report-dashboard-index/.

[266] D. Klevebring, *Select objects based on value of variable in object using jq*, 9. Mai 2020. besucht am 1. Feb. 2025. Adresse: https://stackoverflow.com/a/18608100.

[267] Callie J, *Deleting the first two lines of a file using BASH or awk or sed or whatever*, 13. Jan. 2012. besucht am 1. Feb. 2025. Adresse: https://stackoverflow.com/a/8857745.

[268] FoxIO, *Submit a Fingerprint Set*. besucht am 14. Feb. 2025. Adresse: https://docs.ja4db.com/ja4+-database/usage/submit-a-fingerprint-set.

[269] Censys, *Data Definitions*. besucht am 1. Feb. 2025. Adresse: https://search.censys.io/search/definitions?resource=hosts.

[270] J. Althouse und JOeJOh, *ja4/zeek/ja4l/main.zeek*, 9. Apr. 2024. besucht am 1. Feb. 2025. Adresse: https://github.com/FoxIO-LLC/ja4/blob/main/zeek/ja4l/main.zeek.

[271] Individual mozilla.org contributors, *References - HTTP - Headers - Host*. besucht am 14. Feb. 2025. Adresse: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Host.

[272] FreedomHouse, *Expanding and Defending Freedom Around the World*. besucht am 14. Feb. 2025. Adresse: https://freedomhouse.org/.

[273] H. Gowtham, *A complete guide to M3U8 files in HLS streaming*, 12. Jan. 2025. besucht am 14. Feb. 2025. Adresse: https://www.fastpix.io/blog/a-complete-guide-to-m3u8-files-in-hls-streaming.

[274] Lacework Labs, *AndroxGhOst – the python malware exploiting your AWS keys*. besucht am 14. Feb. 2025. Adresse: https://www.lacework.com/blog/androxghost-the-python-malware-exploiting-your-aws-keys.

[275] J. Althouse, *JA4T: TCP Fingerprinting*, 23. Apr. 2024. besucht am 31. Jan. 2025. Adresse: https://blog.foxio.io/ja4t-tcp-fingerprinting.
[276] C. M. Lonvick und T. Ylonen, *The Secure Shell (SSH) Transport Layer Protocol*, RFC 4253, Jan. 2006. DOI: 10.17487/RFC4253. Adresse: https://www.rfc-editor.org/info/rfc4253.
[277] The Go Authors, *Go Cryptography ssh/transport.go*, 18. Dez. 2023. besucht am 29. Jan. 2025. Adresse: https://cs.opensource.google/go/x/crypto/+/master:ssh/transport.go;l=308?q=SSH-2.0-Go.
[278] RIPE, *RIPE Database Search Result 185.60.136.12*. besucht am 29. Jan. 2025. Adresse: https://apps.db.ripe.net/db-web-ui/query?searchtext=185.60.136.12.
[279] Soroush Media Company, *Catalog*, Elemente von Webseite mittels duckduckgo.com übersetzt. besucht am 29. Jan. 2025. Adresse: https://sinet.ir/catalog/.
[280] The Tor Project Inc., *Tor Project, History*. besucht am 10. Jan. 2025. Adresse: https://www.torproject.org/about/history/.
[281] abuse.ch, *ThreatFox*. besucht am 6. Feb. 2025. Adresse: https://threatfox.abuse.ch.
[282] Gi7w0rm, *ThreatFox database entry for ip:port 78.128.114.66:8888*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/1281855.
[283] abuse.ch, *ThreatFox database entry for ip:port 194.180.48.84:59666*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/1127286.
[284] abus3reports, *ThreatFox database entry for ip:port 204.76.203.70:1311*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/1237475.
[285] DonPasci, *ThreatFox database entry for ip:port 154.213.187.164:25000*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/1315447.
[286] abus3reports, *ThreatFox database entry for ip:port 204.76.203.71:1311*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/1237501.
[287] abuse.ch, *ThreatFox database entry for ip:port 194.180.49.190:9254*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/1180440.
[288] abuse.ch, *ThreatFox database entry for ip:port 194.180.48.115:4042*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/916062.
[289] abuse.ch, *ThreatFox database entry for ip:port 194.180.48.113:1190*. besucht am 14. Feb. 2025. Adresse: https://threatfox.abuse.ch/ioc/1146570.
[290] ONYPHE, *Big Data for Cyber Defense*. besucht am 19. Dez. 2025. Adresse: https://search.onyphe.io/.
[291] ONYPHE, *Suchresultat zu "category:datascan ip:92.246.131.45"*. besucht am 8. Jan. 2025. Adresse: https://search.onyphe.io/search?q=category%3Adatascan+ip%3A92.246.131.45.
[292] ONYPHE, *Suchresultat zu "category:datascan ip:46.183.27.161"*. besucht am 8. Jan. 2025. Adresse: https://search.onyphe.io/search?q=category%3Adatascan+ip%3A46.183.27.161.
[293] Shodan, *Suchresultat zu "8.217.233.46"*. besucht am 19. Feb. 2025. Adresse: https://www.shodan.io/search?query=8.217.233.46.
[294] Censys, *Suchresultat zu "8.217.233.46"*. besucht am 19. Feb. 2025. Adresse: https://search.censys.io/hosts/8.217.233.46.
[295] Whois.com, *Whois mapeilin.com*, 7. Mai 2024. besucht am 19. Feb. 2025. Adresse: https://www.whois.com/whois/mapeilin.com.
[296] Cloudflare, Inc., *WHOIS redaction*. besucht am 19. Feb. 2025. Adresse: https://developers.cloudflare.com/registrar/account-options/whois-redaction/.
[297] LeakIX, *LeakIX*. besucht am 8. Jan. 2025. Adresse: https://leakix.net.
[298] LeakIX, *Suchresultat zu " ip:"176.97.192.247" "*, 9. Dez. 2024. besucht am 19. Feb. 2025. Adresse: https://leakix.net/host/176.97.192.247.
[299] LeakIX, *Suchresultat zu " ip:"8.217.233.46" "*, 27. Nov. 2024. besucht am 19. Feb. 2025. Adresse: https://leakix.net/host/8.217.233.46.

[300] LeakIX, *Suchresultat zu " ip:"92.246.131.45" "*, 9. Okt. 2024. besucht am 19. Feb. 2025. Adresse: https://leakix.net/host/92.246.131.45.
[301] LeakIX, *Suchresultat zu " ip:"195.133.11.199" "*, 1. Aug. 2024. besucht am 19. Feb. 2025. Adresse: https://leakix.net/host/195.133.11.199.
[302] WHOIS API, *pinelair.com WHOIS History details*. besucht am 31. Jan. 2025. Adresse: https://whois-history.whoisxmlapi.com/lookup-report/p85Ew8ABRe.
[303] T. Filatovs, D. Timma, M. B., G. G. et al., *API*, 19. Feb. 2025. besucht am 19. Feb. 2025. Adresse: https://help.mikrotik.com/docs/spaces/ROS/pages/47579160/API.
[304] Threat Research Team, *Avast Q1/2022 Threat Report*, 5. Mai 2022. besucht am 19. Feb. 2025. Adresse: https://decoded.avast.io/threatresearch/avast-q1-2022-threat-report/.
[305] M. Hron, *Mēris and TrickBot standing on the shoulders of giants*, 18. März 2022. besucht am 19. Feb. 2025. Adresse: https://decoded.avast.io/martinhron/meris-and-trickbot-standing-on-the-shoulders-of-giants/.
[306] W. Ross, *The TrickBot and MikroTik Connection – A Story of Investment and Collaboration*, 12. Dez. 2018. besucht am 19. Feb. 2025. Adresse: https://www.infosecurity-magazine.com/blogs/trickbot-mikrotik-connection/.
[307] Google, *Nest thermostat technical specifications*. besucht am 19. Feb. 2025. Adresse: https://support.google.com/googlenest/answer/9230098/.
[308] The MITRE Corporation, *CVE-2022-45045*, 3. Aug. 2024. besucht am 21. Feb. 2025. Adresse: https://www.cve.org/CVERecord?id=CVE-2022-45045.
[309] J. Baines, *Xiongmai IoT Exploitation*, 29. Nov. 2022. besucht am 21. Feb. 2025. Adresse: https://vulncheck.com/blog/xiongmai-iot-exploitation.
[310] Speed Guide, Inc., *Port 34567 Details*. besucht am 21. Feb. 2025. Adresse: https://www.speedguide.net/port.php?port=34567.
[311] Cloudflare, Inc., *DNS amplification attack*. besucht am 21. Feb. 2025. Adresse: https://www.cloudflare.com/learning/ddos/dns-amplification-ddos-attack/.
[312] Daxtorim und worldcitizencane, *Why so many queries for . and sl ?* 21. Feb. 2021. besucht am 21. Feb. 2025. Adresse: https://www.reddit.com/r/pihole/comments/low99y/why_so_many_queries_for_and_sl/.
[313] Cloudflare, Inc., *NTP amplification DDoS attack*. besucht am 21. Feb. 2025. Adresse: https://www.cloudflare.com/learning/ddos/ntp-amplification-ddos-attack/.
[314] Speed Guide, Inc., *Port 123 Details*. besucht am 21. Feb. 2025. Adresse: https://www.speedguide.net/port.php?port=123.
[315] A. Helmenstine, *How Many Zeros in a Million, Billion, and Trillion?* 14. Jan. 2023. besucht am 31. Jan. 2025. Adresse: https://sciencenotes.org/how-many-zeros-in-a-million-billion-and-trillion/.
[316] M. Peischl und A. Habegger, *BFH LaTeX Manual documentation*. besucht am 5. Okt. 2024. Adresse: https://latex.ti.bfh.ch/.
[317] D. Crocker und P. Overell, *Augmented BNF for Syntax Specifications: ABNF*, RFC 5234, Jan. 2008. DOI: 10.17487/RFC5234. Adresse: https://www.rfc-editor.org/info/rfc5234.
[318] APNIC, *APNIC*. besucht am 23. Okt. 2024. Adresse: https://www.apnic.net/.
[319] J. A. Hawkinson und T. J. Bates, *Guidelines for creation, selection, and registration of an Autonomous System (AS)*, RFC 1930, März 1996. DOI: 10.17487/RFC1930. Adresse: https://www.rfc-editor.org/info/rfc1930.
[320] S. Alrwais et al., "Under the Shadow of Sunshine: Understanding and Detecting Bulletproof Hosting on Legitimate Service Provider Networks," in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017, S. 805–823. DOI: 10.1109/SP.2017.32.
[321] Center for Applied Internet Data Analysis, *About CAIDA*. besucht am 23. Okt. 2024. Adresse: https://www.caida.org/about/.

[322] Cloudflare, Inc., *What is a content delivery network (CDN)? | How do CDNs work?* Besucht am 22. Jan. 2025. Adresse: https://www.cloudflare.com/learning/cdn/what-is-a-cdn/.
[323] Software in the Public Interest, Inc. et al., *Reasons to use Debian*, 11. Feb. 2024. besucht am 9. Nov. 2024. Adresse: https://www.debian.org/intro/why_debian.
[324] Elasticsearch B.V., *Elasticsearch: The heart of the Elastic Stack*. besucht am 27. Dez. 2024. Adresse: https://www.elastic.co/elasticsearch.
[325] E. Borges, *Cybersecurity Fingerprinting: Concept, Insights and Strategies*, 4. März 2024. besucht am 7. Aug. 2024. Adresse: https://www.recordedfuture.com/threat-intelligence-101/vulnerability-management-threat-hunting/fingerprinting-in-cybersecurity.
[326] T. Li, D. Farinacci, S. P. Hanks, D. Meyer und P. S. Traina, *Generic Routing Encapsulation (GRE)*, RFC 2784, März 2000. DOI: 10.17487/RFC2784. Adresse: https://www.rfc-editor.org/info/rfc2784.
[327] J. McGinty, *Introducing 'innernet'*, 30. März 2021. besucht am 30. Okt. 2024. Adresse: https://blog.tonari.no/introducing-innernet.
[328] Elasticsearch B.V., *Kibana: Explore, Visualize, Discover Data*. besucht am 8. Jan. 2025. Adresse: https://www.elastic.co/kibana.
[329] J. Sermersheim, *Lightweight Directory Access Protocol (LDAP): The Protocol*, RFC 4511, Juni 2006. DOI: 10.17487/RFC4511. Adresse: https://www.rfc-editor.org/info/rfc4511.
[330] Microsoft, *Microsoft Teams - Videokonferenzen, Besprechungen, Anrufe*. besucht am 1. März 2025. Adresse: https://www.microsoft.com/de-ch/microsoft-teams/group-chat-software.
[331] NetBird, *Introduction to NetBird*. besucht am 30. Okt. 2024. Adresse: https://docs.netbird.io/.
[332] D. Moore, C. Shannon, G. M. Voelker und S. Savage, "Network Telescopes: Technical Report," UC San Diego: Department of Computer Science & Engineering, 7. Juli 2004. Adresse: https://escholarship.org/uc/item/1405b1bz.
[333] The Netfilter webmasters, *What is nftables?* Besucht am 30. Okt. 2024. Adresse: https://www.nftables.org/projects/nftables/index.html.
[334] C. Meadows, J. Graybill, K. Davis und M. Shah, *Introducing OpenSearch*, 12. Apr. 2021. besucht am 27. Dez. 2024. Adresse: https://aws.amazon.com/blogs/opensource/introducing-opensearch/.
[335] OpenSearch contributors, *OpenSearch Dashboards*. besucht am 8. Jan. 2025. Adresse: https://opensearch.org/docs/latest/dashboards/.
[336] T. Keary, *PCAP: Packet Capture, what it is & what you need to know*, 29. Sep. 2023. besucht am 25. Okt. 2024. Adresse: https://www.comparitech.com/net-admin/pcap-guide/.
[337] G. Harris und M. Richardson, "PCAP Capture File Format," Internet Engineering Task Force, Internet-Draft draft-ietf-opsawg-pcap-04, Aug. 2024, Work in Progress, 10 S. Adresse: https://datatracker.ietf.org/doc/draft-ietf-opsawg-pcap/04/.
[338] M. Bishop, *HTTP/3*, RFC 9114, Juni 2022. DOI: 10.17487/RFC9114. Adresse: https://www.rfc-editor.org/info/rfc9114.
[339] Google. "Chrome is deploying HTTP/3 and IETF QUIC," besucht am 10. Jan. 2025. Adresse: https://blog.chromium.org/2020/10/chrome-is-deploying-http3-and-ietf-quic.html.
[340] D. Damjanovic. "QUIC and HTTP/3 Support now in Firefox Nightly and Beta," besucht am 10. Jan. 2025. Adresse: https://hacks.mozilla.org/2021/04/quic-and-http-3-support-now-in-firefox-nightly-and-beta/.
[341] K. Mackie. "Microsoft Embracing Native QUIC in Newer Windows OSes and Edge Browser," besucht am 10. Jan. 2025. Adresse: https://redmondmag.com/articles/2021/08/26/native-quic-in-windows-edge.aspx.
[342] RIPE, *Welcome to the RIPE NCC*. besucht am 23. Okt. 2024. Adresse: https://www.ripe.net/.
[343] Shodan, *Search Engine Improvements*, 6. Sep. 2020. besucht am 23. Okt. 2024. Adresse: https://blog.shodan.io/search-engine-improvements/.

[344] M. Fedor, M. L. Schoffstall, J. R. Davin und D. J. D. Case, *Simple Network Management Protocol (SNMP)*, RFC 1157, Mai 1990. DOI: 10.17487/RFC1157. Adresse: https://www.rfc-editor.org/info/rfc1157.

[345] C. M. Lonvick und T. Ylonen, *The Secure Shell (SSH) Protocol Architecture*, RFC 4251, Jan. 2006. DOI: 10.17487/RFC4251. Adresse: https://www.rfc-editor.org/info/rfc4251.

[346] L. Poettering, Z. Jędrzejewski-Szmek, F. Sumsal, B. Franzke, T. Bernard, F. Brandenburger et al., *System and Service Manager*, 23. Feb. 2024. besucht am 6. Nov. 2024. Adresse: https://github.com/systemd/systemd/blob/main/docs/index.md.

[347] The Tor Project Inc., *Relay Operations, Exit Relay*. besucht am 25. Okt. 2024. Adresse: https://community.torproject.org/relay/setup/exit/.

[348] M. Higgins, *VPN WireGuard: What is it and how does it work?* 17. Aug. 2023. besucht am 30. Okt. 2024. Adresse: https://nordvpn.com/blog/vpn-wireguard/.

[349] M. Wojciakowski, C. Loewen, D. Wilson, W. Bjorn, C. R. Ramirez, A. Jenks et al., *How to install Linux on Windows with WSL*, 19. Nov. 2024. besucht am 19. Feb. 2025. Adresse: https://learn.microsoft.com/en-us/windows/wsl/install.

[350] The Zeek Project, *About Zeek*. besucht am 25. Okt. 2024. Adresse: https://docs.zeek.org/en/master/about.html.

[351] C. for Internet Security Inc. (CIS), *CIS Debian Linux 12 Benchmark*, Version 1.1.0, 27. Sep. 2024.

[352] S. Grover, *Upgrading Malcolm*, 12. Sep. 2024. besucht am 20. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/malcolm-upgrade.md.

[353] S. Grover, *Appendix F - Upgrades*, 3. Apr. 2024. besucht am 20. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/hedgehog-upgrade.md.

[354] Hetzner Online GmbH, *Primary IP Configuration*, 28. Dez. 2023. besucht am 11. Dez. 2024. Adresse: https://docs.hetzner.com/cloud/servers/primary-ips/primary-ip-configuration/.

[355] M. Pietroforte, *Assign an IPv6 address to an EC2 instance (dual stack)*, 15. Dez. 2020. besucht am 11. Dez. 2024. Adresse: https://4sysops.com/archives/assign-an-ipv6-address-to-an-ec2-instance-dual-stack/.

[356] M. Bender, J. Medina, C. McClister, J. Borsecnik und A. Sudbring, *Add a dual-stack network to an existing virtual machine*, 25. Juli 2024. besucht am 13. Dez. 2024. Adresse: https://learn.microsoft.com/en-us/azure/virtual-network/ip-services/add-dual-stack-ipv6-vm-portal.

[357] V. Kavuru et al., *Prepare a Debian VHD for Azure*, 22. Aug. 2024. besucht am 13. Dez. 2024. Adresse: https://learn.microsoft.com/en-us/azure/virtual-machines/linux/debian-create-upload-vhd.

[358] S. Khattak und Johanna, *Re-reading data in the input framework*, 29. Juni 2012. besucht am 30. Dez. 2024. Adresse: https://community.zeek.org/t/re-reading-data-in-the-input-framework/2343.

[359] The Zeek Project, *Troubleshooting*, 31. Dez. 2024. besucht am 2. Jan. 2025. Adresse: https://docs.zeek.org/en/master/troubleshooting.html.

[360] A. Merck, *Zeek Configuration Options*, 31. Dez. 2024. besucht am 2. Jan. 2025. Adresse: https://mistral.pages.oit.duke.edu/mistral-ids-docs/zeek_configuration.html.

[361] S. Grover, *Recommended system requirements*, 12. Sep. 2024. besucht am 6. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/system-requirements.md.

[362] Berner Fachhochschule Hochschulbibliothek, *Kurzanleitung ARBOR*, 4. Apr. 2023. besucht am 8. Nov. 2024. Adresse: https://arbor.bfh.ch/images/general/ARBOR_Kurzanleitung.pdf.

[363] S. Grover, *Search Queries in Arkime and OpenSearch Dashboards*, 20. Juni 2023. besucht am 3. Jan. 2025. Adresse: https://github.com/cisagov/Malcolm/blob/main/docs/README.md.

[364] I. Red Hat, *What does the 'RX-ERR' field mean in the output of netstat -ni and how can we reset the counter to zero?* 7. Aug. 2024. besucht am 29. Jan. 2025. Adresse: https://access.redhat.com/solutions/30575.

[365] S. Grover, *Malcolm Releases*, 21. Okt. 2024. besucht am 6. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/releases.

[366] S. Grover, *arkime-offline.env.example*, 7. Dez. 2023. besucht am 2. Nov. 2024. Adresse: https://github.com/cisagov/Malcolm/blob/main/config/arkime-offline.env.example.

[367] J. Walton, P. Wise, R. Matsumoto, M. Bar, B. Roucaries et al., *SystemGroups*, 9. Okt. 2023. besucht am 29. Jan. 2025. Adresse: https://wiki.debian.org/SystemGroups.

[368] tommie, *Unofficial Innernet APT Repository*, 17. Mai 2024. besucht am 1. Nov. 2024. Adresse: https://github.com/tommie/innernet-debian/blob/main/README.md.

[369] Ansible project contributors, *ansible-vault*, 3. Dez. 2024. besucht am 10. Dez. 2024. Adresse: https://docs.ansible.com/ansible/latest/cli/ansible-vault.html.

[370] Internet Assigned Numbers Authority, *Protocol Numbers*, 6. Nov. 2024. besucht am 24. Okt. 2024. Adresse: https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml.

[371] F. Semperboni, *How to decapsule ERSPAN tunnel*, 16. Feb. 2021. besucht am 6. Nov. 2024. Adresse: https://www.ciscozine.com/decapsule-erspan-tunnel/.

[372] Red Hat, Inc., *Working with systemd unit files*, 16. Aug. 2024. besucht am 1. Nov. 2024. Adresse: https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/using_systemd_unit_files_to_customize_and_optimize_your_system/assembly_working-with-systemd-unit-files_working-with-systemd.

[373] rex, *Systemd wait for network interface to be up before running service*, 17. Jan. 2017. besucht am 1. Nov. 2024. Adresse: https://unix.stackexchange.com/a/417839.

[374] The Zeek Project, *Zeek Cluster Setup*, 13. Dez. 2024. besucht am 18. Dez. 2024. Adresse: https://docs.zeek.org/en/master/cluster-setup.html.

[375] L. Call et al., *Unattended Upgrades*, 8. Sep. 2024. besucht am 29. Nov. 2024. Adresse: https://wiki.debian.org/UnattendedUpgrades.

[376] W. Tu und G. Rose, *ERSPAN Support for Linux*, 11. Juni 2018. besucht am 1. Nov. 2024. Adresse: https://lpc.events/event/2/contributions/98/attachments/97/115/erspan-linux.pdf.

[377] Progress Software Corporation and/or its subsidiaries or affiliates, *ERSPAN on Linux*, 22. Aug. 2024. besucht am 1. Nov. 2024. Adresse: https://support.kemptechnologies.com/hc/en-us/articles/19373117675277-ERSPAN-on-Linux.

[378] M. Cotton, D. K. C. Almeroth, Z. AlBanna und D. Meyer, *IANA Guidelines for IPv4 Multicast Address Assignments*, RFC 3171, Aug. 2001. DOI: 10.17487/RFC3171. Adresse: https://www.rfc-editor.org/info/rfc3171.

[379] A. Keller, *Manual tc Packet Filtering and netem*, 20. Juli 2006. besucht am 23. Dez. 2024. Adresse: http://tcn.hypert.net/tcmanual.pdf.

[380] The Zeek Project, *Script Reference, base/bif/zeek.bif.zeek*. besucht am 23. Nov. 2024. Adresse: https://docs.zeek.org/en/master/scripts/base/bif/zeek.bif.zeek.html.

Abbildungsverzeichnis

Tabellenverzeichnis

Quelltextverzeichnis

Abkürzungsverzeichnis

ABNF Angereicherte Backus-Naur Form. 32, 66, 93

APNIC Asia Pacific Network Information Centre. 7, 93

AS Autonomous System. 4, 6, 9, 28, 31, 46, 49, 93, 164

ASN Autonomous System Number. 7, 93

BFH Berner Fachhochschule. 26, 104, 106, 108, 109, 111, 116–118, 127–129, 131, 133, 135, 136, 158, 159

CAIDA Center for Applied Internet Data Analysis. 7, 93

CDN Content Delivery Network. 31, 93

CISA Cybersecurity Infrastructure Security Agency. 10

DTLS Datagram Transport Layer Security. 44, 49, 94

ERSPAN Encapsulated Remote Switch Port Analyzer. 11–16, 21, 27, 30, 31, 67, 86, 87, 89, 90, 94, 97, 116–119, 121, 131, 136, 137, 139, 140, 143, 144, 149, 151, 152, 155, 157, 158, 161–163, 168, 171

FQDN Fully Qualified Domain Name. 8, 9, 17–19, 23, 60, 65, 89, 94, 117, 121, 137, 165

GRE Generic Routing Encapsulation. 13, 14, 94, 149

IANA Internet Assigned Numbers Authority. 8, 42–46, 94

INL Idaho National Laboratory. 10

IoC Indicators of Compromise. ii, 7–9, 18, 42, 43, 53, 60, 86, 94

IPS Intrusion Prevention System. 6

LDAP Lightweight Directory Access Protocol. 60, 95

MAS Master of Advanced Studies. i–iv, 1–184

MTU Maximum Transmission Unit. 15, 16, 86, 95, 116, 131, 151, 157

NCSC National Cyber Security Centre. 1, 7, 18

NTP Network Time Protocol. 13, 31, 45, 60, 63, 66, 96, 129, 147, 156

NVR Network Video Recorder. 60

PCAP Packet Capture. 10, 21, 30, 47, 51, 95, 96, 98, 138, 152

RIPE Réseaux IP Européens. 7, 18, 37, 38, 44, 51, 96, 121, 136

SIP Session Initiation Protocol. 43, 46, 97

SNMP Simple Network Management Protocol. 60, 97

SPAN Switch Port Analyzer. 97

SSH Secure Shell. 4–6, 9, 11, 13, 14, 27, 42, 51, 58–60, 89, 97, 108, 117, 134, 137, 140–143, 147, 148, 155, 156, 161, 162

TC Traffic Control. 11, 13, 97, 117

TGA Target Generation Algorithm. 5

TLS Transport Layer Security. 5, 9, 47, 94

ufw Uncomplicated Firewall. 11, 14, 51, 98, 119, 155, 156

URI Uniform Resource Identifier. 1, 8, 17, 22–24, 32, 33, 53, 64, 89, 98, 121, 179, 180

VPN Virtual Private Network. 6, 11–15, 21, 26, 27, 63, 67, 89, 97, 98, 106, 121, 128, 129, 134, 137, 139–144, 149, 150, 155–158, 161–163

WSL Windows Subsystem for Linux. 50, 98

Glossar

6to4

Mechanismus zur Kommunikation von isolierten IPv6-Netzwerken über IPv4 [247]. Der reservierte IPv6-Adressbereich hierfür lautet 2002::/16 [246, 247]. 37–39, 42–45, 59, 86, 121

Angereicherte Backus-Naur Form (ABNF)

Metasprache zur Definition einer anderen Sprache oder Syntax [317]. 32, 66

Ansible

Automatisierungs-Software, bei welchem der gewünschte Zustand eines Systems in einem visuell lesbaren Skript namens Playbook festhalten wird [147]. 11, 13, 26, 27, 66, 89, 108, 116–119, 134, 137, 139–144, 150, 155, 161, 162

Asia Pacific Network Information Centre (APNIC)

Regionale Internet-Registrierungsstelle für die Asien-Pazifik-Region [318]. 7

Arkime

Lösung zur Netzwerkanalyse und Sammlung von Netzwerkpaketen („Full Packet Capture“) [107]. 9–11, 18, 20, 21, 24, 25, 27, 29, 30, 37, 41, 46, 47, 49, 50, 55, 56, 61, 67, 86, 95, 115, 119–122, 135, 138, 143, 151, 153, 163, 171

Autonomous System (AS)

Menge von Routern unter einer einzelnen Verwaltung mit gemeinsamem und klarem Routing-Verhalten im Internet [319]. 4, 6, 9, 28, 31, 46, 49, 93, 164

Autonomous System Number (ASN)

Global eindeutige Nummer, die einem AS zugewiesen wird [319]. 7

Banner Grabbing

Methode, um Informationen wie z.B. Versionsnummern über Dienste hinter offenen Ports zu erfahren [47]. 6, 95

Bulletproof Hosting

Gegen Beschwerden resiliente Dienstleistungen/Infrastruktur für Kriminelle [320]. 6

Center for Applied Internet Data Analysis (CAIDA)

Netzwerk-Forschungs-Zentrum am San Diego Supercomputer Center [321]. 7

Content Delivery Network (CDN)

Geografisch verteilte Gruppe von Servern, die Daten zwischenspeichern [322]. 31

Censys

Suchmaschine für Komponenten im Internet, die selber den IPv4-Adressraum und ermittelte IPv6-Adressen scannt [20, 81]. ii, 1, 4, 6, 7, 9, 18, 19, 48, 53, 57, 64, 86, 90, 95, 120, 136, 177

Debian

Linux-Distribution [323]. 11, 13, 27, 31–33, 49, 63, 108, 116, 121, 128, 134, 137, 140, 141, 155, 158–160, 163

Datagram Transport Layer Security (DTLS)

Protokoll zur Anwendung von TLS unter UDP [261]. 44, 49

Elasticsearch

Software-Engine zur Suche und Analyse sowie Speicherung von Daten [324]. 95, 96, *siehe* OpenSearch

Encapsulated Remote Switch Port Analyzer (ERSPAN)

Format zur Weiterleitung von durch Port-Mirroring aufgezeichneten Netzwerkverkehr über IP-Netzwerke [130, 131]. 11–16, 21, 27, 30, 31, 67, 86, 87, 89, 90, 97, 116–119, 121, 131, 136, 137, 139, 140, 143, 144, 149, 151, 152, 155, 157, 158, 161–163, 168, 171

Fingerprint

Korrelation von Daten zur Identifikation von u. a. spezifischen Netzwerkdiensten, Anwendungen, Konfigurationen oder Cyber-Gefahren [325]. 9, 10, 25, 47–52, 54, 59, 60, 64–66, 86, 88, 89, 95, 103, 105, 115, 122

Fully Qualified Domain Name (FQDN)

Absoluter Name einer Domain in der Baum-Hierarchie des Domain Name Systems (DNS). 17–19, 23, 60, 65, 89, 117, 121, 137, 165

Generic Routing Encapsulation (GRE)

Format zur Kapselung bzw. Übertragung von Paketen auf dem Netzwerk-Layer über ein anderes Netzwerk-Layer-Protokoll [326]. 13, 14, 149

Hedgehog Linux

Appliance zur Aufzeichnung von Netzwerkverkehr, Teil von Malcolm [143]. 11–14, 17, 19, 21, 22, 26, 29, 32, 67, 89, 90, 97, 116, 117, 119, 121, 131, 134, 137, 139–141, 147–149, 151, 152, 154, 155, 158, 163, 165, 171, 179

Horizontaler Scan

Port-Scan, bei welchem derselbe Port auf mehreren Zielen geprüft wird [16]. 6

Internet Assigned Numbers Authority (IANA)

Behörde für die Zuordnung von Namen und Nummern im Internet [71]. 8, 42–46

innernet

Konfigurations-Software für WireGuard [128, 327]. 11–13, 89, 116, 131, 139, 141, 142, 149, 150, 156, 157, 161, 162

Indicators of Compromise (IoC)

Merkmale zur Identifizierung, Verfolgung und Blockierung böswilliger Aktivitäten [100]. 8, 9, 18, 42, 43, 53, 60, 86

IVRE

Netzwerkaufklärungs-Software zum aktiven Scannen im Internet [57]. 6, 10

JA4+

Auswahl an Netzwerk-Fingerprinting-Methoden (siehe Tabelle JA4+ Implementationen [106] auf Seite 9), die von diversen Anwendungen und Diensten wie Wireshark, dem Netzwerk-Analyse-Werkzeug Arkime [107] oder Censys unterstützt werden [106]. 9, 10, 25, 47, 48, 50, 54, 59, 60, 64–66, 88, 89, 95, 121, 122

Kibana

Software zur Visualisierung von Elasticsearch-Daten [328]. 96, *siehe* OpenSearch Dashboards

Lightweight Directory Access Protocol (LDAP)

Protokoll zum Zugriff auf Verzeichnisdienste [329]. 60

Malcolm

Sammlung an Werkzeugen zur Netzwerkverkehrsanalyse, vereint unter anderem die Anwendungen Arkime und Zeek. Verarbeitet Netzwerkverkehr in Form von PCAP-Paketaufzeichnungsdateien oder Zeek-Logs [113]. ii, 10–14, 17, 18, 20, 21, 24, 26–30, 47, 60, 61, 64, 66, 67, 86, 89, 94, 97, 115–117, 120, 131, 134, 135, 137–139, 143, 144, 146–148, 152, 153, 158, 183

Manpage

kurz für „manual page“. Dokumentation, die bei Unix- oder Unix-ähnlichen Betriebssystemen mit dem Befehl `man` angezeigt werden kann. 156

masscan

Anwendung für Internet-weite Scans, kann unter anderem beliebige Adress- und Portbereiche anpeilen und Banner Grabbing bzw. „Banner Checking“ für diverse Anwendungen (FTP, HTTP, IMAP4, SSH, SSL, etc.) durchführen kann [49]. 6, 50, 66

Microsoft Teams

Kommunikationsplattform von Microsoft (Chats, Besprechungen, etc.) [330]. 125–127, 132, 135

Maximum Transmission Unit (MTU)

Maximale grösse eines Layer-3-Pakets in Bytes [6]. 15, 16, 86, 116, 131, 151, 157

NetBird

Platform zur Erstellung von sicheren, privaten Netzwerken auf Basis von WireGuard [331]. 115, 131

Netzwerkteleskop (Network Telescope)

Teil eines gerouteten IP-Addressbereichs, in welchem kein oder kaum legitimer Netzwerkverkehr auftritt [332]. Die Analyse von unerwarteten Anfragen, die dort eintreffen, können eigene Perspektiven auf global auftretende Netzwerk-Ereignisse aufzeigen (im Vergleich zu einzelnen Hosts oder End-zu-End-Kommunikationen) [332]. 2

nftables

Netzwerkpaket-Klassifizierungs-Framework im Linux Kernel (Firewalling, NAT, Paketmodifikation) [333]. 115, 131

Nmap

Anwendung zur Netzwerkerkundung und Sicherheitsprüfung [44]. 6, 9, 50, 99

Network Time Protocol (NTP)

Protokoll zur Synchronisation von Uhren auf Computern [263]. 13, 31, 45, 60, 63, 66, 129, 147, 156

OpenSearch

Software-Engine zur Suche und Analyse von u. a. Netzwerkverkehr-Metadaten, Fork von Elasticsearch [140, 197, 334]. 11, 20, 28, 96, 119, *siehe* Elasticsearch

OpenSearch Dashboards

Oberfläche zur Visualisierung von OpenSearch-Daten, Fork von Kibana [197, 334, 335]. 20, 21, 27, 28, 30, 32, 34–36, 39–41, 47, 49, 52, 55, 60, 62, 67, 86, 87, 89, 120–122, 135, 137, 138, 143, 144, 163, 183, 184, *siehe* Kibana

Painless

Für Elasticsearch entwickelte Skriptsprache. 28

Passive DNS

Aufzeichnung bzw. Protokollierung von DNS-Anfragen mit Zeitstempel in Datenbanken. 31, 121

Packet Capture (PCAP)

„Packet Capture"-Dateiformat zur Ablage von Netzwerkpaket-Daten, die von einem Netwerk-Interface aufgezeichnet wurden [336, 337]. 10, 21, 30, 47, 51, 95, 98, 138, 152

Port-Mirroring

Funktion zur Spiegelung von Netzwerkverkehr auf Netzwerk-Interfaces [130, 131]. 2, 11–16, 21, 26, 27, 30, 86, 90, 94, 115, 116, 131, 136, 155, 157, 158, 161, 168, 179

Probe

Englischer Begriff zur Bezeichnung für eine Messung (beziehungsweise Scan) von Scan-Quellen oder zur Bezeichnung für die Scan-Quelle selbst [15]. 8, 18, 38, 44, 121, 136

QUIC

Auf UDP basierendes Transportprotokoll, das u. a. für HTTP/3 [338] verwendet wird und in diversen Webbrowsern bereits im Einsatz ist [262][339][340][341]. 44, 49, 52

Réseaux IP Européens (RIPE)

Regionale Internet-Registrierungsstelle für den Raum Europa, Mittleren Osten und Zentralasien [342]. 7, 18, 37, 38, 44, 51, 121, 136

Scan-Quelle

Instanz, die ein Scan-Ziel kontaktiert (Beispielsweise mittels ICMP-Echo-Request-Paket oder Port-Scan). ii, 1, 2, 4, 6–9, 19, 20, 28, 29, 31, 34, 35, 38–40, 46, 53, 54, 57, 60, 62–67, 86, 89, 105, 113, 119–121, 128, 135, 136, 164

Scan-Target

Englische Variante des Begriffs „Scan-Ziel“. 2, 109, 125, 128, 129, 134, *siehe* Scan-Ziel

Scan-Ziel

Zugangspunkt im Internet, der Untersuchungen bzw. Scans von diversen Quellen erfährt. Spiegelt im Rahmen dieser Arbeit dessen Netzwerkverkehr über einen VPN-Tunnel mittels ERSPAN zur Appliance bzw. zum Sensor mit Hedgehog Linux, der die Daten an die Malcolm-Instanz weiterleitet. ii, 2, 3, 11–14, 16, 18, 21, 26, 27, 29–32, 37–39, 42–46, 51, 52, 54–67, 86–89, 96, 97, 106–111, 116, 118–121, 123, 124, 131, 134–137, 139–144, 155, 156, 158–162, 179, 181, 183

Scapy

Python-Anwendung zur Bearbeitung von Netzwerkpaketen (Senden, Sniffen, Analysieren und Formen) [151]. 11, 21, 117, 179

Shodan

Suchmaschine für Internet-Komponenten, die dafür selber Scans im Internet (IPv4 und IPv6) durchführt [19, 343]. ii, 1, 4–7, 18, 37, 38, 42–45, 53, 57, 63, 64, 86, 120, 136

Session Initiation Protocol (SIP)

Anwendungs-Protokoll u. a. zur Bearbeitung von Sitzungen (in Form von Internet-Telefonie oder Multimedia-Konferenzen) mit einem oder mehreren Teilnehmenden [260]. 43, 46

Simple Network Management Protocol (SNMP)

Protokoll zur u. a. Überwachung von Elementen in einem Netzwerk [344]. 60

Switch Port Analyzer (SPAN)

Funktion zur Spiegelung von Netzwerkverkehr auf Netzwerk-Interfaces [130, 131]. *siehe* Port-Mirroring

Secure Shell (SSH)

Protokoll für sicheren Fernzugriff und andere sichere Netzwerkdienste über ein unsicheres Netzwerk [345]. 4–6, 9, 11, 13, 14, 27, 42, 51, 58–60, 89, 108, 117, 134, 137, 140–143, 147, 148, 155, 156, 161, 162

systemd

System- und Service-Manager für ein Linux System [346]. 14, 137, 139, 140, 151, 157, 162, 163, 168, 171

Traffic Control (TC)

Anwendung zur Konfiguration des Netzwerkverkehrs im Linux Kernel (Teil der `iproute2`-Software-Suite im Linux Kernel) [132, 134–136]. 11, 13, 117

tcpdump

Anwendung zur Analyse von Netzwerkverkehr [156]. 21

Tor

„The onion routing network“ dient als Methode, das Internet mit so viel Privatsphäre wie möglich zu benuzten, wobei der Verkehr über mehrere Server / „Relays“ geleitet und bei jedem Schritt auf dem Weg verschlüsselt wird [280]. 18, 52, 61, 98, *siehe* Tor Netzwerk

Tor Browser

Webbrowser auf Basis von Firefox, der einen Zugang zu Tor ermöglicht [103]. 32

Tor Exit Node / Tor Exit Relay

Typ von Server / „Relay“, über welchen der Netzwerkverkehr aus dem Tor Netzwerk ins öffentliche Internet fliesst [102, 347]. 9, 18, 46, 52, 61, 90, 117, 178

Tor Netzwerk

Verteiltes Netzwerk mit Mitgliedern („Relays“) auf der ganzen Welt, das beim Abhören der Internetverbindung das Auffinden besuchter Webseiten verhindert und die Webseiten daran hindert, den physischen Standort zu ermitteln [103]. 9, 66, 98

TShark

Kommandozeilen-Variante von Wireshark [61]. 10, *siehe* Wireshark

Uncomplicated Firewall (ufw)

Anwendung zur Verwaltung der Software-Firewall [148, 149]. 11, 14, 51, 119, 155, 156

Uniform Resource Identifier (URI)

Zeichenfolge, die eine logische oder physische Ressource identifiziert. 17, 22–24, 32, 33, 53, 64, 89, 121, 179, 180

Virtual Private Network (VPN)

Mechanismus zum Aufbau einer sicheren Verbindung zwischen einem Host und einem Netzwerk oder zwei Netzwerken über ein unsicheres Medium. 6, 11–15, 21, 26, 27, 63, 67, 89, 97, 98, 106, 121, 128, 129, 134, 137, 139–144, 149, 150, 155–158, 161–163

WireGuard

Sicherer Netzwerk-Tunnel, implementiert als virtuelles Netzwerk-Interface (VPN-Protokoll) [127, 348]. 11, 12, 15, 63, 94, 95, 116, 134, 139

Wireshark

Anwendung zur Analyse von Netzwerkverkehr mit grafischer Oberfläche [61]. 6, 9, 10, 30, 47, 87, 95, 98, 157

Windows Subsystem for Linux (WSL)

Erlaubt die Installation einer Linux Distribution direkt in Windows ohne die Nutzung einer virtuellen Maschine [349]. 50

YubiKey

Hardware Security-Token zur Identifizierung und Authentisierung. 140, 142

Zeek

Anpass- und erweiterbare Software zur Analyse von Netzwerkverkehr, mit Fokus auf Zusammenfassen von beobachteten Protokollen und Dateiinhalten (Daten werden nicht vollständig im PCAP-Format gespeichert) [350]. 10, 13, 17–25, 28–35, 37, 39, 41, 50, 52, 54, 55, 58, 61, 62, 64, 66, 67, 86–90, 95, 110, 113, 115–119, 121, 123, 134, 137, 138, 140, 153, 154, 161, 163–165, 176, 177, 179, 180

Zenmap

Grafische Oberfläche für die Anwendung Nmap [44]. 6, *siehe* Nmap

ZGrab

Anwendung für Internet-weite Scans, baut auf ZMap auf und analysiert Ziele auf Anwendungsebene [37]. 4, 6

ZMap

Anwendung für Internet-weite Scans, kann beispielsweise ein TCP-SYN-Segment an jede IPv4-Adresse auf Port 25 senden, um alle darüber erreichbaren Server zu ermitteln [37]. ii, 4, 6, 7, 9, 17, 23, 34, 35, 42–45, 50, 52, 53, 59, 63–66, 86, 99, 117, 121, 122

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die hier vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Sämtliche Ausführungen, die anderen Schriften wörtlich oder sinngemäss entnommen wurden, habe ich als solche kenntlich gemacht. KI-gestützte Werkzeuge wurden lediglich zur Findung weiterer Quellen als Suchmaschine verwendet.

Hiermit stimme ich zu, dass die vorliegende Arbeit in elektronischer Form mit entsprechender Software überprüft wird.

7. März 2025

Unterschrift entfernt in öffentlicher Version

Mauro Guadagnini

Anhang

Anhangsverzeichnis

Anhang A. Projektmanagement

A.1. Zeitplan und Ziel-Definitionen (Arbeitspakete)

Die Zeitplanung dieser Arbeit etabliert die nachfolgenden Phasen mit entsprechender Ziel-Definition. Für jede Phase werden sogenannte Kann- und Muss-Ziele definiert. Ein Ziel entspricht hierbei jeweils einem Arbeitspaket, das innerhalb der zugehörigen Phase behandelt wird. Im Falle eines Kann-Ziels wird dieses nur berücksichtigt, sofern es die gegebene Zeit zulässt.

Die Ziel-Typen erhalten zur Darstellung folgende Symbole:

- ● Muss-Ziel
- ○ Kann-Ziel

Beim entsprechenden Ziel wird ebenfalls aufgeführt, ob dieses im Rahmen dieser Arbeit erreicht wurde. Hierzu wird folgende Darstellung verwendet:

- ✓ Ziel erfüllt
- ⊖ Ziel teilweise erfüllt
- ✗ Ziel nicht erfüllt

Tabelle A.1.: Phasen entsprechend Zeitplan

Zeitraum	Phase und grobe Zielbeschreibung	Kapitel	Aufwand [h]
bis 31. Oktober 2024	P1 **Vorbereitung** Struktur, Kickoff, Vorgehen	A.1.1	20
bis 17. November 2024	P2 **Recherche** Theoretische Grundlagen, Fingerprinting und Identifikation, Werkzeugwahl	A.1.2	40
bis 31. Dezember 2024	P3 **Planung und Design** Prozedur, Analyse-Ablauf, Aufbau lokal, 1. Review	A.1.3	120
bis 31. Januar 2025	P4 **Durchführung** Aufbau global, Scan-Erfassung und -Speicherung, Stichproben-Analysen, 2. Review	A.1.4	100
bis 23. Februar 2025	P5 **Auswertung** Analyse, Ergebnisse festhalten und veranschaulichen	A.1.5	60
bis 10. März 2025	P6 **Abschluss** Bericht, Book-Beitrag, Präsentation	A.1.6	20
		Total Aufwand [h]	360

A.1.1. P1 Vorbereitung

Bericht inklusive Struktur vorbereiten

Phase: P1 Vorbereitung
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 25. August 2024

Beschrieb:

Den Bericht dieser Master Thesis gilt es gemäss Vorgaben der BFH vorzubereiten. Die entsprechende Gliederung der Kapitel sowie vorgegebene Verzeichnisse und Inhalte sind einzufügen oder entsprechende Platzhalter zu setzen. Verfasst wird der Bericht in LaTeX mit unter anderem entsprechenden LaTeX-Software-Paketen der BFH [316].

Kickoff Meeting durchführen und Termine definieren

Phase: P1 Vorbereitung
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 18. September 2024
Der Termin für die Präsentation der Master Thesis wurde am 3. Oktober 2024 definiert, wobei Hansjürg Wenger und Rolf Lanz (als Stellvertretung von Bruce Nikkel) teilnehmen werden (siehe Kapitel A.4.5)

Beschrieb:

Mit den Lead- und Co-Experten Prof. Hansjürg Wenger und Prof. Dr. Bruce Nikkel wird ein Kickoff Meeting durchgeführt. Innerhalb des Meetings werden das weitere Vorgehen basierend auf dem Themenantrag fachlich und organisatorisch besprochen. Es gilt die Erwartungen der Experten abzuholen, die weitere Kommunikation festzulegen und nachfolgende Termine zu vereinbaren.

Das Protokoll zum Kickoff Meeting ist in Kapitel A.4.4 festzuhalten.

Vorgehen ausarbeiten

Phase: P1 Vorbereitung
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 16. Oktober 2024

Beschrieb:

Das Vorgehen der Master Thesis sowie Muss- und Kann-Ziele sind in Form von Arbeitspaketen in Kapitel A.1 festzuhalten.

Entsprechende Inhalte wie eine genaue Definition von Ausgangslage und Zielsetzung sind im Einleitungs-Kapitel mit Kapitelnummer 1 aufzuführen. Der Ansatzpunkt dieser Arbeit und die zu beantwortenden Fragen müssen im selben Kapitel ebenfalls vermerkt sein[67].

[67] Hierzu ist als Vorbedingung der Stand der Forschung zu erörtern, wobei dessen Ausformulierung zu einem späteren Zeitpunkt stattfinden kann (siehe Kapitel A.1.2)

A.1.2. P2 Recherche

Theoretische Grundlagen inklusive Stand der Forschung etablieren

Phase: P2 Recherche
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 25. Oktober 2024

Beschrieb:

Den Stand der Forschung sowie die theoretischen Grundlagen gilt es umfassend zu ermitteln und in Kapitel 2 zusammenzufassen. Anhand dem ermittelten Stand kann das Vorgehen abschliessend geplant werden (siehe Kapitel A.1.1).

Fingerprinting- und Identifikations-Möglichkeiten anhand Scans erörtern

Phase: P2 Recherche
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 25. Oktober 2024

Beschrieb:

Die Möglichkeiten, eine Scan-Quelle anhand deren Scans zu identifizieren, sind zu recherchieren. Hierzu gehören Fingerprinting-Mechanismen und weitere potenzielle Identifikatoren. Die ermittelten Möglichkeiten müssen auf die aufgezeichneten Daten anwendbar sein.

Eine Analyse einer Scan-Quelle durch eigens ausgeführte Scans ist nicht Teil dieses Ziels und im Kann-Ziel „Ermittelte Scan-Quellen scannen und weiter auswerten“ in Kapitel A.1.5 (Phase P5 Auswertung) aufgeführt.

Aufzeichnungs- und Analyse-Werkzeuge evaluieren

Phase: P2 Recherche
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 13. November 2024

Beschrieb:

Es sind Werkzeuge, die im Rahmen der Arbeit Verwendung finden können, zu evaluieren. Allfällige Lizenzen und zugehörige Kosten gilt es zusätzlich zu ermitteln, um entsprechende Bestellungen einleiten zu können.

A.1.3. P3 Planung und Design

Prozedur der Scan-Aufzeichnung definieren

Phase: P3 Planung und Design
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 15. November 2024
siehe Kapitel 3.1

Beschrieb:

Der Aufbau der Analyse-Umgebung inklusive dem Verhalten der Scan-Ziele und des Aufzeichnungs-Servers sind zu definieren. Einige Punkte hierzu wurden im Kickoff Meeting (siehe Kapitel A.4.4) festgelegt:

- Die Scan-Ziele verfügen über öffentliche IPv4- und IPv6-Adressen (Dual-Stack-Betrieb)
- Die Scan-Ziele sollen auf sämtliche ICMP-Ping-Anfragen antworten
- TCP-/UDP-Ports auf den Scan-Zielen sind geschlossen
- Jede Verbindung auf eine öffentliche IPv4- oder IPv6-Adresse gilt es als Scan zu betrachten, sofern sie nicht selbst von der Analyse-Umgebung stammen
- Die Kommunikation der Scan-Ziele mit dem Aufzeichnungs- und Control-Server erfolgt über eine VPN-Verbindung
- Der Aufzeichnungs- und Control-Server wird an der BFH betrieben (Betriebssystem und Zugang werden eventuell vom Laborteam des BFH TI Cyber Security Lab bereitgestellt)

Das Betreiben von offenen TCP-/UDP-Ports ist nicht Teil dieses Ziels und im nachfolgenden Kann-Ziel „Services auf Scan-Ziele bereitstellen“ aufgeführt.

DNS-Einträge für die Scan-Ziele sind optional und werden zu Beginn nicht implementiert (siehe Kapitel A.4.4). Weiteres zu DNS siehe Kann-Ziel „DNS-Einträge für Scan-Ziele wählen und implementieren“ in Kapitel A.1.3 (Phase P3 Planung und Design).

Services auf Scan-Ziele bereitstellen

Phase: P3 Planung und Design
Ziel-Typ: Kann-Ziel
Ziel erfüllt: Nicht erfüllt

Beschrieb:

Die Scan-Ziele betreiben eine Auswahl an bekannten Diensten für zum Beispiel DNS, E-Mail (IMAP, SMTP) oder Webserver (HTTP, HTTPS), die im Internet erreichbar sind. Die Auswahl gilt es anhand meist-gescannten TCP-/UDP-Ports oder entsprechender Verbreitung im Internet festzulegen.

Das Betreiben solcher Dienste lenkt die Implementation der Scan-Ziele in Richtung Honeypot, die im Themenantrag dieser Arbeit explizit ausgeschlossen werden.

DNS-Einträge für Scan-Ziele wählen und implementieren

Phase: P3 Planung und Design
Ziel-Typ: Kann-Ziel
Ziel erfüllt: 17. Januar 2025
DNS-Einträge ausschliesslich für IPv6-Adressen der Scan-Ziele erstellt, siehe Kapitel 3.4

Beschrieb:

Gemäss der Besprechung beim Themenantrag (siehe Kapitel A.4.3) gilt es, DNS bezüglich der Erreichbarkeit zu berücksichtigen. Beim Kickoff Meeting (siehe Kapitel A.4.4) wurde die Verwendung von DNS-Einträgen für Scan-Ziele als Kann-Ziel definiert.

Zu Beginn der Arbeit wird der Aufbau der Analyse-Umgebung ohne DNS durchgeführt (siehe Kapitel A.4.4). Beim Einsatz von DNS gilt es einen Vergleich zum Verhalten ohne DNS zu etablieren. Bei der Auswahl von Domänen für DNS-Einträge, ist zu beachten, dass nur bestimmte DNS-Dienstleistungsunternehmen entsprechende Zonen-Informationen bzw. Zonen-Dateien veröffentlichen [35].

Speicherverhalten der Aufzeichnungen festlegen

Phase: P3 Planung und Design
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 15. November 2024
siehe Kapitel 3.1

Beschrieb:

Es ist festzulegen, wie der aufgezeichnete Netzwerkverkehr gelagert und an den Aufzeichnungs- und Control-Server weitergeleitet werden. Im Kickoff Meeting (siehe Kapitel A.4.4) gab es die Idee, den Netzwerkverkehr möglichst zeitnah und direkt mittels nftables-Weiterleitung zu senden. Ziel ist es, die Daten möglichst rasch und zuverlässig von den Scan-Zielen unabhängig zu speichern.

Analyse-Ablauf bestimmen

Phase: P3 Planung und Design
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 6. Dezember 2024
siehe Kapitel 3.1

Beschrieb:

Der Ablauf einer Analyse der aufgezeichneten Daten ist aufgrund der ermittelten Informationen zu definieren. Um die Nachvollziehbarkeit zu gewährleisten, muss bei der Analyse der Daten wiederkehrend dasselbe Resultat erzielt werden.

Automatisierter und erweiterbarer Aufbau eines Scan-Ziels mit Debian Linux

Phase: P3 Planung und Design
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 13. November 2024
Erstellte Skripts und Ansible-Playbooks sind dokumentiert
sowie zugehörige Variablen möglichst zentral definiert (siehe Kapitel D)

Beschrieb:

Die Scan-Ziele sind mit dem Betriebssystem Debian Linux automatisiert zu installieren. Durch den automatisierten Aufbau werden sämtliche Scan-Ziele gleich implementiert. Somit liegt deren Unterschied lediglich in der Wahl der Hosting-Betriebe inklusive der öffentlichen IP-Adressen und geografischer Lage.

Es ist beim Aufbau zu berücksichtigen, dass das TI Cyber Security Lab der BFH die Scan-Ziele nach dieser Arbeit weiter betreiben oder erweitern können muss (siehe Kapitel A.4.4).

Server absichern (Verringern des Manipulations-Risikos)

Phase: P3 Planung und Design
Ziel-Typ: Kann-Ziel
Ziel erfüllt: 20. Dezember 2024
Scan-Ziele mit bestimmten Konfigurationen via Ansible teilweise abgesichert
(SSH-Server-Konfiguration, Partitionierung gemäss CIS Benchmark [351]
und automatisierte Aktualisierungen)
siehe Ansible-Rolle `guadm.hardening` in Kapitel D

Beschrieb:

Um das Risiko einer Manipulation der Scan-Ziele zu verringern, sind diese zusätzlich zu den Schutzmechanismen einer Standardinstallation weiter abzusichern. Hierzu gehören der Betrieb des Betriebssystems möglichst im „Read-Only-Modus“ sowie die Installation eines Host-based Intrusion Detection Systems (HIDS).

Server-Aufbau lokal implementieren und verifizieren

Phase: P3 Planung und Design
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 9. Dezember 2024
siehe Kapitel 3.1

Beschrieb:

Der zuvor definierte Aufbau der Analyse-Umgebung ist lokal zu implementieren und das vorgesehene Verhalten zu verifizieren.

Server-Hosting-Betriebe auswählen

Phase: P3 Planung und Design
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 18. Dezember 2024
siehe Kapitel 3.3

Beschrieb:

Es wird eine Auswahl an Server-Hosting-Betrieben zur Platzierung der Scan-Targets getroffen. Hierfür sind ungefähr 8 Betriebe zusätzlich zur BFH auszuwählen. Die Betriebe müssen öffentliche IPv4- und IPv6-Adressen bereitstellen können und den eingehenden Netzwerkverkehr möglichst wenig filtern. Da die Scan-Ziele voraussichtlich lediglich den entsprechenden Netzwerkverkehr aufzeichnen und weiterleiten, sollten diese keine hohen Anforderungen an die Ressourcen (Prozessor, Arbeitsspeicher und Datenspeicher) haben.

In der Auswahl sind Unternehmen zu berücksichtigen, die mit einer möglichst hohen Wahrscheinlichkeit für den Betrieb eines Servers am Internet verwendet werden (u. a. Amazon, Microsoft, Hetzner).

Die Scan-Ziele sollen voraussichtlich 2-3 Monate (sicher im Januar und Februar 2025) lauffähig sein. Hierfür gilt es auch das vorgegebene Budget zu beachten und die Experten bei Budget-Problemen zu informieren. Die Bezahlung der Hosting-Dienstleistungsunternehmen erfolgt wenn möglich über eine Prepaid-Kreditkarte mit Unterstützung der BFH.

Weitere Überlegungen zur Platzierung der Scan-Targets sind im Protokoll des Kickoff Meetings in Kapitel A.4.4 aufgeführt.

Hosting-Betriebe zu einer möglichst globalen Abdeckung wählen

Phase: P3 Planung und Design
Ziel-Typ: Kann-Ziel
Ziel erfüllt: 10. Dezember 2024
siehe Kapitel 3.3

Beschrieb:

Zusätzlich zum vorherigen Ziel „Server-Hosting-Betriebe auswählen“ gilt es die Hosting-Dienstleistungsunternehmen so zu wählen, dass damit eine möglichst globale Abdeckung erzielt wird. Im Protokoll des Kickoff Meetings in Kapitel A.4.4 werden Länder und Kontinente zur Erwägung genannt.

1. Review besprechen und Feedback übernehmen

Phase: P3 Planung und Design
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 29. November 2024

Beschrieb:

Als Termin für das 1. Review wurde im Kickoff Meeting der 27. November 2024 festgelegt. Hierfür gilt es mindestens 1 Woche zuvor den aktuellsten Stand den Experten zur Verfügung zu stellen. Daraufhin wird zum vereinbarten Zeitpunkt eine Besprechung durchgeführt sowie entsprechendes Feedback festgehalten und übernommen.

Das entsprechende Besprechungs-Protokoll kann unter Kapitel A.4.7 eingesehen werden.

A.1.4. P4 Durchführung

Server bei Hosting-Betrieben aufbauen und verifizieren

Phase: P4 Durchführung
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 27. Dezember 2024
Sämtliche öffentlichen IP-Adressen in Tabelle C.1 sind erreichbar
Verifikation u. a. mittels Test-Netzwerkpaketen zur In-Band Probe Attribution[68]
(mittels IPv4 und IPv6 in Kombination mit ICMP, TCP und UDP)
siehe Quelltext A.1 und Abbildung A.1

Beschrieb:

Die selektionierten Hosting-Dienstleistungsunternehmen aus den Zielen „Server-Hosting-Betriebe auswählen“ und „Hosting-Betriebe zu einer möglichst globalen Abdeckung wählen“ in Kapitel A.1.3 (Phase P3 Planung und Design) werden hierfür verwendet. Sie dienen dem Aufbau der Analyse-Umgebung analog zum Ziel „Server-Aufbau lokal implementieren und verifizieren“. Es gilt daraufhin die hiermit implementierte Analyse-Umgebung zu verifizieren (Erreichbarkeit der Scan-Ziele nach automatisiertem Aufbau und Speicherverhalten im Zusammenhang mit dem Aufzeichnungs- und Control-Server).

```
1 (echo "from␣scapy.all␣import␣*"; echo 'sr(IP(dst="IPV4ADDRESS")/ICMP()/"\x68\
     x74\x74\x70\x3a\x2f\x2f\x74\x68\x69\x73\x69\x73\x61\x74\x65\x73\x74\x2e\
     x65\x78\x61\x6d\x70\x6c\x65\x2e\x63\x6f\x6d\x2f\x2e\x77\x65\x6c\x6c\x2d\
     x6b\x6e\x6f\x77\x6e\x2f\x70\x72\x6f\x62\x69\x6e\x67\x2e\x74\x78\x74\x00",␣
     timeout=3)') | python3
2 echo "from␣scapy.all␣import␣*"; echo 'sr(IPv6(dst="IPV6ADDRESS")/
     ICMPv6EchoRequest()/"\x68\x74\x74\x70...\x00",␣timeout=3)') | python3
3 (echo "from␣scapy.all␣import␣*"; echo  'sr(IP(dst="IPV4ADDRESS")/TCP(dport
     =56789)/"\x68\x74\x74\x70...\x00",␣timeout=3)') | python3
4 (echo "from␣scapy.all␣import␣*"; echo 'sr(IPv6(dst="IPV6ADDRESS")/TCP(sport
     =34567,dport=56789)/"\x68\x74\x74\x70...\x00",␣timeout=3)') | python3
5 (echo "from␣scapy.all␣import␣*"; echo'sr(IP(dst="IPV4ADDRESS")/UDP(dport
     =56789)/"\x68\x74\x74\x70.\x00",␣timeout=3)') | python3
6 (echo "from␣scapy.all␣import␣*"; echo 'sr(IPv6(dst="IPV6ADDRESS")/UDP(sport
     =34567,dport=56789)/"\x68\x74\x74\x70...\x00",␣timeout=3)') | python3
```

Quelltext A.1: Test-Netzwerkpakete: ICMP Echo-Requests, TCP- und UDP-Pakete mit IPv4- und IPv6-Adressen mit Payload `http://thisisatest.example.com/.well-known/probing.txt0x00`
(Payload nur in erster Zeile komplett aufgeführt)

Message	Count
good scanner on 31. [redacted].228 (<???>): unknown via Probe Description URI In-Band ICMP, method: Probe Description URI found in echo request payload at beginning (http://thisisatest.example.com/.well-known/probing.txt)	10
good scanner on 31. [redacted].228 (<???>): unknown via Probe Description URI In-Band TCP, method: Probe Description URI found in TCP payload at beginning (http://thisisatest.example.com/.well-known/probing.txt)	10
good scanner on 31. [redacted].228 (<???>): unknown via Probe Description URI In-Band UDP, method: Probe Description URI found in UDP payload at beginning (http://thisisatest.example.com/.well-known/probing.txt)	10
good scanner on 2a02: [redacted]::b10 (<???>): unknown via Probe Description URI In-Band ICMP, method: Probe Description URI found in echo request payload at beginning (http://thisisatest.example.com/.well-known/probin	9
good scanner on 2a02: [redacted]::b10 (<???>): unknown via Probe Description URI In-Band TCP, method: Probe Description URI found in TCP payload at beginning (http://thisisatest.example.com/.well-known/probing.txt)	9
good scanner on 2a02: [redacted]::b10 (<???>): unknown via Probe Description URI In-Band UDP, method: Probe Description URI found in UDP payload at beginning (http://thisisatest.example.com/.well-known/probing.txt)	9

Abbildung A.1.: Zeek-Notices gemäss Verifikation anhand Quelltext A.1 und IP-Adressen der 10 Scan-Ziele aus Tabelle C.1
(Ein Scan-Ziel verfügt über keine IPv6-Anbindung, Quell-IP-Adressen des Autors werden nicht veröffentlicht)

[68] siehe Kapitel 3.1.4 und D.3.2 (ebenfalls durchgeführt)

Platzierung eines Scan-Ziels im Tor-Netzwerk

Phase: P4 Durchführung
Ziel-Typ: ○ Kann-Ziel
Ziel erfüllt: ✗ Nicht erfüllt

Beschrieb:

Zusätzlich zum vorherigen Ziel „Server bei Hosting-Betrieben aufbauen und verifizieren“ ist ein Scan-Ziel innerhalb des Tor-Netzwerks zu platzieren. Die Ressourcen für diese Komponente werden von der BFH bereitgestellt (siehe Protokoll Kickoff Meeting in Kapitel A.4.4).

Scans kontrolliert erfassen und zentral abspeichern

Phase: P4 Durchführung
Ziel-Typ: ● Muss-Ziel
Ziel erfüllt: ✓ 27. Dezember 2024
siehe Kapitel 3.1.4 und vorhergehende Ziele

Beschrieb:

Die aufgebaute Analyse-Umgebung ermöglicht das kontrollierte erfassen von Scans und leitet entsprechende Aufzeichnungen automatisiert an den Aufzeichnungs- und Control-Server der Analyse-Umgebung weiter. Gemäss Kickoff Meeting (siehe Kapitel A.4.4) kann ein Server der BFH für das Aufzeichnen und Steuern der Analyse-Umgebung genutzt werden.

Analyse-Werkzeuge bereitstellen

Phase: P4 Durchführung
Ziel-Typ: ● Muss-Ziel
Ziel erfüllt: ✓ 3. Januar 2025
Siehe vorhergehende Ziele und Verifikation der Analyse-Umgebung im Internet gemäss Kapitel 3.1.4

Beschrieb:

Entsprechend dem Ziel „Aufzeichnungs- und Analyse-Werkzeuge evaluieren“ in Kapitel A.1.2 (Phase P2 Recherche) evaluierte Werkzeuge stehen zur Verfügung. Die hierfür benötigte Software inklusive allfällige Lizenzen ist installiert und funktionsfähig.

Stichproben-Analysen erfolgreich durchführen

Phase: P4 Durchführung
Ziel-Typ: ● Muss-Ziel
Ziel erfüllt: ✓ 3. Januar 2025
Scans werden detektiert, siehe auch Verifikation der Analyse-Umgebung im Internet gemäss Kapitel 3.1.4

Beschrieb:

Mit der bei Hosting-Betrieben aufgebauten Analyse-Umgebung können erste Scans aufgezeichnet und mit den Analyse-Werkzeugen ausgewertet werden. Gegebenenfalls sind Korrekturen an dem Analyse- oder Aufzeichnungs-Vorgang vorzunehmen, um die geplanten Informationen herauszufiltern.

2. Review besprechen und Feedback übernehmen

Phase: P4 Durchführung
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 15. Januar 2025

Beschrieb:

Als Termin für das 2. Review wurde im Kickoff Meeting der 15. Januar 2025 festgelegt. Hierfür gilt es mindestens 1 Woche zuvor den aktuellsten Stand den Experten zur Verfügung zu stellen. Daraufhin wird zum vereinbarten Zeitpunkt eine Besprechung durchgeführt sowie entsprechendes Feedback festgehalten und übernommen.

Das entsprechende Besprechungs-Protokoll kann unter Kapitel A.4.9 eingesehen werden.

A.1.5. P5 Auswertung

Aufzeichnungen für Analyse vorbereiten (ggf. Daten-Aufbereitung)

Phase: P5 Auswertung
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 19. Februar 2025

Beschrieb:

Zur vertieften Analyse sowie Auswertung der Aufzeichnungen gilt es diese, sofern notwendig, aufzubereiten.

Ermittelte Scan-Quellen scannen und weiter auswerten

Phase: P5 Auswertung
Ziel-Typ: Kann-Ziel
Ziel erfüllt: 21. Februar 2025
Teilweise in Zeek-Skript behandelt
(Out-of-Band Probe Attribution gemäss RFC 9511 [15])
siehe Kapitel D.3

Beschrieb:

Die in dieser Arbeit ermittelten Scan-Quellen sollen aktiv gescannt beziehungsweise überprüft werden, um mehr Informationen darüber zu gewinnen. Eventuell können einzelne Scan-Quellen als Command & Control Server identifiziert werden.

Ergebnisse ermitteln und festhalten

Phase: P5 Auswertung
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 21. Februar 2025
siehe Kapitel 4

Beschrieb:

Entsprechend den Zielen dieser Auswertungs-Phase gilt es aus den gewonnenen Informationen Aussagen zu treffen, die objektiv, nachvollziehbar, reproduzierbar und möglichst genau ausfallen. Die in Kapitel 1.2 definierte Zielsetzung ist mit diesen Aussagen zu erfüllen. Sollten die Zielsetzung oder Teile davon nicht erfüllt werden können, ist dies entsprechend zu auszuweisen und zu begründen.

Ergebnisse veranschaulichen

Phase: P5 Auswertung
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 21. Februar 2025
siehe Kapitel 4

Beschrieb:

Die Ergebnisse und Aussagen aus dem vorherigen Ziel sind grafisch darzustellen, um diese visuell zusammenzufassen.

A.1.6. P6 Abschluss

Bericht vervollständigen und abschliessen

Phase: P6 Abschluss
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 7. März 2025

Beschrieb:

Der Bericht dieser Master Thesis ist vor dem Abgabetermin am 10. März 2025 abzuschliessen. Der Inhalt wird entsprechend ergänzt oder optimiert.

Gemäss Kickoff Meeting (siehe Kapitel A.4.4) ist ein persönlicher Rückblick gewünscht, der in Kapitel 5.3 festzuhalten ist.

Book-Beitrag finalisieren

Phase: P6 Abschluss
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 5. März 2025
siehe `https://bfh.easydocmaker.ch/search/abstract/4272/`

Beschrieb:

Zusammen mit dem Bericht ist ein Book-Beitrag abzugeben, der unter `https://book.bfh.ch` veröffentlicht wird.

Präsentation und Verteidigung vorbereiten

Phase: P6 Abschluss
Ziel-Typ: Muss-Ziel
Ziel erfüllt: 26. März 2025
Ziel im Voraus als erfüllt markiert
(Abgabe dieser Arbeit findet vor der Präsentation statt)

Beschrieb:

Als Termin für die Präsentation inklusive Verteidigung wurde der 26. März 2025 vereinbart (weitere Details siehe Kapitel A.4.5). Hierfür sind entsprechende Vorbereitungen zu treffen. Die Präsentation und Verteidigungsvorbereitung kann nach Abgabe des Berichts (spätestens 10. März 2025) finalisiert werden.

A.2. Arbeitsjournal (Versionsverzeichnis)

Tabelle A.2.: Arbeitsjournal (Versionsverzeichnis)

Datum	Phase	Tätigkeit	Aufwand [h]
18.08.2024	Vorbereitung	Vorbereitung LaTeX Vorlage	3 h
25.08.2024	Vorbereitung	Vorbereitung LaTeX Vorlage	1 h
25.08.2024	Recherche	Literaturrecherche zum Stand der Forschung	2 h
04.09.2024	Vorbereitung	Vorbereitung Kickoff Meeting	1 h
04.09.2024	Recherche	Literaturrecherche zum Stand der Forschung	2 h
06.09.2024	Recherche	Literaturrecherche zum Stand der Forschung	4 h
14.09.2024	Recherche	Literaturrecherche zum Stand der Forschung	3 h
18.09.2024	Vorbereitung	Kickoff Meeting mit Lead- und Co-Experten Prof. Hansjürg Wenger und Prof. Dr. Bruce Nikkel inkl. Protokoll-Einträge (Kapitel A.4 bzw. A.4.4)	3 h
25.09.2024	Recherche	Literaturrecherche zum Stand der Forschung Zugriffsanfrage an IPv6 Hitlist Service `https://ipv6hitlist.github.io` [39]	5 h
28.09.2024	Vorbereitung	Deklaration Verwendung von KI-gestützten Tools (Kapitel B) LaTeX Vorlage: Verzeichnisse in einem Kapitel	2 h
28.09.2024	Recherche	Literaturrecherche zum Stand der Forschung	2 h
05.10.2024	Vorbereitung	Ausarbeitung Vorgehen inkl. Muss- und Kann-Ziele (Arbeitspakete) zu den Phasen „P1 Vorbereitung“, „P2 Recherche“ und „P3 Planung und Design“	4 h
06.10.2024	Vorbereitung	Ausarbeitung Vorgehen inkl. Muss- und Kann-Ziele (Arbeitspakete) zu den Phasen „P4 Durchführung“, „P5 Auswertung“ und „P6 Abschluss“	2 h
16.10.2024	Vorbereitung	Einleitung inklusive Ausgangslage und Zielsetzung verfassen	4 h
23.10.2024	Recherche	Ausformulierung Stand der Forschung	9 h
25.10.2024	Recherche	Fingerprinting- und Identifikationsmöglichkeiten Aufzeichnungs- und Analyse-Werkzeuge evaluieren (Arkime, Zeek, Malcolm) Troubleshooting Port-Mirroring mittels nftables	8 h
26.10.2024	Recherche	Aufzeichnungs- und Analyse-Werkzeuge evaluieren (Malcolm) Troubleshooting Installation NetBird Troubleshooting Port-Mirroring mittels nftables	4 h

Arbeitsjournal (Versionsverzeichnis) Fortsetzung

Datum	Phase	Tätigkeit	Aufwand [h]
30.10.2024	Recherche	Aufzeichnungs- und Analyse-Werkzeuge evaluieren: Prototyp-Versuche: WireGuard-Verbindungen mit innernet und Port-Mirroring mit ERSPAN Auseinandersetzung MTU	10 h
01.11.2024	Planung und Design	Prototyp Troubleshooting und Anpassung MTU und ERSPAN auf Interfaces inklusive innernet und Malcolm	8 h
02.11.2024	Planung und Design	Prototyp: ERSPAN-Setup-Skript, Troubleshooting und Konfiguration Malcolm- und Hedgehog-Linux-Sensor-Instanz	6 h
06.11.2024	Planung und Design	Prototyp: ERSPAN-Traffic-Entkapslung (Skript zur Erkennung von ERSPAN-Traffic sowie Erstellung und Konfiguration entsprechender Interfaces) Installationsdoku und Skripts (Kapitel C und D)	10 h
08.11.2024	Planung und Design	Installationsdokumentation und Ausarbeitung Skripts (MTU-Konfiguration, Unterbindung eines Loops wenn Port-Mirroring-Daten über Port, Error-Handling bei ERSPAN-Neuaufbau)	10 h
09.11.2024	Recherche	Autoinstall Debian	2 h
09.11.2024	Planung und Design	Prototyp: Autoinstall Debian inkl. Generierung ISO (siehe Kapitel D)	6 h
13.11.2024	Planung und Design	Automatisierter Aufbau eines Scan-Ziels mittels Ansible (Playbook und Rollen schreiben, siehe Kapitel D)	12 h
15.11.2024	Planung und Design	Dokumentation Install. Scan-Ziel und VPN-Client Anfrage an Donatello Gallucci bzgl. Aufbau an BFH Modifikation Handhabung bei „Keepalive“-Fehler ERSPAN-Skripts (siehe Kapitel D.1) Prozedur der Scan-Aufzeichnung definieren und Speicherverhalten der Aufzeichnungen festlegen (siehe Kapitel 3.1)	9 h
16.11.2024	Planung und Design	Vorbereitung und Versand Master Thesis für 1. Review Analyse-Ablauf bestimmen	4 h
20.11.2024	Planung und Design	Prepaid-Kreditkarte organisieren Upgrades Malcolm [352] sowie Upgrade-Versuch und Neuinstallation Hedgehog Linux (Kapitel C.2) [353] Zeek-Script erstellen für Analyse-Ablauf [164]	10 h

Arbeitsjournal (Versionsverzeichnis) Fortsetzung

Datum	Phase	Tätigkeit	Aufwand [h]
22.11.2024	Planung und Design	Zeek-Script erstellen für Analyse-Ablauf [164]: Out-of-Band Probe Attribution [15] und Start Implementation In-Band Probe Attribution [15] für URL und E-Mail in ICMPv4/ICMPv6 Echo-Request-Paketen	9 h
23.11.2024	Planung und Design	Zeek-Script: In-Band Attribution [15] für URL, E-Mail, Telefon in TCP-/UDP-Payload, IPv6 PadN Data [162]	8 h
27.11.2024	Planung und Design	Korrektur aktuelle Fassung 1. Review (siehe Kapitel A.4.7) Installation Aufzeichnungs- und Analyse-Server für globalen Aufbau vor Ort an der BFH (siehe Kapitel A.4.8)	8 h
29.11.2024	Planung und Design	Erweiterung Ansible-Hardening-Rolle um Einrichtung automatischer Updates Erweiterung Zeek-Skript um ZMap-Erkennung Einführungs-Kapitel: Lieferobjekte ausführen Installation Zeek-Skript (siehe Kapitel C.2.4) Start Anreicherung Tabelle mit bekannten Scanner	8 h
30.11.2024	Planung und Design	Anreicherung Tabelle mit bekannten Adressen Erweiterung Zeek-Skript um Erkennung mittels Domain-Adressen (nicht komplette FQDN) Testing Zeek-Skript	5 h
06.12.2024	Planung und Design	Troubleshooting / Versuch zur Ratenlimitierung auf Interfaces mit TC Anreicherung Tabelle mit Tor Exit Nodes Nachfrage bei Donatello zu Stand Pendenzen Notizen für Handover nach Arbeitsabschluss (siehe Kapitel A.4.10)	7 h
07.12.2024	Planung und Design	Fix in ERSPAN-Skript (ausgehende Pakete auch replizieren, am 23.12.2024 wieder ausgeschaltet aufgrund Sensor-Belastung) Umgestaltung Skript-Auflistung in Kapitel D zu Skripts-Übersicht und Aufführung nur noch relevanter Ausschnitte Start Dokumentation der Verifikation mittels Test-Netzwerkpaketen u.a. anhand Scapy	6 h
09.12.2024	Planung und Design	Verifikation mittels Test-Netzwerkpaketen u.a. anhand Scapy (siehe Tabellen 3.1 und D.2) Passwort-Wechsel bei Malcolm-Instanz und Sensor/Appliance mit Hedgehog Linux im BFH Cyberlab Cyberlab-SSH-Public-Keys in bestehenden Code integrieren	7 h

Arbeitsjournal (Versionsverzeichnis) Fortsetzung

Datum	Phase	Tätigkeit	Aufwand [h]
09.12.2024	Abschluss	BFH Book: Tool-Zugang und -Anleitung prüfen Termin für Fotoshooting vereinbaren	1 h
10.12.2024	Planung und Design	Hosting-Betrieb-Recherche und -Wahl Konfiguration der VMs im Cyber-Security-Lab abschliessen Test-Verbindung und Verifikation mit lokalem Scan-Ziel zur Cyber-Security-Lab-Infrastruktur	8 h
11.12.2024	Durchführung	Konfig. Scan-Ziel `st001` in Cyber-Security-Lab Registrierung bei Hosting-Betrieben Aufbau Scan-Ziele `st002` , `st003` [354] und `st004` [355] (siehe Tabelle C.1 in Kapitel C.4)	8 h
13.12.2024	Planung und Design	Ansible-Rolle `guadm.hardening`: Existenz der Disk-Volume-Gruppe vor Anpassung prüfen Anpassung Zeek-Skript: Prüfen, ob String „contact“ (case insensitive) in Antwort bei Out-of-Band Probe Attribution vorkommt, um False-Positives zu vermeiden	1 h
13.12.2024	Durchführung	Registrierung bei Hosting-Betrieben Identitätsprüfung für `st005` Disk-Upload-Versuche zu Azure und Netzwerk-Konfiguration [356, 357] für `st006` Aufbau Scan-Ziele `st006` und `st007` Support-Tickets für IPv6 zu `st007` und `st009` (Scan-Ziele siehe Tabelle C.1 in Kapitel C.4)	8 h
14.12.2024	Durchführung	Registrierung bei Hosting-Betrieben Aufbau Scan-Ziele `st008` und `st009` Überprüfung Dashboards und Zeek	6 h
18.12.2024	Durchführung	Anfrage bei Alibaba für China abgelehnt, Aufbau Scan-Ziel `st005` in Hong Kong Anpassung ERSPAN-Entkapslung-Skript (Sub-Interface-Bezeichnungen, Loop-Handling bei Sender-Erkennung) Analyse Zeek Paket-Drops [226] Anpassung Zeek-Skript (Ergebnisse beim Prüfen der Out-of-Band Probe Attribution in Variable zwischenspeichern und Adressen nicht jedes mal testen) Anreicherung bekannter Domänen aus Zeek-Logs (siehe Kapitel 3.1.3) Aufbau Scan-Ziel `st010`	10 h

Arbeitsjournal (Versionsverzeichnis) Fortsetzung

Datum	Phase	Tätigkeit	Aufwand [h]
20.12.2024	Durchführung	Anpassung Ansible-Rolle `guadm.hardening`: ufw nicht zurücksetzen wenn bereits installiert ist Anpassung ERSPAN-Skript: Keepalive-Mechanismus auch für Hosts ohne IPv6 Sensor mit Hedgehog Linux: Erhöhung der CPU-Kerne von 8 auf 16 aufgrund „Dropped Packets“ in Zeek und Auslastung durch Zeek-Skript Verifikation der Server bei Hosting-Betrieben (Ausnahme IPv6-Adresse von `st010` 🇬🇧, da Aufschaltung durch Support noch ausstehend) Kleines Ansible-Playbook (< 30 Zeilen) zur Auswertung des derzeitigen Ressourcenverbrauchs auf den Hosts (für manuelle Überprüfung) Ausprobieren von OpenSearch-Visualisierungen	8 h
23.12.2024	Durchführung	Anpassung ERSPAN-Skript auf Scan-Zielen, um Last auf Sensor mit Hedgehog Linux zu Verringern (Dieser meldet nach CPU-Ressourcenerhöhung immer noch „Dropped Packets“): Nur noch eingehende Pakete replizieren, Pakete an Broadcast, Link-Local-IPv6-Adressen und Link-Local-Multicast-IP-Adressen nicht replizieren Erneute Verifikation der Scan-Ziele nach ERSPAN-Skript-Update Status-Mail an Experten verfassen	4 h
27.12.2024	Durchführung	Zeek-Dateiextrahierung aufgrund Ressourcenansprüchen deaktivieren Verifikation IPv6-Adresse bei Scan-Ziel `st010` 🇬🇧 (Support hat IPv6-Adresse konfiguriert) Troubleshooting mit OpenSearch-Visualisierungen	4 h
30.12.2024	Durchführung	OpenSearch-Visualisierungen und Arkime-Views Recherche GeoIP-Unterschiede zwischen Einträgen von Arkime und Zeek Anreicherung bekannter Domänen aus Zeek-Logs (siehe Kapitel 3.1.3) Anpassung Zeek-Skript: Input-Streams nicht schliessen, um Dateiänderungen zu erfahren [358]	5 h
02.01.2025	Durchführung	Troubleshooting Zeek-Ressourcenverbrauch und Vergrösserung Paket-Puffergrösse von `67108864` auf `536870912` Bytes (64 zu 512 MB, siehe Ende von Kapitel C.2.3) [141, 359, 360] Tägliche Aktualisierung bekannter Scan-Quellen mit Skript `loaddata.sh` (siehe Tabelle D.1 in Kapitel D) auf Sensor mit Hedgehog Linux gemäss Kapitel 3.1.3 und D.3.1 OpenSearch-Dashboard „Scanner Detection“ erstellt	4 h

Arbeitsjournal (Versionsverzeichnis) Fortsetzung

Datum	Phase	Tätigkeit	Aufwand [h]
03.01.2025	Durchführung	Kontrolle tägliche Aktualisierung bekannter Scan-Quellen Anpassung Visualisierungen für besseres Filtern nach Detektionsmethode Ausarbeitung Kapitel 3.5 bezüglich Benutzeroberflächen Stichproben-Analyse	6 h
04.01.2025	Durchführung	Troubleshooting: Standortangaben aus Arkime mit Quell-IP-Adressen von OpenSearch Dashboards Korrektur aktuelle Fassung Vorbereitung Master Thesis für 2. Review	7 h
05.01.2025	Durchführung	Versand Master Thesis Version für 2. Review	-
08.01.2025	Durchführung	README-Datei für `opensearchdashboards-objects` Daten-Verfügbarkeit detektierter Scan-Quellen prüfen, z.B. ob diese eine Suchmaschine anbieten Analysieren gescannter Scan-Ziel-IP-Adressen auf Censys, Shodan, ONYPHE [290], Driftnet [178, 249], LeakIX [297] und Google LaTeX-Vorlage: Datumsangaben bei Online-Quellen (nicht nur „besucht am …“)	7 h
10.01.2025	Durchführung	Anfügen der Flaggen an Scan-Ziele in Text zur Übersicht Arbeit an Kapitel 4 (Analyse-Verhalten, Zeitraum, Ausreisser, Allgemeine Merkmale) Erweiterung Kapitel 3.1.3 um Info zu Datenverarbeitung in Malcolm Vor-Analyse Dashboards „Connection“ und „Scanner Detection“	9 h
15.01.2025	Durchführung	Korrektur aktuelle Fassung 2. Review (siehe Kapitel A.4.9)	4 h

Arbeitsjournal (Versionsverzeichnis) Fortsetzung

Datum	Phase	Tätigkeit	Aufwand [h]
17.01.2025	Durchführung	Ergänzungen in Grundlagen-Kapitel 2 bezüglich Privatsphäre in Whois-Einträgen und weiteren IP-Geolocation-Dienstleistungsunternehmen `st002` 🇳🇬 und `st003` 🇩🇪 haben nach automatisiertem Updates-Reboot jeweils am 13. und 14. Januar um 02:00 Uhr UTC das ERSPAN-Skript nicht gestartet (Timeout beim Warten auf Swap- und VPN-Client-Netzwerk-Interface) → Services wieder gestartet Organisation Domain `pinelair.com` mit DNS-AAAA-Einträgen zu IPv6-Adressen der Scan-Ziele Fehldetektion bei Out-of-Band Probe Attribution: Manuelle Prüfung von URIs und URI-Rewrite-Test mit nginx Weitere Betrachtungen des IPv4-Identifikations-Feld im Zusammenhang mit ZMap	8 h
18.01.2025	Durchführung	Überprüfung IPv6-Hitlisten von Gasser et al., IPv6 RIPE-Atlas-Probe und ermittelte FQDNs gemäss Quelltext 3.1 Erweiterung Dashboard mit IPv4 vs IPv6 mit einmaliger Zählung pro Quell-IP und Betrachtung Detektionen von Scan-Quellen mit mehreren Methoden Anpassung Arkime-Views entsprechend Filter in OpenSearch Dashboards	4 h
22.01.2025	Durchführung	Ausführungen zu Mehrfachdetektionen, System-Updates, angewendeten Filter und betrachteter Verbindungen in Kapitel 4 Analyse DNS-Einträge zu Debian-Repositories inklusive Passive DNS	6 h
29.01.2025	Durchführung	Vor-Analyse IPv6-Scans inklusive 6to4-Verbindungen in Kapitel 4.2 Betrachtung System-Logs des Sensors mit Hedgehog Linux und Durchführen entsprechender Konfigurationen („sendmail“-Deaktivierung, System-Gruppen anlegen) Vor-Analyse JA4+ in Kapitel 4.6	9 h
31.01.2025	Durchführung	Ausarbeitung und Unterkapitel-Gliederung in Kapitel 4 Entwurf Kapitel 4.10 mit persönlicher Interpretation Weitere Ausführung zu geografischen Merkmalen in Kapitel 3.5.2 Hedgehog Linux Service-Restart (Zeek-Worker-Prozesse wechselten in Status „D disk sleep (uninterruptible)“, ca. 2h Downtime)	6 h

Arbeitsjournal (Versionsverzeichnis) Fortsetzung

Datum	Phase	Tätigkeit	Aufwand [h]
01.02.2025	Auswertung	Betrachtung JA4+-Fingerprints (Kapitel 4.6) OpenSearch Dashboards-Reporting (Troubleshooting Limit im Zusammenhang mit Felder `tcp.ja4l` und `tcp.ja4t`)	7 h
06.02.2025	Auswertung	Weitere Vor-Analysen und Ausführungen zu JA4+ in Kapitel 4.6 Vorbereitung und Ausprobieren Diagramme in LaTeX zur Ergebnis-Darstellung Weitere Ausführungen zur Intention der Scan-Quellen in Kapitel 4.7	10 h
07.02.2025	Auswertung	Verifikation und Korrektur Filter in OpenSearch Dashboards und Arkime Vor-Analyse und Ausführungen zu Standortangaben in Kapitel 4.8 sowie Ausführungen in Kapitel 4.6 und 4.10	7 h
12.02.2025	Auswertung	Information an Experten zur Anpassung des auszuwertenden Zeitraums (Verkürzung um zwei bis drei Tage, siehe Nachtrag in Kapitel A.4.9) Erweiterungen der Analyse in Kapitel 4 zu Debian Aktualisierungsvorgängen, Microsoft Azure cloud-init, Probe Attribution gemäss RFC 9511, ZMap, IPv6 und allgemeinen Verbindungsmerkmalen Analyse kontaktierte Ports (Top Ziel-Ports und mehr zu den Ports 8728 und 22)	9 h
14.02.2025	Auswertung	Analyse kontaktierte Ports (Ports 23, 80, 34567, 8080, 443, 53 und 123) sowie ICMPv4-Echo-Requests in Kapitel 4.4 Analyse JA4+ in Kapitel 4.6 (JA4T-Abgleich mit JA4+-Datenbank [110] gestartet) Analyse und Erweiterungen zur Intention der Scan-Quellen Ausführungen und Korrektur-Filter zur Mehrfachdetektionen in Kapitel 4.1.6 (IP-Adressen, die aufgrund IP und Subnetz bekannt sind, werden doppelt gezählt) Start Korrektur ermittelter Werte in Kapitel 4 nach Korrektur-Filter	13 h
15.02.2025	Auswertung	Korrektur ermittelter Werte in Kapitel 4 nach Anwendung Korrektur-Filter aus Kapitel 4.1.6 inkl. Abbildungen und Tabellen Analyse zu Standortangaben mittels Arkime in Kapitel 4.8	7 h

Arbeitsjournal (Versionsverzeichnis) Fortsetzung

Datum	Phase	Tätigkeit	Aufwand [h]
19.02.2025	Auswertung	JA4T-Fingerprint-Analyse in Kapitel 4.6 Analyse meist kontaktierte Ports zwischen einzelnen Scan-Zielen in Kapitel 4.5 Analyse zu öffentlichen Informationen zu Scan-Zielen Ausführungen zur persönlichen Interpretation in Kapitel 4.10 bis und mit dem Vergleich mit der Arbeit von Heo und Shin Stopp und Abbau aller Scan-Ziele ausser `st001`	9 h
21.02.2025	Auswertung	Weitere Ausführungen zur persönlichen Interpretation und Vergleich mit Arbeiten von Heo und Shin [22] und Durumeric et al. [27] in Kapitel 4.10 Anpassung und Test Zeek-Skript gemäss Kapitel 4.1.6 (siehe Zeilen 27, 28 und 32 in Quelltext D.3 in Kapitel D.3) Implementation der regelmässigen Bereinigung bekannter Einträge entsprechend Kapitel 4.10.5 und C.2.4 Start Vorbereitung Handover in Kapitel A.4.10	8 h
22.02.2025	Auswertung	Vorbereitung Handover in Kapitel A.4.10 Anfügen von Screenshots zum „Scanner Detection“-Dashboard im Anhang in Kapitel E.2	3 h
26.02.2025	Abschluss	Entwurf Abschluss-Kapitel 5 Festhalten der Meilensteine in Kapitel A.3 Entwurf Abstract und Book-Beitrag	9 h
28.02.2025	Abschluss	Korrektur und Erweiterungen in Abschluss-Kapitel 5 und Book-Beitrag Einreichung Book-Beitrag Review und Korrektur der Arbeit ohne Anhang	8 h
01.03.2025	Abschluss	Review und Korrektur des Anhangs	4 h
05.03.2025	Abschluss	Anpassung und Abgabe Book-Beitrag Erneutes Review und Korrektur der Arbeit (u. a. Dezimaltrennzeichen) Vorbereitung Abgabe inklusive Code bzw. Skripts	8 h
07.03.2025	Abschluss	Abgabe Arbeit inklusive Code bzw. Skripts	-
		Total Aufwand	**439 h**

Tabelle A.3.: Aufwand pro Phase (mit Vergleich zu Tabelle A.1 in Kapitel A.1)

Phase	Geplant [h]	Effektiv [h]
P1 Vorbereitung	20	20
P2 Recherche	40	51
P3 Planung und Design	120	142
P4 Durchführung	100	123
P5 Auswertung	60	73
P6 Abschluss	20	30
Total Aufwand	360	439

A.3. Meilensteine

Die nachfolgende Tabelle zeigt zur Übersicht die Meilensteine dieser Arbeit.

Tabelle A.4.: Meilensteine

Datum	Meilenstein	Referenzkapitel
14. August 2024	Themenantrag zugelassen	A.4.3
18. September 2024	Kickoff Meeting	A.4.4
25. Oktober 2024	Grundlagen etabliert	A.1.2
15. November 2024	Design Analyse-Umgebung festgelegt	A.1.3
27. November 2024	Besprechung 1. Review	A.4.7
27. November 2024	Installation Aufzeichnungs- und Analyse-Server vor Ort	A.4.8
27. Dezember 2024	Analyse-Umgebung inklusive Scan-Ziele aufgebaut und verifiziert (Start Analyse-Zeitraum)	A.1.4
15. Januar 2025	Besprechung 2. Review	A.4.9
10. Februar 2025	Stopp Analyse-Zeitraum	A.4.9
21. Februar 2025	Ergebnisse ermittelt und festgehalten	A.1.5
7. März 2025	Abgabe Arbeit	A.1.6
26. März 2025	Präsentation und Verteidigung	A.1.6
26. März 2025	Übergabe / Handover der Analyse-Umgebung	A.4.10

A.4. Protokolle

A.4.1. Diskussion Themenantrag (5. Juni 2024)

Info

Teilnehmende:
- Hansjürg Wenger
- Donatello Gallucci
- Kevin Georg Schrag
- Mauro Guadagnini

Datum: 5. Juni 2024, 14:30 Uhr

Dauer: 30 min

Ort: Teams-Meeting

Protokoll

- Anfrage von Mauro Guadagnini zuvor per E-Mail gestellt, ob potenzielles Thema für eine Master Thesis im Bereich Cyber Security vorhanden ist
- Hansjürg nennt Idee mit Scans im Internet
  - System im Internet erhält schnell Scans (von Forschern, Angreifern, Suchmaschinen, etc.)
  - Hauptfrage: Wie viele Scans sind bösartig?
  - Wie können Scan-Akteure klassiert werden?
  - Wie finden Scan-Akteure ihre Ziele?
  - Scan-Target bauen und bei verschiedenen geografischen Orten platzieren
  - Brainstorming zur Identifizierung von Scans, Aufbau eines „Sensor-Netzes“
- Themensponsor: Berner Fachhochschule TI Cyber Security Lab
- Master Thesis mit diesem Thema als Forschungsarbeit durchführen
- Hansjürg Wenger und Bruce Nikkel als Expertenwunsch in Themenantrag hinterlegen

Pendenzen

- Hansjürg
  - Sucht potenziell verwandte Bachelor Thesis heraus (`https://bfh.easydocmaker.ch/search/abstract/1088/`) ✔ 5. Juni 2024
  - Gegenlesen von zukünftig verfasstem Themenantrag ✔ 12. Juni 2024
- Mauro
  - Entscheid an Hansjürg mitteilen, ob Thema für Master Thesis genommen wird ✔ 9. Juni 2024
  - Entwurf Themenantrag an Hansjürg senden ✔ 9. Juni 2024

A.4.2. Diskussion Themenantrag (12. Juni 2024)

Info

Teilnehmende:	Datum: 12. Juni 2024, 14:30 Uhr
Hansjürg Wenger	Dauer: 30 min
Donatello Gallucci	Ort: Teams-Meeting
Kevin Georg Schrag	
Mauro Guadagnini	

Protokoll

- Entscheid (Thema wird für Master Thesis genommen) zuvor per E-Mail mitgeteilt
- Kurz-Review Themenantrag
- Hansjürg hat OK zur Einreichung des besprochenen Themenantrags erteilt

A.4.3. Präsentation Themenantrag (14. August 2024)

Info

Teilnehmende:	Datum: 14. August 2024, 10:45 Uhr
Arno Schmidhauser	Dauer: 45 min
Gerhard Hassenstein	Ort: Teams-Meeting
Rolf Lanz	
Mauro Guadagnini	

Protokoll

- Präsentation Themenantrag durchgeführt
- Zulassungsentscheid: Zugelassen
- Feedback
  - DNS bezüglich Erreichbarkeit der Ziel-Server berücksichtigen
  - Bewertungskriterien Forschungsarbeit berücksichtigen
  - Ausgangslage genau definieren inklusive Stand der Forschung, Nutzen der Arbeit für Forschung, Beantwortung welcher Fragen sowie Startpunkt der Arbeit (wo setzt Arbeit an)
  - Eventuell Empfehlungen am Ende zum allgemeinen Betrieb einer Ressource im Internet verfassen
  - Publikation Arbeit für Community sowie Budget mit Hansjürg Wenger besprechen

A.4.4. Kickoff Meeting (18. September 2024)

Info

Teilnehmende:
- Hansjürg Wenger
- Bruce Nikkel
- Mauro Guadagnini

Datum: 18. September 2024, 8:30 Uhr

Dauer: 90 min

Ort: Biel, SIPBB, Raum S.247

Protokoll

- Organisatorisches
  - Sitzungsprotokolle und Versionsverzeichnis in Anhang (gehört zu Projektmanagement)
  - Versionsverzeichnis / Arbeitsjournal darf im Laufe der Arbeit detaillierter ausfallen
  - Termine definiert
    - Besprechung 1. Review: 27. November 2024, 8:30 Uhr
    - Besprechung 2. Review: 15. Januar 2025, 8:30 Uhr
    - Mittwochs beste Erreichbarkeit, Bruce und Hansjürg sind dann jeweils in Biel vor Ort
  - Kontakt jeweils per E-Mail und bei Bedarf als Teams-Sitzung
  - Abwesenheiten
    - 7. bis 13. Oktober 2024: Mauro abwesend
    - 22. Dezember 2024 bis 4. Januar 2025: Niemand erreichbar
    - Ab 1. Februar 2025: Bruce eventuell Sabbatical für 6 Monate, per E-Mail sporadisch erreichbar
    - 3. bis 9. Februar 2025: Hansjürg abwesend
  - Zugang an Bruce und Hansjürg zu privatem Github-Repository für Master Thesis erteilt (der aktuellste Stand der Arbeit kann dort eingesehen werden)
  - Publikation der Arbeit für Community eventuell unter `https://arbor.bfh.ch/`, sicher aber Link zu Arbeit in BFH-Book-Beitrag (Hansjürg klärt genauer ab)
  - KI-Verwendung entsprechend ausweisen (bisher nur zum Finden weiterer Quellen als Suchmaschine und nicht für die Generierung von Text oder Grafiken verwendet, derzeit keine weitere KI-Verwendung geplant)
  - Budget für Server-Hosting
    - 2-3 Monate (sicher Januar und Februar)
    - Prepaid Kreditkarte zur selbstständigen Verwendung wird organisiert

- Master Thesis
  - Ziele
    - In Bericht festhalten (Muss- und Kann-Ziele)
    - Muss-Ziele (unvollständig):
      - Scan-Target automatisiert aufbaubar mit Debian Linux Betriebssystem und einfachen Erweiterungsmöglichkeiten (z.B. bei Skripts simple Anpassbar- und Wiederverwendbarkeit berücksichtigen)
      - Erreichbarkeit Scan-Targets im Internet nach Aufbau testen
      - Scan-Target auf niedrigem Interaktionslevel behalten (Auf Pings antworten, aber Ports geschlossen halten)
    - Kann-Ziele (unvollständig):
      - Scan-Target gehärtet und möglichst „read-only“ mit Host-based Intrusion Detection System (HIDS) betreiben
      - Scan-Target in Tor-Netzwerk platzieren (gehostet bei BFH)
      - DNS-Einträge für Scan-Targets festlegen (zu Beginn ohne DNS arbeiten und bei DNS-Verwendung prüfen welche TLD Provider ihre Zonenfiles veröffentlichen (machen nicht alle [35]))
      - Services wie DNS, HTTP, HTTPS, SSH, SNMP, SMTP, IMAP, POP3, etc. auf Scan-Target hosten (eher Honeypot)
      - Ermittelte Scan-Quellen scannen und auswerten (vielleicht findet man Command & Control Server)
  - Analyse-Umgebung bzw. Scan-Targets sollen erweiterbar und weiter betreibbar sein („darauf weiter aufbauen können“)
  - Zusätzlich auch Server von BFH nutzbar (Internetanbindung über Provider init7 und NorthC)
  - Platzierung Scan-Targets
    - Anzahl: Ungefähr 8 Scan-Targets plus BFH-Server
    - Interessante Länder / Kontinente (zu erwägen): Ukraine, Taiwan, USA, Kanada, Schweiz, Australien, Japan, China, Brasilien, Deutschland, Russland, Indien, Korea, Afrika
    - Interessante Hoster (zu erwägen): Amazon, Microsoft, Hetzner, evtl. „Shady Provider“
    - Traffic-Filtering von Hoster beachten und testen (bestenfalls Scan-Target mit Public-IP-Adressen ohne Schutz davor platzieren)
    - Targets möglichst repräsentativ für globalen Aufbau wählen und klar in Arbeit ausweisen (Wahl von „Shady Provider“ somit mit 8 Targets eher unwahrscheinlich)
    - Targets über VPN mit Aufzeichnungs- und Control-Server verbinden (z.B. mit Tailscale oder netbird.io)
  - Platzierung Scan-Aufzeichnungen und zentrale Steuerung der Targets auf Server der BFH
  - Speichern der Aufzeichnungen zum Beispiel „live“ mittels nftables-Weiterleitung über einen VPN-Tunnel direkt auf den Aufzeichnungs-Server oder anhand gesplitteter Wireshark-Aufzeichnungen (in Design-Phase entscheiden)

- Scan-Targets im Dual Stack betreiben (gleichzeitig öffentliche IPv4- und IPv6-Adressen)
- IP-Adressen von Scan-Targets, die zuvor für andere Services verwendet wurden (z.B. Webseite) können Daten beeinflussen (nicht weiter zu beachten, aber zu erwähnen)
- IP-Hitlisten können auch generiert werden, indem man einen NTP-Server in einem Pool bereitstellt und die IP-Adressen der NTP-Clients speichert
- Jede Anfrage von Aussen auf Scan-Target als Scan betrachten
- Ergebnisse müssen nicht anonymisiert werden, da die Daten öffentlich einsehbar sind
- Von RFC 9511 erzählt
- Erster Entwurf von Dokument-Layout und Struktur gezeigt
  - Abstract zu Beginn so OK
  - Persönlicher Rückblick von Hansjürg nach Fazit in Arbeit gewünscht
  - Zitierstil nach IEEE in Ordnung
- Klare Abgrenzung in Dokumentation festhalten (was wird nicht angeschaut? Primär kein DNS- und Server-Hosting)
- Auswertungsvorgang klar dokumentieren
- Bei Präsentation 2-3 aussagekräftige Folien mit ermittelten Statistiken zeigen

Pendenzen

- Hansjürg
  - Vorgehen einholen bzgl. Publikation der Arbeit für Community
    ✔ 8. November 2024
    Rechte an Arbeit gehören Mauro Guadagnini, Publikation selber überlassen
    siehe Kapitel A.4.6
  - Organisation Prepaid-Kreditkarte für Server-Miete (ca. 300 Franken)
    ✘ 8. November 2024
    Karte selber organisieren und Quittung senden
    siehe Kapitel A.4.6
- Mauro
  - Phasen gemäss Zeitplanung in Versionsverzeichnis aufnehmen und Stunden entsprechend Phasen summieren
    ✔ 18. September 2024
  - Protokoll in Projektmanagement-Teil in Anhang dieser Arbeit
    ✔ 18. September 2024
  - Info an Bruce und Hansjürg zum Finden des aktuellsten Stands der Arbeit per E-Mail inkl. Einladung zu Github-Repository für Master Thesis
    ✔ 18. September 2024
  - Info an Hansjürg betreffend Bedarf an Server der BFH (Aufzeichnungs- und Control-Server inkl. VPN-Kommunikation) ✔ 8. November 2024
  - Im November / Dezember bezüglich Hoster und Budget informieren und melden, falls Budget nicht reicht
    ✔ 23. Dezember 2024
  - Terminfindung für Master Thesis Präsentation im Zeitraum 17. März bis 5. April 2025 ✔ 3. Oktober 2024

A.4.5. Terminfindung Präsentation (Mail-Austausch, 18. September bis 3. Oktober 2024)

Info

Teilnehmende:

Hansjürg Wenger

Bruce Nikkel

Rolf Lanz

Mauro Guadagnini

Datum: 18. September bis 3. Oktober 2024

Mail-Austausch

Protokoll

- Abstimmung mit Terminvorschlägen für Präsentation an Bruce und Hansjürg gesendet (19.3.25, 26.3.25 oder 2.4.25, jeweils um 10 Uhr)
- Bruce kann aufgrund Sabbatical ab Februar 2025 nicht an der Präsentation teilnehmen und schlägt stattdessen Kevin Georg Schrag vor
- Hansjürg präferiert eine Person mit Erfahrung in MAS-Thesis-Arbeiten (lehnt somit den Vorschlag von Bruce ab) und fragt bei Rolf Lanz zur Stellvertretung von Bruce nach
- Rolf wird ebenfalls zur Termin-Abstimmung eingeladen
- Bruce, Hansjürg und Rolf haben an Termin-Abstimmung teilgenommen
  - Bruce hat abgesagt
  - Hansjürg und Rolf haben sämtlichen Vorschlägen zugesagt
  - Rolf bevorzugt den Termin am 26. März 2025
  - Termineinladung für Präsentation am 26. März 2025 um 10 Uhr bis 11:30 Uhr an Hansjürg und Rolf versandt
  - Rolf nach Absprache zu 2. Review eingeladen

Pendenzen

- Hansjürg
  - Raum für Präsentation reservieren ✔ 23. Oktober 2024
- Mauro
  - Zwischenstand-Versand vor Review-Terminen an Hansjürg, Bruce und Rolf
    - 1. Review ✔ 16. November 2024
    - 2. Review ✔ 5. Januar 2025

A.4.6. Fragen und Status-Update (Mail-Austausch, 8. bis 11. November 2024)

Info

Teilnehmende:

Hansjürg Wenger

Bruce Nikkel

Mauro Guadagnini

Datum: 8. bis 11. November 2024

Mail-Austausch

Protokoll

- Status mitgeteilt:
  - Port-Mirroring mittels NetBird und nftables ohne Erfolg, dafür mit innernet und ERSPAN
  - Malcolm und Sensor (Hedgehog Linux) zur Aufzeichnung und Analyse als geeignete Lösung genannt inkl. Systemanforderungen von Malcolm (8 CPU Kerne und 16 GB RAM) [361]
  - Abbildung ähnlich zu Abbildung 3.3 ohne MTU-Werte angefügt
- Fragen und Antworten von Hansjürg:
  - **Frage**: Betrieb der Server mit Malcolm und Hedgehog Linux (schätzungsweise ungefähr halber Ressourcenanspruch von Malcolm) möglich?
    Antwort: „Ja, sollte möglich sein“
  - **Frage**: Wieviel Speicherplatz würde zur Verfügung stehen? Der benötigte Platz kann derzeit leider schlecht beurteilt werden, da die Belastung der Scan-Ziele unbekannt ist
    Antwort: „Wenn NFS möglich ist, sollten wir schon etwas im Bereich 20-40TB bereitstellen können“
  - **Frage**: Stand Prepaid-Kreditkarte?
    Antwort: „Die meisten Prepaid Karten sind App gebunden und es muss eine Identitätsprüfung gemacht werden. Am einfachsten ist es, wenn Du so eine Kreditkarte beschaffst [...] und diese gleich mit 300.- lädst. Dann gibst Du mir die Quittung und ich hole das Geld per Spesen zurück.“
  - **Frage**: Stand Publikation der Arbeit für Community?
    Für die Publikation unter `https://arbor.bfh.ch/` sind studentische Arbeiten ausgeschlossen [362]. Gibt es einen anderen Dienst von der BFH, der die Publikation einer studentischen Arbeit für die Community unterstützt? Ansonsten wird sie auf meiner Seite `https://guadm.github.io/` verlinkt werden.
    Antwort: „Die Rechte an der Arbeit gehören Dir, d.h. Du kannst diese publizieren wie Du willst. Wir wünschen uns aber, dass wir Dein Setup im CyberLab weiter verwenden dürfen.“
  - **Frage:** Wie ist bezüglich der VM-Installation an der BFH vorzugehen?
    Antwort: „Melde Dich betreffend Installation der VMs bei Donatello, er kann Dir da behilflich sein.“

Pendenzen

- Mauro
  - Prepaid-Kreditkarte organisieren, mit 300.- CHF aufladen und Quittung an Hansjürg senden ✔ 22. November 2024
  - Donatello Gallucci betreffend VM-Installation an der BFH kontaktieren ✔ 15. November 2024
    Termin für 27. November 2024 vereinbart

A.4.7. Besprechung 1. Review (27. November 2024)

Info

Teilnehmende:

Hansjürg Wenger

Bruce Nikkel

Mauro Guadagnini

Datum: 27. November 2024, 8:30 Uhr

Dauer: 60 min

Ort: Biel, SIPBB, Raum S.247 und Teams-Meeting

Protokoll

- Aktueller Stand und Aufbau Analyse-Umgebung erläutert
- Feedback
  - Allgemein Arbeit bisher in Ordnung
  - Feedback Hansjürg
    - In Einleitungskapitel: Gelieferte Ergebnisse aus Zielsetzung in separates Unterkapitel
    - Auf Muss- und Kann-Ziele im Anhang verweisen
  - Feedback Bruce
    - Handover-Termin mit Donatello und Kevin vereinbaren inkl. Troubleshooting und Tipps (nach Schlusspräsentation)
- Weiteres / Vorbereitete Fragen
  - Hansjürg reserviert Raum für Schlusspräsentation sobald möglich und teilt diesen mit
  - Donatello und Kevin für Schlusspräsentation und Handover einladen
  - Schlusspräsentation auf Hochdeutsch halten
  - Gesamten Quelltext nicht im Anhang führen, nur Ausschnitte und auf ganze Skripts in Git- bzw. GitLab Repository verweisen
  - Rolf als zweiten Co-Experten auf Titelblatt ausweisen
  - Kreditkarte Rückzahlung des Betrags sollte bis Mitte Dezember erfolgen (im schlimmsten Fall übergibt Hansjürg das Geld bar)
  - Status-Update vor Weihnachten versenden, danach wenn nichts mehr auftaucht vor 2. Review nochmals Status-Update senden

Pendenzen auf nächster Seite

Pendenzen

- Hansjürg
  - Definitiver Ort für Schlusspräsentation organisieren und mitteilen
    ✔ 15. Januar 2024
    Ort: Biel, SIPBB, Raum S.245 (Ausweich-Option S.344)
  - Repository in GitLab von BFH erstellen für Handover ✔ 27. November 2024
- Mauro
  - Einladung Kevin und Donatello zu Schlusspräsentation und Handover (beides am 26. März 2025)
    ✔ 27. November 2024
  - Handover vorbereiten
    ✔ 22. Februar 2025
  - Hansjürg informieren, wenn bis Mitte Dezember keine Zahlung eingetroffen ist ✔ 23. Dezember 2024
  - Status-Update vor Weihnachten senden ✔ 23. Dezember 2024
  - Rolf als Co-Experten vermerken
    ✔ 27. November 2024

A.4.8. Installation Aufzeichnungs- und Analyse-Server vor Ort

Info

Teilnehmende:
Donatello Gallucci
Mauro Guadagnini

Datum: 27. November 2024, 10 Uhr
Dauer: 5 h
Ort: Biel, SIPBB, Raum S.344

Protokoll

- Malcolm-Instanz und Appliance mit Hedgehog Linux VM auf Cyberlab Infrastruktur ab ISO-Datei gemäss Kapitel C installiert
  - Malcolm CPU 16 Cores (2 Sockets à 8 Cores), 32 GB RAM und 500 GB Disk mit IP `LAB-INTERN.200/24` [69]
  - Hedgehog Linux CPU 16 Cores (2 Sockets à 8 Cores), 16 GB RAM und 500 GB Disk mit IP `LAB-INTERN.201/24` [69]

    Nachtrag vom 20. Dezember 2024
    CPU-Ressourcen von 8 auf 16 Cores angepasst aufgrund erhöhtem Anspruch des Zeek-Skripts bei der Analyse (trotz Optimierungsversuche). Dies resultierte bei zu wenig CPU-Kernen in „Dropped_Packets"-Meldungen als Zeek-Notice im Dashboard „Zeek Notices"
- SSH-Public-Keys von Cyberlab und Mauro auf beiden Hosts hinterlegt
- WireGuard-VPN-Zugang eingerichtet für Zugang zu den VMs inklusive Hypervisor
- Scan-Ziel mit Debian ISO installiert
  CPU 1 Core (1 Socket mit 1 Core), 1 GB RAM und 20 GB Disk mit temporärer IP `LAB-INTERN.203/24` [69]

Pendenzen

- Donatello
  - Scan-Target mit öffentlicher IP
    ✔ 10. Dezember 2024
    IP-Adressen der Cyber-Security-Lab-Infrastruktur dürfen nicht veröffentlicht werden
    Automatische IPv6-Adresskonfiguration hinterlegt und funktional
  - VPN-Server im Internet erreichbar
    ✔ 10. Dezember 2024
    IP-Adressen der Cyber-Security-Lab-Infrastruktur dürfen nicht veröffentlicht werden
- Mauro
  - Passwörter anpassen
    ✔ 9. Dezember 2024
  - Cyberlab-SSH-Public-Keys in bestehenden Code integrieren
    ✔ 9. Dezember 2024
    „Authorized-Keys"-Datei in Ansible-Playbook verschlüsselt und in Debian-Preseed-Datei ohne Namen bzw. Identitäten in Schlüssel-Kommentar hinterlegt
  - Konfiguration der aufgebauten VMs abschliessen ✔ 10. Dezember 2024

[69] Die Adressen der Cyber-Security-Lab-Infrastruktur werden nicht öffentlich bekanntgegeben

A.4.9. Besprechung 2. Review (15. Januar 2025)

Info

Teilnehmende:

- Hansjürg Wenger
- Bruce Nikkel
- Rolf Lanz
- Donatello Gallucci
- Mauro Guadagnini

Datum: 15. Januar 2025, 8:30 Uhr

Dauer: 60 min

Ort: Biel, SIPBB, Raum S.247 und Teams-Meeting

Protokoll

- Aktueller Stand und Aufbau Analyse-Umgebung erläutert inkl. Demonstration Malcolm mit Arkime und OpenSearch Dashboards
- Feedback
  - Allgemein Arbeit bisher in Ordnung
- Weiteres
  - Book-Beitrag-Einladung erhalten, für Weiterbildungs-Bereich sind laut Anleitung keine Experten oder Betreuer im Tool zu hinterlegen
  - Experten in Book-Beitrag im Fliesstext aufführen
  - Präsentation muss nicht mit Thesis abgegeben werden
  - Setup in BFH Cyber-Security-Lab-Infrastruktur mit Unterstützung von Donatello abgeschlossen (BFH-IP-Adressen werden nicht veröffentlicht)
  - Handover-Termin mit Donatello und Kevin am 26. März 2025 (Protokoll inklusive Tipps in Vorbereitung, siehe Kapitel A.4.10)
  - Daten über Scan-Quellen werden täglich eingeholt (z.B. zwischen 30.11.2024 und 02.01.2025 21'000 neue Einträge, von 2. auf 3. Januar 600 neue Einträge)
  - Scan-Ziele werden nach Arbeit gelöscht, Scan-Ziel bei Handover gemeinsam neu aufbauen
  - Nur eingehende Verbindungen werden derzeit von Scan-Ziel gespiegelt
    - Replikation ausgehender Verbindungen in Skript (Quelltext D.1) auskommentiert
    - Grund ist Last-Verringerung auf Malcolm-Komponenten („Dropped Packets“ wurden zuvor geloggt, zusätzlich weitere Optimierungen getätigt)
    - Ausgehende Verbindungen in Ausblick erwähnen (z.B. zur Analyse bei gehacktem Host)
  - Hoster mit gesamten /64-IPv6-Präfix (Hetzner [354]): Nur 1 IP nehmen
  - Microsoft akzeptiert keine Prepaid-Kreditkarte, daher private Karte genommen
  - Betrag noch nicht von BFH rückerstattet, Abrechnung am Schluss der Arbeit
  - Public IPv6-Adressen teilweise nicht bei allen Orten (Anbieter bieten für bestimmte Orte keinen IPv6 Support)
  - Gewählte geografische Standorte der Scan-Ziele für diese Arbeit OK

- In Analyse-Teil zu beobachtender Zeitraum: Ab 27. Dezember 2024 bis 12. Feburar 2025 ist in Ordnung

> **Nachtrag vom 12. Februar 2025**
> Die Scan-Ziele `st002` und `st003` starteten im Januar aufgrund von Timeouts nach einem Neustart (automatische Aktualisierungen) den ERSPAN-Port-Mirroring-Service nicht automatisch (siehe Tabelle C.1).
> Der Vorfall trat am 10. Februar nochmals bei denselben Scan-Zielen auf.
>
> Daher wird der Zeitraum auf folgenden verkürzt:
> 27. Dezember 2024 bis und mit 9. Februar 2025
> Die Experten werden über diese Anpassung per E-Mail informiert.
> Mehr zum ausgewerteten Zeitraum ist dem Beginn von Kapitel 4 zu entnehmen.

- Stehen BFH Suchmaschinen wie Censys oder Shodan in kostenpflichtigen Varianten zur Verfügung? Unter anderem kostenlos verwendet: Shodan, Censys, onyphe, driftnet [19, 20, 249, 290]. shadowserver.org, recyber.net unterstützen z.B. Bildungsinstitutionen [21, 185]
  - ⋆ Gerade nichts bekannt
  - ⋆ Hansjürg fragt bei IT nach, ob so ein Zugang für „Intelligence Feeds“ mit nicht öffentlichen Daten vorhanden ist
- Interpretationskapitel mit eigener Ansicht (u. a. unbestätigte Vermutungen oder Hypothesen) in Ergebnisse aufführen (z.B. Unterschiede zwischen Länderangaben unterschiedlicher Quellen)
- IPv6-Adressen der Scan-Ziele kaum/nicht genutzt
  - ⋆ Eine RIPE Atlas Probe zu `st003` (IP endet mit `::1`) am 8.1.25
  - ⋆ Einzelne Verbindungen von internen Netzen an Scan-Ziele
  - ⋆ DNS-Einträge nur für IPv6-Adressen bei beispielsweise Amazon AWS mit .com Domäne erstellen (möglichst neutral), bei Unterstützungsbedarf Hansjürg kontaktieren
  - ⋆ DNS-Einträge für IPv4-Adressen in Ausblick vermerken
- Quelltext-Ausweisung gemäss aktuellem Stand in Ordnung
- In Arbeit ausweisen, dass Scan-Ziele nicht für produktiven Traffic verwendet werden und somit Anteil an Scan-Verkehr höher ist als z.B. bei Heo und Shin, die den Verkehr auf produktiven Firewalls betrachtet haben
- Meinung zu Darstellung von Flaggen bei Scan-Zielen? OK

Pendenzen

- Hansjürg
  - Abklärung bei BFH-IT bezüglich Möglichkeiten für nicht öffentlich einsehbare Informationen bzw. „Intelligence Feeds“ zur Analyse der Scan-Quellen
    ✗ 19. Februar 2025
    Es stehen derzeit keine solchen Mittel zur Verfügung
- Mauro
  - Gemeinsamer Aufbau eines Scan-Ziels für Handover (26. März 2025) vorbereiten / kontrollieren (inkl. Aktivierung Port-Mirroring für ein- und ausgehende Kommunikationen)
    ✓ 22. Februar 2025
  - DNS-Einträge zu IPv6-Adressen der Scan-Ziele ✓ 17. Januar 2025

A.4.10. Handover (26. März 2025)

Hierbei handelt es sich um ein vorbereitetes Protokoll. Es stellt nicht dar, was effektiv stattfand, jedoch was dafür vorbereitet wurde.

Info

Teilnehmende:

Donatello Gallucci

Kevin Georg Schrag

Mauro Guadagnini

Datum: 26. März 2025, 13 Uhr

Ort: Biel, SIPBB, Raum S.344

Protokoll

- Übergabe-Pendenzen Mauro
  - Passwörter übergeben inklusive Passwort-Wechsel-Empfehlung
    - Ansible-Vault
    - Malcolm-WebGUI-Administrator
    - Malcolm-Instanz
    - Sensor / Appliance mit Hedgehog Linux
    - Scan-Ziel
  - Zugang zur Cyberlab-Infrastruktur entfernen
    - VPN-Verbindung
    - Persönlicher Malcolm-WebGUI-Account
    - Proxmox-Account
  - Übergabe Kontaktdaten bei Unterstützungsbedarf
- Kurzbeschrieb Analyse-Umgebung und Zusammenhänge zwischen Malcolm-Instanz, Sensor und Scan-Ziel (Abbildung und Weiteres dazu in Kapitel 3.1)
- Installationsdokumentation in Kapitel C
- Allgemeine Malcolm-Dokumentation unter `https://github.com/cisagov/Malcolm/blob/main/README.md` oder unter der beim GitHub-Repository verlinkten Webseite
- Code (Skripts in Tabelle D in Kapitel D beschrieben)
  - Ordner `erspan` beinhaltet das ERSPAN-Entkapselungs-Skript inklusive systemd-Service-Datei, das auf dem Sensor mit VPN-Server-Funktionalität ausgeführt wird
  - Ordner `zeek` enthält das Zeek-Skript inklusive `loaddata.sh` zur automatisierten Anreicherung der Tabellen mit bekannten IP-Adressen, -Subnetzen oder FQDNs/Domänen
  - Ordner `opensearchdashboards-objects` beinhaltet Visualisierungen und das „Scanner Detection“-Dashboard als exportierte „Saved Objects“ aus OpenSearch Dashboards (falls die Analyse-Umgebung mit Malcolm neu aufgebaut wird)
    Die Visualisierungen sind vor dem Dashboard zu importieren
  - Ordner `debian-install` ermöglicht die Modifikation einer Debian-ISO-Datei zur u. a. Hinterlegung erlaubter SSH-Public-Keys oder Definition von Benutzerkennwörter
    Startet ein Scan-Ziel von der ISO-Datei, wird die Installation automatisch durchgeführt
  - Ordner `ansible` enthält das Playbook sowie zugehörige Skripts (inklusive ERSPAN-Skript) zur Konfiguration der Scan-Ziele
    „README“ pro Rolle und für ganzes Playbook vorhanden

- Handhabung Weboberfläche
  - Unter der IP-Adresse der Malcolm-Instanz steht eine Weboberfläche zur Verfügung und beinhaltet u. a.
    - Arkime
    - OpenSearch Dashboards
    - Benutzerverwaltung
    - Cyberchef
    - Möglichkeit, PCAP-Dateien hochzuladen
  - OpenSearch Dashboards
    - In OpenSearch Dashboards gespeicherte Objekte sind für alle Benutzer ersichtlich
    - Malcolm bringt viele Visualisierungen und Dashboards mit, die unabhängig von der Scan-Detektion dieser Arbeit nützlich sein können
  - Arkime-Views können je nach Konfiguration mit anderen Benutzern geteilt werden
  - Von Arkime aufgezeichnete Sitzungen können als PCAP-Dateien exportiert werden (Zeek behaltet keine Aufzeichnungen, nur Logs)
  - Suche / Filter
    - Immer auf gewählten Zeitrahmen achten
    - Cheat Sheet: `https://github.com/cisagov/Malcolm/blob/main/docs/README.md` [363]
    - ScannerDetection-Zeek-Notices unter Arkime: `rule.category == ScannerDetection`
  - OpenSearch Dashboards
    - ScannerDetection-Zeek-Notices: `rule.category: ScannerDetection`
    - „good“ in ScannerDetection-Zeek-Notice-Message: `zeek.notice.msg: *good*`
    - Für Arbeit hinterlegte „Saved Queries“ können weitere Beispiele enthalten (absichtlich nicht exportiert, da sie Autor- und Lab-IP-Adressen beinhalten)
  - Bisher bewährtes Vorgehen: Mit OpenSearch Dashboards Ansatzpunkte finden und mit Arkime bis zur PCAP-Datei eintauchen
  - Anfragen können je nach Zeitspanne die Ressourcen der Server spürbar beanspruchen

- Handhabung Malcolm-Instanz
  - Upgrade via `https://github.com/cisagov/Malcolm/blob/main/docs/malcolm-upgrade.md` zu Beginn einmal durchgespielt, verlief erfolgreich
  - Malcolm-interne Skripts für z.B. Status-Informationen unter `/home/user/Malcolm/scripts`
  - Komponenten auf Malcolm-Instanz werden als Docker-Container betrieben
  - Hier sind keine eigenen Skripts platziert

- Handhabung Sensor/Appliance mit Hedgehog Linux + VPN-Server
  - Eigene Skripts und Zusatz-Software: innernet (VPN-Server) und ERSPAN-Entkapselungs-Skript (Starten als systemd-Services automatisch mit Server)
  - Befehle und Informationen bei Login angezeigt (`/etc/motd`)
  - Upgrade-Versuch im November 2024 gescheitert, Neu-Installation gemäss Kapitel C.2 innernet Daten unter `/etc/innernet-server` und `/var/lib/innernet-server` zuvor sichern, bei Neu-Installation nach Vorgang mit `apt` die Dateien zurückkopieren und Befehl `innernet-server new` überspringen
  - innernet wurde für den vereinfachten Aufbau gewählt, kann aber zukünftig durch eine native WireGuard-Konfiguration ersetzt werden. Die entsprechende Ansible-Rolle zur Scan-Ziel-Konfiguration prüft, ob der Server-Public-Key in der Konfiguration bereits hinterlegt ist und konfiguriert die Verbindung nur wenn sie nicht hinterlegt ist
  - `journalctl` zeigt für Aufzeichnung nicht relevante Fehlermeldungen (Aus Zeitgründen nicht betrachtet, evtl. per Update von Malcolm oder Hedgehog Linux bereinigt)
  - VPN-Serverdienst
    - Installiert mittels Package-Manager `apt` von eigenem Repository
    - innernet-Server-Status prüfen:
      ```
      systemctl status \
       innernet-server@innernet
      ```
    - Client-Konfiguration siehe Kapitel C.2.1
    - Anwendung `innernet-server` als `root` ausführen
  - ERSPAN-Entkapselungs-Skript
    - ERSPAN-systemd-Service-Name: `erspan-decapsule@innernet`
    - Skript-Platzierung unter `/usr/local/bin/erspan-decapsule.sh`
    - Für jeden VPN-Client wird ein ERSPAN-Subinterface `myerspanXXXX` (mit Nummer als Suffix) aufgebaut, dessen Pakete zum ERSPAN-Haupt-Interface `myerspan` gespiegelt werden
    - ERSPAN-Interface-Übersicht als `root` (ohne `root` werden derzeit die Service-Logs nicht ausgegeben):

      ```
      1 watch -n 1 -d "netstat -i | grep -E \"(table|Iface|myerspan)\" ;
           journalctl -r -n 5 -t erspan-decapsule.sh ; cat /var/run/
           erspan-decapsule.sh.run"
      ```

      Quelltext A.2: Befehl zu Anzeige der ERSPAN-Interface-Übersicht auf dem Hedgehog-Linux-Sensor bzw. VPN-Server

    - Temporäre Datei `/var/run/erspan-decapsule.sh.run` zeigt zu welcher VPN-Client-IP welches ERSPAN-Interface angelegt wird
    - ERSPAN-Service baut Sub-Interfaces bei Übertragungsfehler neu auf
      Scan-Ziel in Azure (`st006` 🇦🇺) erzeugte als einziges regelmässig einen ERSPAN-Interface-Neuaufbau (Übertragungsfehler `RX-ERR` und `RX-OVR` [364], scheint auf Aufzeichnungen keinen Einfluss zu haben)

  - Hedgehog Linux Komponenten und Zeek-Skript
    - Hedgehog Service Status: `/opt/sensor/sensor_ctl/status`
    - Logs pro Service unter `/opt/sensor/sensor_ctl/log/`
    - Zeek-Logs unter `/home/sensor/zeek_logs/logs/`
    - Eigenes Zeek-Skript unter `/opt/sensor/sensor_ctl/zeek/custom`
  - Ansible-Playbook zur Konfiguration der Scan-Ziele wird hier ebenfalls unter `/home/sensor/ansible` abgelegt und ausgeführt
- Scan-Ziel
  - Installation mittels modifizierter Debian-ISO-Datei, Konfiguration mittels Ansible
  - Bei der Installation wird nicht der gesamte Speicher einer Disk alloziert, damit er darauf mit Ansible gemäss CIS Debian Linux 12 Benchmark [351] eingerichtet werden kann
  - Die Ansible-Rolle `guadm.hardening` implementiert automatisierte Updates mit Neustart um 2 Uhr (In `playbook.yml` wird die Zeitzone `UTC` mit einer Zeitsynchronisation definiert)
  - Handhabung Scan-Ziel
    - ERSPAN-systemd-Service-Name: `erspan@innernet`
    - SSH-Zugang derzeit via Hedgehog-Sensor/-Appliance als Jumphost
      Derzeit nur via IP-Adressen von VPN-Server aus erreichbar

```
1 Host chedgehog
2         HostName LAB-INTERN.201
3         User sensor
4         ForwardAgent yes # to use key for ansible
5 Host st001
6         HostName 10.25.1.1
7         User configuration
8         PreferredAuthentications publickey,password
9         ProxyJump chedgehog
```

Quelltext A.3: SSH-Client-Konfigurationsdatei `~/.ssh/config` zum Zugriff auf Scan-Ziele über den VPN-Server als Jumphost

    - Ausführung von Ansible auf eigenem Laptop mit Sensor/Appliance (Hedgehog Linux) als Jumphost zusammen mit YubiKey hat Probleme
      Als Workaround Ansible direkt auf Sensor/Appliance ausgeführt
    - Bei Initial-Setup mit Ansible unter `/home/sensor/ansible`
      `ansible-playbook playbook.yml -i inventory --ask-vault-pass`
      - Öffentliche IP-Adresse des Scan-Ziels im SSH-Client-Konfiguration hinterlegen, danach Scan-Ziel-IP wechseln auf interne entsprechend VPN-Konfiguration
      - In `ansible/group_vars/scantargets.yml` besagt die Variable `ssh_allowed_source_ips` welche Quell-IP-Adressen auf die Scan-Ziele per SSH zugreifen dürfen
      - Nach erster Konfiguration muss für nachfolgende SSH-Zugriffe oder Playbook-Durchführung für jedes Ziel der YubiKey berührt werden (eine LED am YubiKey blinkt falls nötig)
      - Auch bei Reboot-Tasks im Ansible-Playbook wird eine Verbindung neu aufgebaut (YubiKey nochmals berühren)

- Gemeinsamer Aufbau eines Scan-Ziels
  - Debian-ISO
    - `debian-install/scantarget-preseed.cfg` gegebenenfalls modifizieren (Passwort-String erstellbar mittels `mkpasswd -m sha-512`)
    - Das neue Passwort muss dann in der entsprechenden Ansible-Variable ebenfalls angepasst werden (Datei `ansible/group_vars/scantargets.yml`,
      String mittels Ansible-Vault wie folgt verschlüsseln
      `echo "hello" | ansible-vault encrypt_string --ask-vault-pass`)
    - Sicherstellen, dass SSH-Public-Key für Ansible-Konfiguration hinterlegt ist (sonst nach Installation hinzufügen)
    - ISO-Datei generieren mittels folgendem Befehl
      ```
      ./modify-debian-iso.sh \
      --iso ~/Downloads/debian-12.9.0-amd64-netinst.iso \
      --preseed scantarget-preseed.cfg
      ```
    - ISO-Datei wird neben originaler Datei abgelegt
    - Scan-Ziel von ISO-Datei booten (Installation sollte durchlaufen mit Netzwerkkonfiguration mittels DHCP)
    - Login mittels Benutzer `configuration` möglich (hat `sudo`-Rechte)
    - Scan-Ziel sollte mindestens über 1 GB Arbeitsspeicher verfügen, sonst gibt es vom Debian-Installer eine „Low memory“-Warnung
  - innernet-Client-Konfiguration vorbereiten gemäss Listenpunkt 6 in Kapitel C.2.1 (Mehr Infos zum Verhalten und zu verwendenden Parameter sind dort notiert)
    - Via SSH auf Sensor / Appliance mit Hedgehog Linux einloggen
    - Zu `root`-Sitzung wechseln (mit `su -`)
    - In Ordner wechseln, in den die Client-Konfiguration abgelegt werden soll
    - Client-VPN-Netzwerk anlegen mittels `innernet-server add-cidr innernet`
      (`Parent CIDR: innernet`)
      Netzwerke bis Nummer 10 vorbereitet und verwendet, jedoch ist nur noch ein Scan-Ziel im ersten Netzwerk aktiv. Andere Scan-Ziele wurden deaktiviert, aber werden nicht aus der innernet-Konfiguration gelöscht, weshalb diese Netzwerke schon vorbereitet sind
    - Client-Konfiguration anlegen mittels `innernet-server add-peer innernet`
      (Client keine Admin-Berechtigungen erteilen, sonst kann dieser die innernet-Konfiguration modifizieren)
    - Angelegte Datei (mit `.toml`-Endung) weg kopieren und von Server löschen (wird später mit Ansible verwendet)
    - `root`-Sitzung verlassen und zurück zu vorheriger Benutzer-Sitzung (Befehl `exit`)

- SSH-Konfiguration und Ansible auf z.B. Sensor / Appliance vorbereiten
  - ⋆ IP-Adresse von neuem Scan-Ziel in Erfahrung bringen oder anpassen (mittels DHCP-Leases oder Konsole)
  - ⋆ SSH-Client-Konfiguration vorbereiten in `~/.ssh/config`
    Wie in Quelltext A.3 für `st001`, je nach Verbindung ohne `ProxyJump`-Angabe
  - ⋆ Sicherstellen, dass SSH-Public-Key auch in `ansible/files/ssh_authorized_keys` vorkommt (`ansible-vault edit files/ssh_authorized_keys`)
  - ⋆ SSH-Verbindung testen mit `ssh stXXX`
  - ⋆ Netzwerk-Interface-Bezeichnung notieren, die gespiegelt werden soll
    Wird später fürs Inventar bzw. Quelltext A.4 benötigt
  - ⋆ Ansible-Playbook vorbereiten
    - · Dateien z.B. unter `~/ansible` ablegen
      Von anderem Rechner via SSH synchronisieren mittels `rsync -avP -e ssh ansible hedgehog:`
    - · In `ansible`-Ordner wechseln
    - · Sollen ausgehende Verbindungen ebenfalls aufgezeichnet werden? Hierfür wären nun Zeilen 20-21 in D.1 zu auskommentieren
    - · innernet-Client-Konfigurationsdatei anlegen mittels z.B.
      `ansible-vault create files/innernet_client_config/client-XX-1.toml`
      Passwort für Ansible-Vault angeben
    - · `inventory`-Datei ergänzen (neuen Eintrag unter `hosts:` hinzufügen)

```
1 global:
2   hosts:
3     ...
4     stXXX:
5       innernet_client_configfile: innernet_client_config/client
            -XX-1.toml
6       erspan_porttomirror: enp0s3
```

Quelltext A.4: Ansible-Inventory für neues Scan-Ziel ergänzen

- Ansible-Playbook ausführen
  - ⋆ `ansible-playbook playbook.yml -i inventory --ask-vault-pass`
  - ⋆ `--limit 'stXXX'` anfügen, um Playbook nur für diesen Host auszuführen
  - ⋆ Scan-Ziel wird bei Partitions-Anpassungen neu starten (ggf. ist dann der YubiKey erneut zu berühren)
  - ⋆ Bei Fehler sind u. a. die innernet-Client-Konfiguration sowie die Verbindung vom Scan-Ziel zum VPN-Server zu prüfen
  - ⋆ IP-Adresse in `~/.ssh/config` anpassen, sodass interne VPN-IP-Adresse verwendet wird (erlaubte SSH-Quell-IP-Adressen in Ansible-Variable sind nun in Firewall auf Scan-Ziel aktiv)

- Konfiguration kontrollieren
  - SSH-Verbindung zu Scan-Ziel testen (`ssh stXXX`)
  - Kontrolle auf Scan-Ziel
    - ERSPAN-Service prüfen mittels `sudo systemctl status erspan@innernet`
    - Service sollte aktiv sein und im Log ist das zu spiegelnde Netzwerk-Interface vermerkt
  - Kontrolle auf Sensor/Appliance/VPN-Server
    - `watch`-Befehl aus Quelltext A.2 ausführen
    - Es soll ersichtlich sein, dass für das neue Scan-Ziel ein neues Interface angelegt wurde
    - Je nach Aktivität beim Scan-Ziels sollten sich die Statistiken des ERSPAN-Interfaces aktualisieren
    - Weiter testen mit `tcpdump` und z.B. Ping-Anfrage an Scan-Ziel
  - ERSPAN-Haupt-Interface wird aufgezeichnet, wessen Daten im Malcolm-Webinterface ersichtlich sein sollten
- Malcolm-Dashboards Filter
  - Filter in OpenSearch Dashboards und Views in Arkime anpassen
  - Je nach Bedarf, falls z.B. nur Verkehr ersichtlich sein soll, der nicht vom Scan-Ziel kommt, aber zum Scan-Ziel geht
  - Gegebenenfalls unerwünschte Protokolle oder Adressen filtern

- Troubleshooting-Tipps
  - Kein Zugriff per SSH auf Scan-Ziel
    - Darf die Quell-IP-Adresse eine Verbindung herstellen?
      Prüfen der Scan-Ziel-Firewall mittels `sudo ufw status`
    - Ist der entsprechende SSH-Public-Key auf dem Scan-Ziel hinterlegt?
    - Im schlimmsten Fall muss auf die Konsole des Scan-Ziels direkt zugegriffen werden
    - Eventuell gilt es in `ansible/group_vars/scantargets.yml` die Variable `ssh_allowed_source_ips` anzupassen und das Playbook nochmals auszuführen (derzeit am besten nochmals `ufw` mit `apt` von Scan-Ziel deinstallieren, da Konfiguration nur nach `ufw`-Task mit Status „Change“ angewendet wird)
    - Bei manueller Anpassung (zu vermeiden) auf dem Scan-Ziel auch die Werte bei den Ansible-Variablen nachführen
  - Es treffen keine Aufzeichnungen des Scan-Ziels ein
    - Besteht eine Verbindung zwischen dem Scan-Ziel und dem VPN-Server?
      - Service prüfen auf dem Server als `root` mittels `systemctl status innernet-server@innernet` oder `wg show`
      - Service prüfen auf dem Client als `root` mittels `systemctl status innernet@innernet` oder `wg show`

        - Ist der ERSPAN-Dienst auf dem Scan-Ziel aktiv?
          Prüfen als `root` mittels `systemctl status erspan@innernet`
        - Sind auf dem Scan-Ziel die Angaben in `/etc/environment` korrekt? Variablen beginnen mit `ERSPAN_` und beinhalten
            - Ziel-IP-Adresse (hier der Sensor/Appliance mit VPN-Server)
            - ERSPAN-Key (Muss auf Client und Server derselbe sein, auf Server auch unter `/etc/environment` hinterlegt)
            - Zu spiegelndes Interface unter `ERSPAN_PORT_TO_MIRROR`
            - Nach Anpassung ist ein Service-Neustart durchzuführen
        - Werden die Netzwerkpakete auf dem Server an das Interface `myerspan` gespiegelt?
            - Status mittels Befehl aus Quelltext A.2 prüfen
              `/var/run/erspan-decapsule.sh.run` zeigt welches Interface zu welchem Client gehört
              `RX-OK`-Wert sollte sich beim entsprechenden Interface regelmässig erhöhen, sofern das entsprechende Scan-Ziel aktiv angesprochen wird
            - Mittels `tcpdump -i myerspan` oder `tcpdump -i myerspanXXXX` (Client-spezifisches Interface) sollten die Netzwerkpakete angezeigt werden
            - Sind die Dienste auf dem Sensor bzw. der Appliance aktiv?
              `/opt/sensor/sensor_ctl/status` sollte viele Dienste als `RUNNING` deklariert haben (einige Dienste, die nicht verwendet werden, sind mit `STOPPED` notiert)
        - Funktioniert das Netzwerk zwischen Sensor und Malcolm-Instanz?
    - Im „Zeek Notices“-Dashboard in OpenSearch Dashboards wurden zu Beginn Einträge des Typs „Dropped Packets“ geloggt
        - Dies kann vorkommen, wenn die Ressourcen der Malcolm-Komponenten, besonders die des Sensors nicht genügen
        - Aus diesem Grund wird derzeit nur eingehender Verkehr auf den Scan-Zielen gespiegelt
          Anpassbar in `ansible/roles/guadm.erspan/files/erspan.sh`, siehe Zeilen 20-21 in Quelltext D.1 in Kapitel D.1 (Aktualisierung auf Scan-Zielen mittels Ansible)
        - Eventuell kann auch ein Ausbau der Ressourcen helfen
- Ausbau-Optionen
    - Monitoring der Malcolm-Komponenten und Scan-Ziele einführen
      Scan-Ziele `st002` 🇳🇬 und `st003` 🇩🇪 starteten zweimal nach Neustart aufgrund automatischer Updates den ERSPAN-Service nicht mehr
    - Evtl. Ressourcen der Malcolm-Komponenten ausbauen

Anhang B. Verwendung von KI-gestützten Tools

Tabelle B.1.: Verwendung von KI-gestützten Tools (ChatGPT)

ChatGPT (Model „GPT-4o“)		
Datum	Funktionsart und Umfang	Prompt
19.08.2024	Findung Quellen als Suchmaschine (Versuch, Überblick zu verschaffen)	`are there any reports about identifiying sources scanning servers in the internet`
19.08.2024	Findung Quellen als Suchmaschine (Versuch, Überblick zu verschaffen)	`can you give me links to some of those reports please`
19.08.2024	Findung Quellen als Suchmaschine (Versuch, Überblick zu verschaffen)	`More reports available?`
19.08.2024	Findung Quellen als Suchmaschine (Versuch, Überblick zu verschaffen)	`are there reports about the attribution of scans being done in the internet`

Anhang C. Installationsdokumentation

Dieses Kapitel beschreibt die Installation der Analyse-Umgebung. Hierbei werden Abweichungen zwischen dem lokalen Aufbau und der Variante im Internet entsprechend aufgeführt.

Folgende Symbole werden zum Unterschied zwischen den Umgebungen verwendet:

- 💻 Lokal aufgebaute Analyse-Umgebung
- 🌐 Globale, im Internet aufgebaute Analyse-Umgebung

Zur grafischen Übersicht wird auf Abbildung 3.1 verwiesen.

C.1. Malcolm-Instanz

Malcolm wird auf einem eigenen Server installiert. Dazu wird die Malcolm-ISO-Datei (hier `malcolm-24.11.0.iso`) gemäss Anweisungen unter `https://github.com/cisagov/Malcolm/releases` heruntergeladen und dessen Hash-Wert überprüft [365].

Die System-Ressourcen sind entsprechend zu beachten (`https://github.com/cisagov/Malcolm/blob/main/docs/system-requirements.md`), da zum Beispiel eine Maschine mit 8 GB Arbeitsspeicher zu wenig ist[70] [361].

Die Malcolm-Installation verläuft entsprechend folgenden Schritten [141, 142, 153, 366]:

1. Boot-Vorgang der Maschine ab ISO-Datei
2. Standard-Netzwerk-Interface auswählen
   💻 : `wlp0s20f3`, 🌐 : `ens18`
3. Hostname: 💻 : `malcolm`, 🌐 : `malcolm`
4. Domain: 💻 : keine Angabe, 🌐 : `LAB-DOMAIN` [71]
5. Account-Passwörter anlegen
6. Automatischen Installationsvorgang abwarten
7. Fragen beantworten
   - „Format non-OS drive(s) for artifact storage?“: `Yes`
   - „Disable IPv6?“: `No`
   - „Automatically login to the GUI session?“: `No`
   - „Should the GUI session be locked due to inactivity?“: `Yes`
   - „Display the Standard Mandatory DoD Notice and Consent Banner?“: `No`
   - „Allow SSH password authentication?“: `Yes` (Wird in einem späteren Schritt deaktiviert)
8. Betriebssystem startet und zeigt Login-Maske
9. Mit Nicht-`root`-Account anmelden
10. Fenster mit Konfigurations-Skript `/home/user/Malcolm/scripts/configure` startet automatisch, Fenster schliessen

[70] Hier wird beobachtet, dass mit 8 GB RAM Container beendet werden und das System nicht funktionsfähig wird.
[71] Die Adressen der Cyber-Security-Lab-Infrastruktur werden nicht öffentlich bekanntgegeben

11. Terminal öffnen und folgenden Befehl eingeben, um die Konfiguration durchzuführen:

```
./Malcolm/scripts/configure \
--defaults true \
--restart-malcolm true \
--suricata-rule-update true \
--file-extraction none \
--auto-arkime false \
--live-capture-arkime false \
--live-capture-netsniff false \
--auto-zeek false \
--live-capture-zeek false \
--auto-suricata false \
--live-capture-suricata false \
--opensearch-expose true \
--logstash-expose true \
--filebeat-tcp-expose true \
--netbox true \
--netbox-enrich true \
--netbox-autopopulate true \
--netbox-auto-prefixes true \
--netbox-site-name analysisenv
```

Quelltext C.1: Malcolm-Konfiguration mit Parameter, um externe Daten zu akzeptieren (Beispielsweise von Hedgehog Linux)

12. Authentisierungs-Setup im Terminal ausführen: `./Malcolm/scripts/auth_setup` und Fragen wie folgt beantworten:
    - `all` wählen
    - „Store administrator username/password for local Malcolm access?“: `Yes` → Administrator-Konto angeben (Name und Passwort)
    - „Configure remote primary or secondary OpenSearch/Elasticsearch instance?“: `No`
    - „Store username/password for OpenSearch Alerting email sender account?“: `No`
    - „(Re)generate internal passwords for NetBox?“: `Yes`
    - „Store password hash secret for Arkime viewer cluster?“: `Yes`
    - „Arkime password hash secret:“: Secret eingeben und notieren (wird nochmals in Kapitel C.2 verwendet)
    - „Transfer self-signed client certificates to a remote log forwarder?“: `No`
13. Malcolm im Terminal mittels `./Malcolm/scripts/start` starten
14. Als `root` folgende Befehle zur Konfiguration der Zeit mittels NTP ausführen
    - `apt install -y systemd-timesyncd`
    - `timedatectl set-ntp true`
15. SSH-Public-Key hinterlegen und reine Passwort-Authentisierung deaktivieren
    - SSH-Public-Key unter `~/.ssh/authorized_keys` des Nicht-`root`-Accounts hinterlegen
    - Als `root` folgende Befehle ausführen:
        - `sed -i 's/PasswordAuthentication yes/PasswordAuthentication no/' \ /etc/ssh/sshd_config`
        - `systemctl restart sshd`

C.2. Hedgehog Linux Sensor

Hedgehog Linux wird auf einem eigenen Server installiert. Dazu wird die Hedgehog-Linux-ISO-Datei (hier `hedgehog-24.11.0.iso`) gemäss Anweisungen unter `https://github.com/cisagov/Malcolm/releases` heruntergeladen und dessen Hash-Wert überprüft [365].

Es ist zu beachten, dass diese Installation und die Malcolm-Instanz sich gegenseitig erreichen können müssen. Die Host-Firewall beider Installationen werden mit einer entsprechenden Konfiguration ausgeliefert. Diese kann wie folgt eingesehen werden, um z.B. entsprechende Ports auf einer dazwischenliegenden Firewall freizuschalten: Als `root` den Befehl `ufw status` ausführen.

Die Installation von Hedgehog Linux verläuft entsprechend folgenden Schritten [153]:

1. Boot-Vorgang der Maschine ab ISO-Datei
2. Account-Passwörter anlegen
3. Automatischen Installationsvorgang abwarten
4. Fragen beantworten
   - „Format non-OS drive(s) for artifact storage?“: `Yes`
   - „Disable IPv6?“: `No`
   - „Automatically login to the GUI session?“: `No`
   - „Should the GUI session be locked due to inactivity?“: `Yes`
   - „Display the Standard Mandatory DoD Notice and Consent Banner?“: `No`
   - „Allow SSH password authentication?“: `Yes` (Wird in späterem Schritt deaktiviert)
5. Betriebssystem startet und zeigt Login-Maske
6. Mit Account `sensor` und definiertem Passwort anmelden
7. Der Kiosk-Modus wird angezeigt, welcher mit der Tastenkombination `Alt+F4` geschlossen wird
8. „Configure Interfaces and Hostname“ in Toolbar oben klicken
9. In neuem Fenster das `root`-Passwort eingeben und folgende Konfiguration durchführen:
   - `Continue` wählen
   - Interface: Interfaces konfigurieren
   - Hostname: Hostname definieren 💻 : `hedgehog`, 🌍 : `hedgehog`
   - Time Sync: „Use a Malcolm Server“: IP der Malcolm-Instanz und Port 443
   - „Allow SSH password authentication?“: `Yes` (Wird in nächstem Schritt deaktiviert)
10. SSH-Public-Key hinterlegen und reine Passwort-Authentisierung deaktivieren
    - SSH-Public-Key unter `~/.ssh/authorized_keys` des Nicht-`root`-Accounts hinterlegen
    - Als `root` folgende Befehle ausführen:
      - `sed -i 's/PasswordAuthentication yes/PasswordAuthentication no/' \ /etc/ssh/sshd_config`
      - `systemctl restart sshd`
11. Sendmail-Anwendung deaktivieren (Erzeugt Fehler im System-Log und wird nicht verwendet) `systemctl disable --now sendmail`
12. Gruppe „lp“ anlegen (Notiert aufgrund vorkonfigurierten udev-Rules Fehler im System-Log, wird nicht weiter verwendet, da Gruppe für Drucker vorgesehen ist [367]) `groupadd lp`

C.2.1. VPN-Server

Der VPN-Server wird mittels innernet auf der Installation von Hedgehog Linux aufgebaut[72]. Dazu ist wie folgt mit dem Account `root` in einer Kommandozeile vorzugehen [128, 368]:

1. Repository für `innernet-server` hinterlegen
   - `curl -sS https://tommie.github.io/innernet-debian/repository.asc \ | tee /etc/apt/keyrings/github-tommie-innernet.asc >/dev/null`
   - `cat >/etc/apt/sources.list.d/innernet.list <<EOF`
     `deb [signed-by=/etc/apt/keyrings/github-tommie-innernet.asc]`
     `https://tommie.github.io/innernet-debian/debian jammy contrib`
     `EOF`
   - `chmod 644 /etc/apt/keyrings/github-tommie-innernet.asc`
   - `apt update`
2. `innernet-server` installieren
   `apt install innernet-server`
3. `innernet-server` konfigurieren
   `innernet-server new`
   - `network name:` innernet
   - `network cidr:` 10.25.0.0/16
   - `listen port:` 51820 (Default)
   - `Auto-detect external endpoint IP address (via a DNS query to 1.1.1.1)?:` No
   - `External endpoint`: Adresse (IP-Adresse und Port), unter welcher die Clients Kontakt zum VPN-Server aufbauen können
     💻 : 10.1.2.2:51820, 🌍 : LAB-PUBLIC-IPV4-VPN:51820 [73]
4. Firewall von Hedgehog Linux für VPN-Traffic konfigurieren
   - Verkehr zum externen Endpunkt erlauben
     `ufw insert 1 allow proto udp to VPNEXTIP port 51820`
     VPNEXTIP = 💻 : 10.1.2.2, 🌍 : LAB-INTERN.201/24 [73]
   - Kommunikation der innernet-Dienste innerhalb des VPN-Tunnels erlauben
     `ufw insert 2 allow proto tcp from 10.25.0.0/16 to 10.25.0.1 port 51820`
   - Senden von ERSPAN-GRE-Paketen innerhalb des VPN-Tunnels erlauben
     `ufw insert 3 allow proto gre from 10.25.0.0/16`
   - Restliche Kommunikation innerhalb des VPN-Tunnels blockieren
     `ufw insert 4 deny from 10.25.0.0/16`
5. `innernet-server` Dienst aktivieren und starten
   `systemctl enable --now innernet-server@innernet`
   Ein neues Netzwerk-Interface mit Namen `innernet` wird angelegt

[72] Die Konfiguration des innernet-Serverdienstes wird unter `/etc/innernet-server` sowie `/var/lib/innernet-server` angelegt. Somit können diese Verzeichnisse gesichert und bei einem Neuaufbau nach der Installation mit `apt` (Schritt 2) wiederhergestellt werden. Daraufhin ist lediglich wieder der Dienst gemäss Schritt 5 zu starten

[73] Die Adressen der Cyber-Security-Lab-Infrastruktur werden nicht öffentlich bekanntgegeben

6. Client-Konfiguration anlegen (pro Client ausführen)

   - Client-VPN-Netzwerk anlegen
     `innernet-server add-cidr innernet`
     `NETNR` entspricht hier einer aufzählenden Nummer (beim dritten Client ist also `NETNR` = 3 zu wählen)
     Die VPN-Clients sollen untereinander keine Kommunikation aufbauen können[74]
     - `Parent CIDR: innernet (10.25.0.0/16)`
     - `Name: innernet-NETNR`
     - `CIDR: 10.25.NETNR.0/24`
     - `Create CIDR "innernet-NETNR"?: yes`
   - Client-Konfiguration anlegen
     - `mkdir ~/innernet`
     - `cd ~/innernet`
     - `innernet-server add-peer \ innernet`
       - `Eligible CIDRs for peer: innernet-NETNR (10.25.NETNR.0/24)`
       - `IP: 10.25.NETNR.1`
       - `Name`: Client-Bezeichnung, z.B. `client-NETNR-1`
       - `Make client-NETNR-1 an admin?: No`
       - `Invite expires after: 14d`
       - `Save peer invitation file to`: entsprechend Client-Bezeichnung, z.B. `client-NETNR-1.toml`
       - `Create peer client-NETNR-1?: Yes`
     - Die angelegte Datei `~/innernet/client-NETNR-1.toml` ist nun auf den Client zu kopieren und vom VPN-Server zu löschen
       Beispiel für automatisierte Installation mittels Ansible:
       - Die Client-Konfigurationsdatei wird mittels Ansible-Vault [369] verschlüsselt beim Ansible-Playbook unter `ansible/files/innernet_client_config/client-NETNR-1.toml` hinterlegt (siehe Tabelle D.1 in Kapitel D)
       - Diese Datei wird im Inventar des Ansible-Playbooks als Host-spezifische Variable angegeben (siehe Quelltext C.2)

```
specificscantarget:
        innernet_client_configfile: innernet_client_config/client-NETNR-1.toml
        erspan_porttomirror: eth0
```

Quelltext C.2: Angabe der innernet-Client-Konfigurationsdatei sowie das aufzuzeichnende Interface als Host-spezifische Variablen im Inventar des Ansible-Playbooks

[74] Die innernet-Standardkonfiguration erlaubt die Kommunikation der Clients in denselben Netzwerken plus zum automatisch angelegten „infra“-Netzwerk, in welchem der VPN-Server platziert ist [128]

C.2.2. ERSPAN-Entkapslung und -Spiegelung

Damit unter Hedgehog Linux die ERSPAN-Pakete der Clients aufgezeichnet werden können, sind diese zu entkapseln und in einem Interface zu bündeln. Die Netzwerkverkehrsaufzeichnung und -Analyse mit Arkime unter Hedgehog Linux hat ansonsten Probleme beim Interpretieren der ERSPAN-Pakete. Zudem ist es für die Aufzeichnungskonfiguration vonnöten, dass das aufzuzeichnende Interface nach einem Neustart dieselbe MAC-Adresse beibehält.

Hierfür ist wie folgt vorzugehen [154, 156, 370–373]:

- ERSPAN-Entkapslungs-Skript (siehe Kapitel D und Quelltext D.2) unter `/usr/local/bin/erspan-decapsule.sh` abspeichern
  Datei-Berechtigungen definieren: `chmod 755 /usr/local/bin/erspan-decapsule.sh`
  - Dieses Skript hört regelmässig auf eingehende ERSPAN-Pakete auf dem entsprechenden Interface und legt pro Sender ein ERSPAN-Interface an. Dort wird der Traffic entkapselt und zum „ERSPAN-Haupt-Interface“ gespiegelt. Die MTU-Werte werden auf den Interfaces ebenfalls entsprechend definiert (siehe Kapitel 3.1.2)
- systemd-Service unter `/etc/systemd/system/erspan-decapsule@.service` abspeichern (siehe Kapitel D)
  Datei-Berechtigungen definieren: `chmod 644 /etc/systemd/system/erspan-decapsule\@.service`
- Zeilen `ERSPAN_DEV_NAME=myerspan` und `ERSPAN_KEY=10` zur Datei `/etc/environment` hinzufügen
- Angelegter ERSPAN-Entkapslungsdienst aktivieren und starten
  `systemctl enable --now erspan-decapsule@innernet.service`
- Ausgabe des Dienstes kann mittels `systemctl status erspan-decapsule@innernet` oder `journalctl -t erspan-decapsule.sh` geprüft werden

C.2.3. Aufzeichnungs-Konfiguration

Nachdem das „Haupt-Interface“ für ERSPAN angelegt ist, kann dieses als aufzuzeichnendes Interface in Hedgehog Linux hinterlegt werden.

Hierfür gilt es ein internes Skript zu modifizieren, damit auch virtuelle Interfaces angegeben werden können. Dazu ist folgender Befehl mit dem Account `root` auszuführen:

```
sed -i 's/\(available_iface_list.*all_iface_list\)\ if.*$/\1]/g' \
 /usr/local/bin/sensorcommon.py
```

Quelltext C.3: Anpassung von `/usr/local/bin/sensorcommon.py` unter Hedgehog Linux, um auch virtuelle Interfaces als Aufzeichnungsquelle angeben zu können

Nun ist der Aufzeichnungsvorgang als Benutzer `sensor` zu konfigurieren [153]. Dies ist mittels Befehl `/usr/local/bin/configure-capture.py` oder in der grafischen Oberfläche durch Klick auf „Configure Capture and Forwarding“ in der Toolbar oben möglich [153].

Im erscheinenden Menü ist folgendes zu konfigurieren [153]:

- Option `Continue` wählen
- Option `Configure Capture` wählen
  - Konfiguriertes ERSPAN-Haupt-Interface wählen, hier `myerspan`
  - `PCAP capture filter`: leer lassen
  - „PCAP Path“ und „Zeek Log Path“ auf Standardeinstellung belassen (`/home/sensor/net_cap` und `/home/sensor/zeek_logs`
  - `Is the sensor being used to monitor an Operational Technology/ Industrial Control Systems (OT/ICS) network?`: `No`
  - `Specify Zeek file carving mode`: `none`
- Option `Configure Forwarding` wählen
  - Option `ssl-client-receive` wählen
    - Auf Malcolm-Instanz `./Malcolm/scripts/auth_setup` ausführen und `txfwcerts` wählen
    - Unter Hedgehog Linux die IP der Malcolm-Instanz sowie den Code angeben, der auf der Malcolm-Instanz angezeigt wird
    - Zertifikate werden übertragen
  - Option `arkime-capture` wählen
    - `HTTPS` wählen
    - `SSL verification`: `none`
    - `OpenSearch/ElasticSearch Host`: IP der Malcolm-Instanz
    - `OpenSearch/ElasticSearch Port`: `9200`
    - `OpenSearch/ElasticSearch HTTP/HTTPS server username`: Malcolm-Account, der mittels `https://MALCOLM-ADDRESS/auth/` angelegt wurde
      Hier wird auf der Malcolm-Instanz ein neuer Benutzer `sensor` angelegt und angegeben
    - `OpenSearch/ElasticSearch HTTP/HTTPS server password`: Passwort des angegebenen Accounts
    - `Test connection` sollte `OK` anzeigen
    - „Arkime password hash secret“ aus Malcolm-Instanz-Einrichtung angeben
    - `Specify Arkime PCAP compression mode`: `none`
      mit `zstd`-Kompression wirft `arkime-viewer` auf einen Fehler beim Export einer PCAP-Datei

  - Option `filebeat` wählen
    - `Log Path`: `/home/sensor/zeek_logs`
    - `Logstash Host`: IP der Malcolm-Instanz
    - `Logstash Port`: 5044
    - `NetBox site name`: `analysisenv`
    - `Forward to Logstash over SSL?`: `SSL`
    - `Logstash SSL verification`: `None`
    - `SSL Files`: Standardangaben beibehalten
  - Option `miscbeat` wählen
    - `Logstash Host + Port` vordefiniert übernehmen
    - `NetBox site name`: `analysisenv`
    - `Forward to Logstash over SSL?`: `SSL`
    - `Logstash SSL verification`: `None`
    - `SSL Files`: Standardangaben beibehalten
- Option `Configure Autostart Services` wählen
  - Sämtliche Dienste auswählen, ausser `AUTOSTART_NETSNIFF` und `AUTOSTART_TCPDUMP`, da bereits mittels Arkime aufgezeichnet wird

Die Anzahl der Zeek-Worker-Prozesse[75] wird entsprechend vorhandener Prozessor-Ressourcen definiert [141]. Hierzu wird in der Datei `/opt/sensor/sensor_ctl/control_vars.conf` der Wert `ZEEK_LB_PROCS_WORKER_DEFAULT=0` definiert (Variable ist bereits mit Wert `2` definiert, daher ist hier `2` durch `0` zu ersetzen) [141, 374]. Per Standardkonfiguration werden bei `x` CPU-Kernen `x-4` Zeek-Worker angelegt [141].

In derselben Datei (`control_vars.conf`) ist die Paket-Puffergrösse zu erhöhen. Hierzu wird die Angabe `ZEEK_AF_PACKET_BUFFER_SIZE=67108864` mit `ZEEK_AF_PACKET_BUFFER_SIZE=536870912` ersetzt [141, 360]. Jeder Zeek-Worker alloziert diese Puffergrösse, demnach muss genügend Arbeitsspeicher zur Verfügung stehen [141].

Danach ist ein Neustart des Servers durchführen, um die Änderungen zu übernehmen.
Mittels Benutzer `sensor` kann kontrolliert werden, ob die Sensoren funktionsfähig sind: `/opt/sensor/sensor_ctl/status` [153].

[75] Ein Worker in Zeek betreibt u. a. die Analyse des aufgezeichneten Netzwerkverkehrs [374]

C.2.4. Zeek-Skript

Das Zeek-Skript (siehe Kapitel D.3) zur weiteren Auswertung des Netzwerkverkehrs wird auf dem Server mit Hedgehog Linux wie folgt angelegt [116]:

1. Mit `sensor`-Account zu Ordner `/opt/sensor/sensor_ctl/zeek/custom` navigieren
2. Datei `__load__.zeek` mit folgendem Inhalt anlegen:
   `@load ./scannerdetection.zeek`
3. Dateien `knownscannersfqdn.table`, `knownscannersip.table`, `knownscannerssubnet.table`, `loaddata.sh` und `scannerdetection.zeek` nach `/opt/sensor/sensor_ctl/zeek/custom/` kopieren[76]
4. `loaddata.sh` in Crontab von `sensor` hinterlegen
   Mittels `crontab -e -u sensor` folgenden Eintrag hinzufügen
   `0 3 * * * /opt/sensor/sensor_ctl/zeek/custom/loaddata.sh`
   Regelmässige Bereinigung der automatisierten Einträge in `.table`-Dateien in Skript implementiert nach dem Analyse-Zeitraum dieser Arbeit (21. Februar 2025)
   Es werden ausser der Header-Zeile alle Einträge entfernt, die nicht `manualinput` beinhalten
   Siehe Variable `CLEANUP` in Skript
5. Aufzeichnungsprozesse neu starten mittels:
   ```
   /opt/sensor/sensor_ctl/shutdown && sleep 30 && \
   /opt/sensor/sensor_ctl/supervisor.sh
   ```

[76] Genauere Beschreibung zu den Dateien ist in Kapitel D.3 aufgeführt, wie auch eine entsprechende Erstellung

C.3. Scan-Ziel und VPN-Client

Beim Scan-Ziel handelt es sich um eine Debian-Standardinstallation. Hierbei wird der Netzwerkverkehr auf dem Internet-Interface mittels ERSPAN über die VPN-Verbindung an Hedgehog Linux kopiert.

C.3.1. Automatisierte Installation

Die Installation eines Scan-Ziels erfolgt in folgenden Schritten:

- Automatisierte Debian-Betriebssystem-Installation mittels modifizierter Debian-ISO-Datei aus Kapitel D
- Vorbereiten der VPN-Client-Konfigurationsdatei gemäss Kapitel C.2.1
- Konfiguration mittels Ansible (siehe Kapitel D und A.4.10)
  - VPN-Client-Konfigurationsdatei beim Ansible-Playbook ablegen
  - Scan-Ziel im Inventar für das Ansible-Playbook hinterlegen und die Host-Variable `innernet_client_configfile` mit dem Pfad zur VPN-Client-Konfigurationsdatei hinterlegen
  - Verifizieren, dass das neu installierte Scan-Ziel mittels SSH sowie hinterlegtem Schlüssel erreichbar ist und gegebenenfalls entsprechende Anpassungen vornehmen (z.B. Firewall-Regeln oder Netzwerk-Konfiguration)
  - Ansible-Playbook ausführen
    `ansible-playbook playbook.yml --inventory inventory --ask-vault-pass`

Daraufhin ist das Scan-Ziel inklusive folgenden Eigenschaften vorbereitet:

- Port-Mirroring mittels ERSPAN zu hinterlegtem Server inklusive VPN-Client-Konfiguration installiert und gestartet
- SSH-Server-Konfiguration mit Password- und Public-Key-Authentisierung angepasst sowie bestimmte Public-Keys hinterlegt
- Lokale Firewall konfiguriert mit ufw-Standardkonfiguration [160] und SSH-Zugang (TCP-Port 22) nur für hinterlegte IP Adressen geöffnet
- Partitionierung gemäss CIS Debian Linux 12 Benchmark [351]
- Automatische Updates des Betriebssystems konfiguriert [375]

C.3.2. Manuelle Installation

Zur Vollständigkeit werden die minimal nötigen Schritte zum manuellen Aufbau eines Scan-Ziels in diesem Kapitel festgehalten. Bei einer automatisierten Installation werden zusätzlich bestimmte Hardening-Vorkehrungen getroffen, die hier nicht weiter ausgeführt werden (wie SSH-Server-Konfiguration oder weitere Partitionierung [351]). Sofern nicht anders erwähnt, werden sämtliche Operationen in diesem Kapitel unter `root` ausgeführt.

Zeitsynchronisation mittels NTP

Als `root` folgende Befehle zur Konfiguration der Zeit mittels NTP ausführen

- `apt install -y systemd-timesyncd`
- `timedatectl set-ntp true`

Firewall-Konfiguration

Mittels folgenden Befehlen wird die Firewall-Anwendung ufw installiert und konfiguriert [160]:

- `apt install ufw`
- `ufw allow log proto tcp \`
  `from SSHCLIENTIP to any port 22`
- `ufw default deny incoming`
- `ufw enable`
- `ufw reload`
- `ufw status verbose`

`SSHCLIENTIP` ist in diesem Falle die IP Adresse, von welcher das Scan-Ziel via SSH erreichbar sein darf. Der entsprechende Befehl kann auch mehrmals für mehrere IP Adressen ausgeführt werden. Es ist zu vermeiden, dass der Port für jeden aus dem Internet erreichbar ist. Es gilt zudem die ufw-Standardkonfiguration in dessen Manpage zu beachten [160].

VPN-Client

Die VPN-Client-Konfiguration entspricht folgendem Ablauf [128, 368]:

1. Repository für `innernet` hinterlegen
   - `curl -sS https://tommie.github.io/innernet-debian/repository.asc \`
     `| tee /etc/apt/keyrings/github-tommie-innernet.asc >/dev/null`
   - `cat >/etc/apt/sources.list.d/innernet.list <<EOF`
     `deb [signed-by=/etc/apt/keyrings/github-tommie-innernet.asc]`
     `https://tommie.github.io/innernet-debian/debian jammy contrib`
     `EOF`
   - `chmod 644 /etc/apt/keyrings/github-tommie-innernet.asc`
   - `apt update`
2. `innernet` installieren
   `apt install innernet`
3. Die VPN-Client-Konfigurationsdatei aus Kapitel C.2.1 sollte nun auf dem Client zur Verfügung stehen (z.B. `client-NETNR-1.toml`)
4. `innernet` konfigurieren
   `innernet install client-NETNR-1.toml`
   - `Interface name: innernet`
5. `innernet` Dienst aktivieren und starten
   `systemctl enable --now \`
   `innernet@innernet`
   Ein neues Netzwerk-Interface mit Namen `innernet` wird angelegt

Port-Mirroring mittels ERSPAN

Zur Konfiguration des Port-Mirroring mittels ERSPAN ist wie folgt vorzugehen [130, 132, 137–139, 154, 157, 372, 373, 376, 377]:

- ERSPAN-Skript (siehe Kapitel D und Quelltext D.1) unter `/usr/local/bin/erspan.sh` abspeichern
  Das Skript konfiguriert das Port-Mirroring entsprechend nachher definierter Variablen inklusive entsprechender MTU-Werte (siehe Kapitel 3.1.2)[77]
- systemd-Service für ERSPAN-Skript unter `/etc/systemd/system/erspan@.service` abspeichern (siehe Kapitel D)
- Folgende Zeilen zur Datei `/etc/environment` hinzufügen:
  - `ERSPAN_RECEIVER_IP=10.25.0.1`
    Entspricht der IP-Adresse des VPN-Servers (siehe Kapitel C.2.1)
  - `ERSPAN_PORT_TO_MIRROR=enp0s3`
    Zu spiegelndes Interface
  - `ERSPAN_DEV_NAME=myerspan`
  - `ERSPAN_KEY=10`
    Muss denselben Wert wie auf dem Server betragen (siehe Kapitel C.2.2)
- Angelegter ERSPAN-Dienst aktivieren und starten
  ```
  systemctl enable --now \
  erspan@innernet.service
  ```
- Ausgabe des Dienstes mittels `systemctl status erspan@innernet` oder `journalctl -t erspan.sh` prüfen

Ohne fest hinterlegte MTU-Werte können Pakete fragmentiert werden und Interpretationsfehler entstehen (bei Betrachtung der gekapselten ERSPAN-Pakete):

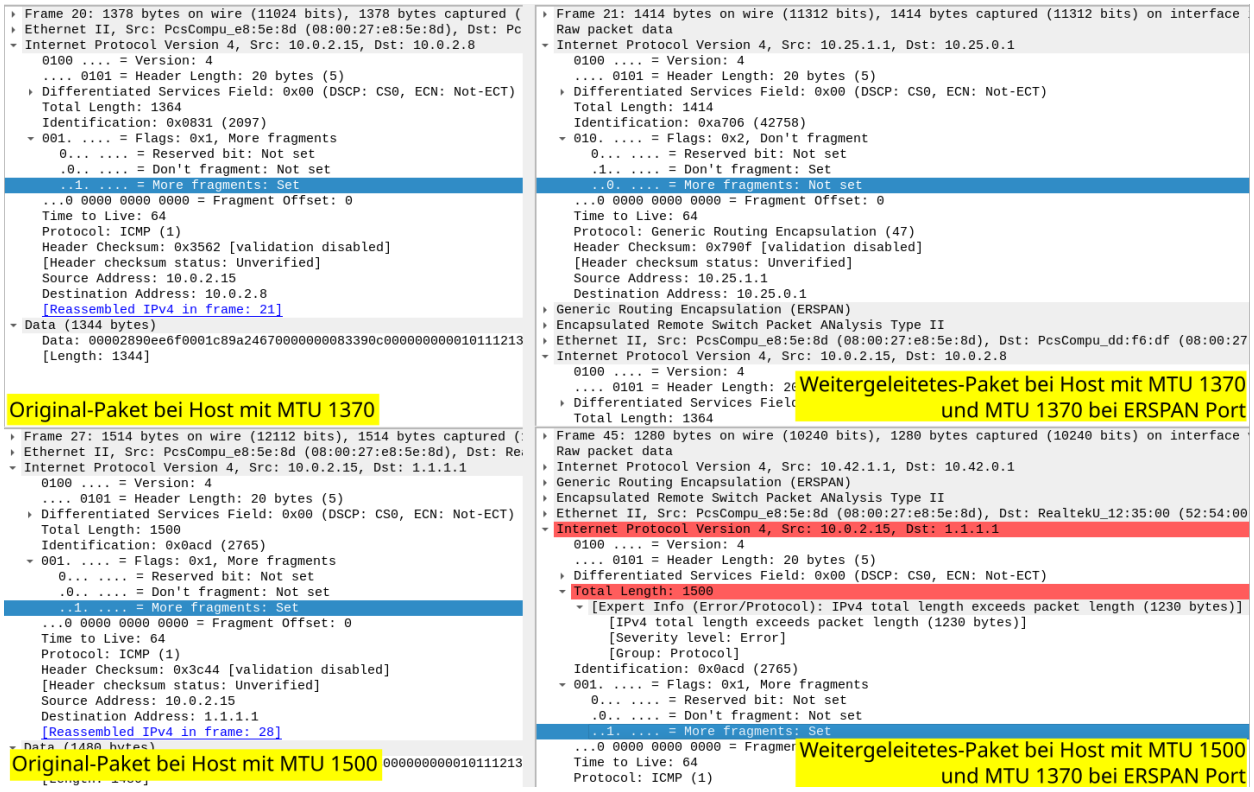


Abbildung C.1.: Verhalten von Wireshark: Via ERSPAN weitergeleitete, bereits fragmentierte Pakete (oben bei ERSPAN-Port nicht fragmentiert, unten wurde das Paket erneut fragmentiert)

[77] Der `--mtu`-Parameter bei innernet scheint aktuell die Angabe nicht zu übernehmen

C.4. Übersicht Analyse-Umgebung

Dieses Kapitel beinhaltet eine Übersicht der Analyse-Umgebung, dessen Scan-Ziele über das Internet erreichbar sind. Die davor lokal aufgebaute Umgebung ist hier nicht aufgeführt. Der Aufbau sowie das Verhalten der Analyse-Umgebung ist in Kapitel 3.1 beschrieben.

Die IP-Adressen der Cyber-Security-Lab-Infrastruktur werden nicht öffentlich bekanntgegeben und daher mit entsprechenden Platzhaltern versehen. Je nach Infrastruktur kann eine öffentliche IP-Adresse direkt einer Netzwerkschnittstelle zugewiesen werden. Andernfalls ist zusätzlich die interne, private IP-Adresse vermerkt.

IPv6-Adressen der Scan-Ziele erhalten am 17. Januar 2025 einen DNS-Eintrag, der mit Ausnahme des Cyber-Security-Lab-Scan-Ziels neben entsprechender Adresse aufgeführt wird (siehe Kapitel 3.4).

> ⚠ Die hierbei verwendete Domäne `pinelair.com` befindet sich seit dem 20. Februar 2025 nicht mehr im Besitz der Autorschaft

Der betrachtete Zeitraum, in welchem die Aufzeichnungen analysiert werden, ist dem Beginn von Kapitel 4 zu entnehmen. Die nachfolgenden Datumsangaben sind wie folgt zu interpretieren:

- Aufbau des Scan-Ziels und Start dessen Port-Mirroring- und Aufzeichnungsvorgangs
- Abbau des Scan-Ziels
- Aufzeichnungsunterbruch

Tabelle C.1.: Analyse-Umgebung-Übersicht

Typ	Infrastruktur / Land	IP-Adressen	Zusatzinformationen
Malcolm-Instanz	Cyber-Security-Lab-Infrastruktur der Berner Fachhochschule (BFH) Schweiz	`LAB-INTERN.200`	Sammelt Daten und stellt Weboberfläche zu dessen Analyse bereit
Appliance / Sensor mit Hedgehog Linux	Cyber-Security-Lab-Infrastruktur der BFH Schweiz	`LAB-INTERN.201` VPN-Server über `LAB-PUBLIC-IPV4-VPN`	Leitet Daten an Malcolm-Instanz weiter VPN-Server (Über das Internet erreichbar via UDP-Port `51820`) Erhält Netzwerkpakete von den Scan-Zielen / VPN-Clients (Port-Mirroring mittels ERSPAN) Zeichnet via ERSPAN erhaltene Netzwerkpakete auf

Scan-Ziel	Alibaba Cloud ECS [205]
	China, Hong Kong
		`8.217.233.46`
		Intern: `10.5.0.111`
		
		`240b:4001:158:d500:a721:8781:950:dae6`
		`pacemaker.pinelair.com`
			Hostname: `st005`
			
			Aufzeichnungszeitraum
			18. Dezember 2024
			19. Februar 2025
			
			Vordefiniertes Debian-Image anstelle von Debian-ISO-Datei verwendet

Analyse-Umgebung-Übersicht Fortsetzung

Typ	Infrastruktur / Land	IP-Adressen	Zusatzinformationen
Scan-Ziel	Amazon EC2 [206] USA	54.193.126.149 Intern: 172.31.2.25 2600:1f1c:4fc:7ae1:b36f:9b5a:74e4:2ac8 treat.pinelair.com	Hostname: st004 Aufzeichnungszeitraum 11. Dezember 2024 19. Februar 2025 Vordefiniertes Debian-Image anstelle von Debian-ISO-Datei verwendet
Scan-Ziel	BFH Cyber-Security-Lab-Infrastruktur Schweiz	LAB-PUBLIC-IPV4-SCANTARGET LAB-PUBLIC-IPV6-SCANTARGET SUBDOMAIN.pinelair.com	Hostname: st001 Aufzeichnungszeitraum 11. Dezember 2024 Scan-Ziel bleibt nach Arbeitsabschluss bestehen
Scan-Ziel	Hetzner Cloud [207] Deutschland	195.201.150.203 2a01:4f8:c0c:6dfb::1 audition.pinelair.com Eine Adresse aus zur Verfügung stehendem /64-IPv6-Netzwerk gewählt	Hostname: st003 Aufzeichnungszeitraum 11. Dezember 2024 19. Februar 2025 14. bis 17. Januar 2025 Aufgrund Timeout nach Neustart bei automatisierter Aktualisierung Vordefiniertes Debian-Image anstelle von Debian-ISO-Datei verwendet
Scan-Ziel	JustHost [209] Russland	195.133.11.199 2a00:b700:3::e8 barterer.pinelair.com	Hostname: st008 Aufzeichnungszeitraum 14. Dezember 2024 19. Februar 2025
Scan-Ziel	Microsoft Azure [208] Australien	20.191.203.224 Intern: 10.1.1.100 2603:1010:3:7::2d cupping.pinelair.com Intern: 2404:f800:8000:122::10 [356]	Hostname: st006 Aufzeichnungszeitraum 13. Dezember 2024 19. Februar 2025 Vordefiniertes Debian-Image anstelle von Debian-ISO-Datei verwendet

Analyse-Umgebung-Übersicht Fortsetzung

Typ	Infrastruktur / Land	IP-Adressen	Zusatzinformationen
Scan-Ziel	Serverspace Cloud Servers [210] Brasilien	92.246.131.45 Keine IPv6 Adresse	Hostname: st007 Aufzeichnungszeitraum 13. Dezember 2024 19. Februar 2025 Vordefiniertes Debian-Image anstelle von Debian-ISO-Datei verwendet
Scan-Ziel	Ultahost VPS [211] Singapur	46.183.27.161 2a02:6ea0:d15c:0000:dead:beef:face:cafe stupor.pinelair.com	Hostname: st009 Aufzeichnungszeitraum 14. Dezember 2024 19. Februar 2025 Vordefiniertes Debian-Image anstelle von Debian-ISO-Datei verwendet
Scan-Ziel	VPS.us [212] Nigeria	176.97.192.247 2a06:f901:4001:0100:0000:0000:ac72:3bbb agnostic.pinelair.com	Hostname: st002 Aufzeichnungszeitraum 11. Dezember 2024 19. Februar 2025 13. bis 17. Januar 2025 Aufgrund Timeout nach Neustart bei automatisierter Aktualisierung
Scan-Ziel	VPSServer.com [213] UK	185.227.109.33 2a06:c5c0:600:3::abcd:1235 backstage.pinelair.com	Hostname: st010 Aufzeichnungszeitraum 18. Dezember 2024 19. Februar 2025 Vordefiniertes Debian-Image anstelle von Debian-ISO-Datei verwendet

Anhang D. Skripts

Dieses Kapitel beinhaltet in dieser Arbeit erstellte Skripts.

Aus Übersichtsgründen werden nur relevante Ausschnitte aufgeführt.

Sämtlicher Code wird dem Auftraggeber übergeben und ist zusätzlich öffentlich einsehbar unter `https://guadm.github.io`. In der nachfolgenden Tabelle werden die zugehörigen Dateien aufgelistet. Zugehörige `README.md`-Dateien mit jeweils weiterführenden Informationen sind mit dem Symbol ℹ markiert.

Ausnahme bilden die Befehle zur Generierung von Netzwerkpaketen in Kapitel 3.1.4, A.1.4 und D.3.2, welche in keiner eigenen Datei abgespeichert sind. Befehle zur Tabellen-Anreicherung für das Zeek-Skript in Kapitel D.3.1 werden in der Datei `zeek/loaddata.sh` zusammengeführt.

Tabelle D.1.: Skript-Übersicht

Pfad	Beschreibung
`ansible/` *`ansible.cfg`*	Ansible-Playbook-spezifische Konfigurationen
`ansible/` *`inventory`*	Ansible-Inventar (Zu konfigurierende Hosts inklusive Host-spezifischen Variablen)
`ansible/` *`monitoring.yml`*	Kleines Ansible-Playbook (< 30 Zeilen) zur Auswertung des derzeitigen Ressourcenverbrauchs auf den Hosts (für manuelle Überprüfung)
`ansible/` *`playbook.yml`*	**Ansible-Playbook** zur Konfiguration der Analyse-Umgebung
`ansible/` *`README.md`* ℹ	Beschrieb des Playbooks inklusive zugehöriger Rollen
`ansible/files/` *`motd`*	„Message of the day“ Textdatei mit Info, dass Host per Ansible konfiguriert wird
`ansible/files/` *`ssh_authorized_keys`*	Liste an SSH-Public-Keys zur Authentisierung
`ansible/files/` `innernet_client_config/`*`*.toml`*	innernet-Konfigurationsdateien (eine Datei pro VPN-Client)
`ansible/group_vars/` *`scantargets.yml`*	Ansible-Gruppen-spezifische Variablen für die im Inventar definierte Gruppe „scantargets“
`ansible/roles/guadm.erspan/` *`README.md`* ℹ	Beschrieb der Ansible-Rolle zur Konfiguration des Port-Mirrorings via ERSPAN auf dem Scan-Ziel
`ansible/roles/guadm.erspan/` `defaults/`*`main.yml`*	Standard-Werte der Ansible-Rolle `guadm.erspan`

Skript-Übersicht Fortsetzung

Pfad	Beschreibung
ansible/roles/guadm.erspan/files/*erspan.sh*	**ERSPAN-Konfigurationsskript** Ausschnitte siehe Quelltext D.1
ansible/roles/guadm.erspan/files/*erspan@.service*	systemd-Service zum ERSPAN-Skript
ansible/roles/guadm.erspan/handlers/*main.yml*	Handlers der Ansible-Rolle guadm.erspan
ansible/roles/guadm.erspan/meta/*main.yml*	Meta-Infodatei der Ansible-Rolle guadm.erspan
ansible/roles/guadm.erspan/tasks/*main.yml*	Tasks der Ansible-Rolle guadm.erspan
ansible/roles/guadm.hardening/*README.md*	Beschrieb der Ansible-Rolle zur Härtung des Scan-Ziels inklusive SSH-Server-Konfiguration und Partitionierung
ansible/roles/guadm.hardening/defaults/*main.yml*	Standard-Werte der Ansible-Rolle guadm.hardening
ansible/roles/guadm.hardening/handlers/*main.yml*	Handlers der Ansible-Rolle guadm.hardening
ansible/roles/guadm.hardening/meta/*main.yml*	Meta-Infodatei der Ansible-Rolle guadm.hardening
ansible/roles/guadm.hardening/tasks/*main.yml*	Tasks der Ansible-Rolle guadm.hardening
ansible/roles/guadm.hardening/tasks/*partitions.yml*	Partitionierung-spezifische Tasks der Ansible-Rolle guadm.hardening
ansible/roles/guadm.hardening/templates/*sshd_config.j2*	Template zur Konfiguration des SSH-Serverdienstes
ansible/roles/guadm.innernet/*README.md*	Beschrieb der Ansible-Rolle zur Einrichtung des VPN-Clients auf dem Scan-Ziel mittels innernet
ansible/roles/guadm.innernet/defaults/*main.yml*	Standard-Werte der Ansible-Rolle guadm.innernet
ansible/roles/guadm.innernet/handlers/*main.yml*	Handlers der Ansible-Rolle guadm.innernet
ansible/roles/guadm.innernet/meta/*main.yml*	Meta-Infodatei der Ansible-Rolle guadm.innernet
ansible/roles/guadm.innernet/tasks/*main.yml*	Tasks der Ansible-Rolle guadm.innernet
ansible/roles/guadm.innernet/templates/*innernet_install.exp.j2*	Template des Skripts zur Einrichtung von innernet

Skript-Übersicht Fortsetzung

Pfad	Beschreibung
`debian-install/` *modify-debian-iso.sh*	Bash-Skript zur **Modifikation einer Debian-ISO-Datei** mit einer Preseed-Datei zur automatisierten Debian-Installation
`debian-install/` *README.md*	Beschreibung des Bash-Skripts zur Debian-ISO-Modifikation
`debian-install/` *scantarget-preseed.cfg*	**Preseed-Datei** zur automatisierten Debian-Installation
`erspan/` *README.md*	Beschrieb des ERSPAN-Entkapselungs-Skript zur Einrichtung auf dem VPN-Server bzw. Appliance / Sensor mit Hedgehog Linux
`erspan/etc/` *environment*	Umgebungsvariablen für das ERSPAN-Entkapselungs-Skript
`erspan/etc/systemd/system/` *erspan-decapsule@.service*	systemd-Service zum ERSPAN-Entkapselungs-Skript
`erspan/usr/local/bin/` *erspan-decapsule.sh*	**ERSPAN-Entkapselungs-Skript** Ausschnitte siehe Quelltext D.2
`opensearchdashboards-objects/` *dash - Scanner Detection*	Dashboard mit u. a. nachfolgend aufgelisteten Visualisierungen
`opensearchdashboards-objects/` *README.md*	Beschreibung der Objekte zum Import in OpenSearch Dashboards
`opensearchdashboards-objects/` *vis - Connections - Destination - Sum of Source Bytes.ndjson*	Summe der Bytes von Quell-Paketen, aufgeteilt nach Ziel-IP-Adresse
`opensearchdashboards-objects/` *vis - Controls.ndjson*	Steuerungsfelder für weitere Filter
`opensearchdashboards-objects/` *vis - Country ISO Code vs Event Provider.ndjson*	Visualisierung (Kuchendiagramm) mit ermittelten Länder-Codes pro Ziel-IP-Adresse und Event-Provider (Arkime und Zeek) Nicht in Dashboard „Scanner Detection“ verwendet, jedoch in Abbildung 3.9
`opensearchdashboards-objects/` *vis - Destination Top 10 IP and Top 3 Ports.ndjson*	Visualisierung (Säulendiagramm) mit Top 10 Ziel-IP-Adressen, unterteilt in jeweils Top 3 Ziel-Ports
`opensearchdashboards-objects/` *vis - Destination Ports Top 20 Scanner portions.ndjson*	Visualisierung (Säulendiagramm) mit Top 20 Ziel-Ports, mit prozentualem Anteil der Scanner pro Port (Benötigt aktiven Filter `rule.category: ScannerDetection`)
`opensearchdashboards-objects/` *vis - Notices - Unique Source IP Count.ndjson*	Zählung eindeutiger Quell-IP-Adressen

Skript-Übersicht Fortsetzung

Pfad	Beschreibung
opensearchdashboards-objects/ *vis - Scanner Detection Top 20 Source IP - By Method.ndjson*	Tabelle mit Top 20 Quell-IP-Adressen mit jeweiliger Detektionsmethode
opensearchdashboards-objects/ *vis - Scanner Detection bad - By Method.ndjson*	Visualisierung (Kuchendiagramm) mit als bösartig eingestuften Scan-Quellen, aufgeteilt in Detektionsmethoden des Zeek-Skripts
opensearchdashboards-objects/ *vis - Scanner Detection good - By Method.ndjson*	Visualisierung (Kuchendiagramm) mit als gut eingestuften Scan-Quellen, aufgeteilt in Detektionsmethoden des Zeek-Skripts
opensearchdashboards-objects/ *vis - Scanner Detection Over Time.ndjson*	Visualisierung (Flächerdiagramm) mit erfolgreich angewandten Detektionsmethoden über einen gegebenen Zeitrahmen
opensearchdashboards-objects/ *vis - Scanner Detection - Combined.ndjson*	Visualisierung (Kuchendiagramm) mit möglichst vielen Informationen anderer Visualisierungen vereint (IP-Version, Detektionsmethode, Quell-AS, Scan-Quellnamen, Protokoll, Ziel-Port)
opensearchdashboards-objects/ *vis - Scanner Detection - IPv4 vs IPv6.ndjson*	Visualisierungen (Kuchendiagramme), aufgeteilt in IPv4 und IPv6 mit guten und bösartigen Verbindungen sowie angewandten Detektionsmethoden
opensearchdashboards-objects/ *vis - Scanner Detection - IPv4 vs IPv6 (unique source ip count).ndjson*	Dieselbe Visualisierung wie die Vorherige mit dem Unterschied, dass jede Quell-IP-Adresse nur einmal gezählt wird
opensearchdashboards-objects/ *vis - Scanner Detection Scanner Names - By good and bad.ndjson*	Visualisierungen (Kuchendiagramme), mit Scan-Quellen, aufgeteilt in gut oder bösartig Zeigt Anteil einzelner Scan-Quellen an
opensearchdashboards-objects/ *vis - Scanner Detection Scanner Names - By good and bad (unique source ip count).ndjson*	Dieselbe Visualisierung wie die Vorherige mit dem Unterschied, dass jede Quell-IP-Adresse nur einmal gezählt wird
opensearchdashboards-objects/ *vis - Scanner Detection Info regarding filtering.ndjson*	Markdown-Text mit Hinweis zu Filter
zeek/ *__load__.zeek*	Datei, die Zeek-Skripts zum Laden auflistet (wird von Zeek beim Laden eines Ordner-Pfads automatisch aufgerufen [116])
zeek/ *loaddata.sh*	Skript zur Anreicherung bekannter Adressen gemäss Kapitel 3.1.3 und D.3.1

Skript-Übersicht Fortsetzung

Pfad	Beschreibung
zeek/ *knownscannersfqdn.table*	Tabelle mit bekannten FQDNs (siehe Kapitel 3.1.3) Wird auf dem Sensor mit Hedgehog Linux unter `/opt/sensor/sensor_ctl/zeek/custom` abgelegt und täglich von `loaddata.sh` erweitert
zeek/ *knownscannersip.table*	Tabelle mit bekannten IP-Adressen (siehe Kapitel 3.1.3) Wird analog zu `knownscannersfqdn.table` von `loaddata.sh` erweitert
zeek/ *knownscannerssubnet.table*	Tabelle mit bekannten IP-Subnetz-Adressen (siehe Kapitel 3.1.3) Wird analog zu `knownscannersfqdn.table` von `loaddata.sh` erweitert
zeek/ *README.md*	Beschreibung des Zeek-Skripts zur Erweiterung der Analysefunktionalitäten (siehe Kapitel 3.1.3)
zeek/ *scannerdetection.zeek*	**Zeek-Skript** zur Erweiterung der Analysefunktionalitäten inklusive Prüfungen des Netzwerkverkehrs entsprechend RFC 9511 und zugehörig vorbereiteten Tabellen (siehe Kapitel 3.1.3) Ausschnitte siehe Quelltext D.3

D.1. ERSPAN auf Scan-Ziel

```
#!/bin/bash
...
erspan_run () {
  ...
  # configure ERSPAN interface
  ip link add dev $ERSPAN_DEV_NAME type erspan seq key $ERSPAN_KEY local
     $ERSPAN_IPSRC remote $ERSPAN_RECEIVER_IP erspan_ver 1 erspan 123
  if [ $? -ne 0 ]; then erspan_error; fi
  ip link set dev $ERSPAN_DEV_NAME promisc on
  if [ $? -ne 0 ]; then erspan_error; fi
  tc qdisc add dev $ERSPAN_PORT_TO_MIRROR handle ffff: ingress
  if [ $? -ne 0 ]; then erspan_error; fi
  tc qdisc add dev $ERSPAN_PORT_TO_MIRROR handle 1: root prio
  if [ $? -ne 0 ]; then erspan_error; fi

  # mirror ingoing traffic
  tc filter add dev $ERSPAN_PORT_TO_MIRROR parent ffff: protocol all u32 match
      u32 0 0 action mirred egress mirror dev $ERSPAN_DEV_NAME
  if [ $? -ne 0 ]; then erspan_error; fi

  # mirror outgoing traffic (COMMENTED OUT, NOT MIRRORED AT THE MOMENT)
  # tc filter add dev $ERSPAN_PORT_TO_MIRROR parent 1: protocol all u32 match
     u32 0 0 action mirred egress mirror dev $ERSPAN_DEV_NAME
  # if [ $? -ne 0 ]; then erspan_error; fi

  if [ $MIRRORING_THROUGH_PORT_TO_MIRROR ]; then
    # if mirror data is being sent through the port to mirror...
    tc qdisc add dev $ERSPAN_DEV_NAME handle 1: root prio
    if [ $? -ne 0 ]; then erspan_error; fi
    # ... drop GRE packets IP protocol 47 ERSPAN on outgoing ERSPAN interface
    tc filter add dev $ERSPAN_DEV_NAME parent 1: protocol ip u32 match ip
       protocol 47 0xff action drop
    if [ $? -ne 0 ]; then erspan_error; fi
    # ... drop UDP packets from and to udp/51820 (wireguard) on outgoing
       ERSPAN interface
    if [ $WIREGUARD_INTERFACE == $ERSPAN_SENDING_INTERFACE ]; then
      tc filter add dev $ERSPAN_DEV_NAME parent 1: protocol ip u32 match ip
         sport $UDP_PORT_TO_EXCLUDE 0xffff match ip protocol 17 0xff action
         drop
      if [ $? -ne 0 ]; then erspan_error; fi
      tc filter add dev $ERSPAN_DEV_NAME parent 1: protocol ip u32 match ip
         dport $UDP_PORT_TO_EXCLUDE 0xffff match ip protocol 17 0xff action
         drop
      if [ $? -ne 0 ]; then erspan_error; fi
    fi

    # ... drop arp on outgoing ERSPAN interface
    tc filter add dev $ERSPAN_DEV_NAME parent 1: protocol arp u32 match u32 0
       0 action drop
    if [ $? -ne 0 ]; then erspan_error; fi
    # ... drop ethernet broadcast dst address on outgoing ERSPAN interface
    tc filter add dev $ERSPAN_DEV_NAME parent 1: protocol 802_3 u32 match u32
       0xffffffff 0xffffffff at 0 match u32 0xffff0000 0xffff0000 at 4 flowid
        1:11 action drop
    if [ $? -ne 0 ]; then erspan_error; fi
```

```
44      # ... drop IPv4 local multicast addresses (RFC3171) on outgoing ERSPAN
            interface
45      tc filter add dev $ERSPAN_DEV_NAME parent 1: protocol ip u32 match ip dst
            224.0.0.0/24 action drop
46      if [ $? -ne 0 ]; then erspan_error; fi
47
48      if [ $IPV6DISABLED -eq 0 ]; then
49        # ... drop link local IPv6 packets (RFC4291) on outgoing ERSPAN
              interface
50        tc filter add dev $ERSPAN_DEV_NAME parent 1: protocol ipv6 u32 match ip6
              dst fe80::/10 action drop
51        if [ $? -ne 0 ]; then erspan_error; fi
52        # ... drop IPv6 link local multicast addresses (RFC4291) on outgoing
              ERSPAN interface
53        tc filter add dev $ERSPAN_DEV_NAME parent 1: protocol ipv6 u32 match ip6
              dst ff02::/16 action drop
54        if [ $? -ne 0 ]; then erspan_error; fi
55      fi
56    fi
57
58    ip link set $ERSPAN_DEV_NAME mtu $ERSPAN_DEV_MTU
59    if [ $? -ne 0 ]; then erspan_error; fi
60    ip link set $ERSPAN_DEV_NAME up
61    if [ $? -ne 0 ]; then erspan_error; fi
62    echo "ERSPAN␣set␣up␣($ERSPAN_IPSRC␣to␣$ERSPAN_RECEIVER_IP␣mirroring␣port␣
          $ERSPAN_PORT_TO_MIRROR␣to␣interface␣$ERSPAN_DEV_NAME␣with␣KEY␣
          $ERSPAN_KEY)"
63
64    # keepalive (ping to ipv6 all nodes or ipv4 gateway), else ERSPAN may drop
          on other side
65    if [ $IPV6DISABLED -eq 0 ]; then
66      echo "keepalive␣by␣sending␣ping␣every␣$KEEPALIVE_PING_INTERVAL_SEC␣seconds
            ␣to␣ff01::1%$ERSPAN_PORT_TO_MIRROR␣(IPv6␣all␣nodes)"
67      ping -i $KEEPALIVE_PING_INTERVAL_SEC ff01::1%$ERSPAN_PORT_TO_MIRROR > /dev
            /null
68      if [ $? -ne 0 ]; then erspan_error; fi
69    else
70      IPV4GW=$(ip route | grep default | grep -o "via\␣[^\␣]*" | head -n 1 | awk
            '{print␣$NF}')
71      echo "keepalive␣by␣sending␣ping␣every␣$KEEPALIVE_PING_INTERVAL_SEC␣seconds
            ␣to␣$IPV4GW␣(IPv4␣gateway␣ip)"
72      ping -i $KEEPALIVE_PING_INTERVAL_SEC $IPV4GW > /dev/null
73      if [ $? -ne 0 ]; then erspan_error; fi
74    fi
75    ...
76  }
77
78
79
80
81
82
83
84
85
86
87
88
```

```
89  erspan_stop () {
90    ...
91    # remove ERSPAN interface
92    tc qdisc del dev $ERSPAN_PORT_TO_MIRROR handle ffff: ingress
93    if [ $? -ne 0 ]; then erspan_error; fi
94    tc qdisc del dev $ERSPAN_PORT_TO_MIRROR handle 1: root prio
95    if [ $? -ne 0 ]; then erspan_error; fi
96
97    if [ $MIRRORING_THROUGH_PORT_TO_MIRROR ]; then
98      tc qdisc del dev $ERSPAN_DEV_NAME handle 1: root prio
99      if [ $? -ne 0 ]; then erspan_error; fi
100   fi
101
102   ip link set $ERSPAN_DEV_NAME down
103   if [ $? -ne 0 ]; then erspan_error; fi
104   ip link del dev $ERSPAN_DEV_NAME
105   if [ $? -ne 0 ]; then erspan_error; fi
106   echo "ERSPAN␣stopped␣(mirroring␣port␣$ERSPAN_PORT_TO_MIRROR␣to␣interface␣
        $ERSPAN_DEV_NAME)"
107   ...
108 }
109
110 erspan_error () {
111   echo "ERROR␣check␣output␣and␣help␣(-h␣or␣--help)"
112   exit 1
113 }
114 ...
```

Quelltext D.1: Ausschnitte aus `ansible/roles/guadm.erspan/files/erspan.sh`: Skript zur Konfiguration von Port-Mirroring mittels ERSPAN [7, 130, 132, 137–139, 154, 157, 372, 373, 376–379]

Um Port-Mirroring auf dem Interface `enp0s3` zu definieren ist demnach wie folgt vorzugehen:

- Einträge in `/etc/environment` hinterlegen
  - Ziel-IP des ERSPAN-Traffics
    (Analyse-Sensor-Server, hier `ERSPAN_RECEIVER_IP=10.25.0.1`)
  - Zu betrachtender Port (hier `ERSPAN_PORT_TO_MIRROR=enp0s3`)
  - Name für das ERSPAN-Interface hinterlegen (hier `ERSPAN_DEV_NAME=myerspan`)
  - `KEY`-Wert definieren
    (muss auf der anderen Seite derselbe Wert sein, hier `ERSPAN_KEY=10`)
- systemd-Service mittels `systemctl enable --now erspan@innernet` aktivieren
  Das hier gewählte Interface (`innernet`) setzt die Quell-IP-Adresse für den ERSPAN-Traffic

D.2. ERSPAN auf Analyse-Sensor

```
#!/bin/bash
...
erspan_decapsule_run () {
  ...

  # set receiving interface to promiscuous mode
  ip link set dev $ERSPAN_RECEIVING_INTERFACE promisc on
  if [ $? -ne 0 ]; then erspan_decapsule_error; fi

  # set up main interface
  ip link add dev $ERSPAN_DEV_NAME type erspan seq key $ERSPAN_KEY local
     $ERSPAN_IPDST remote 127.0.0.1 erspan_ver 1
  if [ $? -ne 0 ]; then erspan_decapsule_error; fi
  ip link set dev $ERSPAN_DEV_NAME address $ERSPAN_DEV_MAC
  if [ $? -ne 0 ]; then erspan_decapsule_error; fi
  ip link set dev $ERSPAN_DEV_NAME promisc on
  if [ $? -ne 0 ]; then erspan_decapsule_error; fi
  ip link set $ERSPAN_DEV_NAME mtu $ERSPAN_DEV_MTU
  if [ $? -ne 0 ]; then erspan_decapsule_error; fi
  ip link set $ERSPAN_DEV_NAME up
  if [ $? -ne 0 ]; then erspan_decapsule_error; fi
  echo "created␣$ERSPAN_DEV_NAME␣as␣main␣ERSPAN␣interface"

  ...

  echo "start␣listening␣for␣new␣ERSPAN␣senders"
  # run in loop until EXIT signal, then trap function happens
  while true; do

    # From INTERFACE get GRE packets sent to IP with GRE protocol ERSPAN 0
       x88be and get unique sender addresses
    # ERSPAN III has GRE protocol number 0x22eb and is not supported at the
       moment
    # Protocol numbers: https://www.iana.org/assignments/protocol-numbers/
       protocol-numbers.xhtml
    # checking for new senders (tcpdump for $TCPDUMP_DURATION seconds)
    ERSPAN_ACTIVE_SENDERS=$(tcpdump -i $ERSPAN_RECEIVING_INTERFACE -n -U -l -G
        $TCPDUMP_DURATION -W 1 -w /dev/null --print "proto␣47␣and␣dst␣
       $ERSPAN_IPDST␣and␣ip[22]␣==␣0x88␣and␣ip[23]␣==␣0xbe" 2>/dev/null | awk
        -F'␣' '!seen[$3]++␣{print␣$3}')

    # check if interface is having errors (due to other side restarting) and
       restart corresponding interface
    ERSPAN_INTERFACES=$(ip link show | grep -Eo "${ERSPAN_DEV_NAME}[^@:\␣]*")
    for INTERFACE in $ERSPAN_INTERFACES; do
      if [ $(cat /sys/class/net/$INTERFACE/statistics/rx_errors) -ne 0 ]; then
        echo "detected␣rx␣errors␣on␣$INTERFACE,␣probably␣due␣to␣restart␣from␣
           other␣side,␣recreating␣interface"
        erspan_decapsule_remove_subinterface $INTERFACE
        ERSPAN_SENDER=$(grep -m 1 "${INTERFACE}=" $RUNTIMEFILE | awk -F'=' '{
           print␣$NF}')
        erspan_decapsule_build_subinterface $INTERFACE $ERSPAN_IPDST
           $ERSPAN_SENDER $ERSPAN_KEY $ERSPAN_DEV_MTU
      fi
    done

```

```
46      # check every found sender IP and create new interface if not already set
47      for ERSPAN_SENDER in $ERSPAN_ACTIVE_SENDERS; do
48        # check if in RUNTIMEFILE and skip for loop if already in it
49        if [ $(grep -xF $ERSPAN_SENDER $RUNTIMEFILE) ]; then
50          break
51        fi
52
53        # add ERSPAN_SENDER to RUNTIMEFILE
54        echo $ERSPAN_SENDER >> $RUNTIMEFILE
55
56        # create new interface
57        # if new sender found, RUNTIMEFILE is longer, therefore we can use the
              linecount as index
58        ERSPAN_DEV_SUB_NAME=${ERSPAN_DEV_NAME}$(grep -v "=" $RUNTIMEFILE | wc -l
              )
59        erspan_decapsule_build_subinterface $ERSPAN_DEV_SUB_NAME $ERSPAN_IPDST
              $ERSPAN_SENDER $ERSPAN_KEY $ERSPAN_DEV_MTU
60        echo "${ERSPAN_DEV_SUB_NAME}=${ERSPAN_SENDER}" >> $RUNTIMEFILE
61      done
62
63      # sleeping until next check for $SLEEPBETWEENLOOPS seconds
64      sleep $SLEEPBETWEENLOOPS
65
66    done
67    exit 0
68  }
69
70  erspan_decapsule_build_subinterface () {
71    ...
72
73    ip link add dev $ERSPAN_DEV_SUB_NAME type erspan seq key $ERSPAN_KEY local
          $ERSPAN_IPDST remote $ERSPAN_SENDER erspan_ver 1 erspan 123
74    if [ $? -ne 0 ]; then erspan_decapsule_error; fi
75    ip link set dev $ERSPAN_DEV_SUB_NAME promisc on
76    if [ $? -ne 0 ]; then erspan_decapsule_error; fi
77    ip link set $ERSPAN_DEV_SUB_NAME mtu $ERSPAN_DEV_MTU
78    if [ $? -ne 0 ]; then erspan_decapsule_error; fi
79    ip link set $ERSPAN_DEV_SUB_NAME up
80    if [ $? -ne 0 ]; then erspan_decapsule_error; fi
81
82    # mirror traffic of ERSPAN_DEV_SUB_NAME to ERSPAN_DEV_NAME
83    tc qdisc add dev $ERSPAN_DEV_SUB_NAME handle ffff: ingress
84    if [ $? -ne 0 ]; then erspan_decapsule_error; fi
85    tc filter add dev $ERSPAN_DEV_SUB_NAME parent ffff: protocol all u32 match
          u32 0 0 action mirred egress mirror dev $ERSPAN_DEV_NAME
86    if [ $? -ne 0 ]; then erspan_decapsule_error; fi
87    tc qdisc add dev $ERSPAN_DEV_SUB_NAME handle 1: root prio
88    if [ $? -ne 0 ]; then erspan_decapsule_error; fi
89    tc filter add dev $ERSPAN_DEV_SUB_NAME parent 1: protocol all u32 match u32
          0 0 action mirred egress mirror dev $ERSPAN_DEV_NAME
90    if [ $? -ne 0 ]; then erspan_decapsule_error; fi
91
92    echo "ERSPAN␣set␣up␣($ERSPAN_SENDER␣to␣$ERSPAN_IPDST␣mirroring␣port␣
          $ERSPAN_DEV_SUB_NAME␣to␣interface␣$ERSPAN_DEV_NAME␣with␣KEY␣$ERSPAN_KEY)
          "
93  }
94
95
```

```
96  erspan_decapsule_remove_subinterface () {
97    INTERFACE=$1
98
99    tc qdisc del dev $INTERFACE handle ffff: ingress 2> /dev/null
100   tc qdisc del dev $INTERFACE handle 1: root prio 2> /dev/null
101   ip link set $INTERFACE down
102   if [ $? -ne 0 ]; then erspan_decapsule_error; fi
103   ip link del dev $INTERFACE
104   if [ $? -ne 0 ]; then erspan_decapsule_error; fi
105   echo "ERSPAN␣removed␣interface␣$INTERFACE"
106 }
107
108 erspan_decapsule_stop () {
109   ...
110
111   # clean up all interfaces named after ERSPAN_DEV_NAME
112   ERSPAN_INTERFACES=$(ip link show | grep -Eo "${ERSPAN_DEV_NAME}[^@:\␣]*")
113   for INTERFACE in $ERSPAN_INTERFACES; do
114     erspan_decapsule_remove_subinterface $INTERFACE
115   done
116   echo "ERSPAN␣decapsule␣stopped␣(interface␣$ERSPAN_DEV_NAME␣and␣subinterfaces
        ␣with␣number␣suffix)"
117
118   # restore MTU of ERSPAN_RECEIVING_INTERFACE from RUNFILE
119   ERSPAN_RECEIVING_INTERFACE_MTU=$(grep -m 1 ${ERSPAN_RECEIVING_INTERFACE}
        $RUNTIMEFILE | awk -F'=' '{print␣$NF}')
120   ip link set $ERSPAN_RECEIVING_INTERFACE mtu $ERSPAN_RECEIVING_INTERFACE_MTU
121   if [ $? -ne 0 ]; then erspan_error; fi
122   echo "restored␣MTU␣of␣$ERSPAN_RECEIVING_INTERFACE␣to␣
        $ERSPAN_RECEIVING_INTERFACE_MTU"
123
124   ...
125 }
126
127 erspan_decapsule_error () {
128   echo "ERROR␣check␣output␣and␣help␣(-h␣or␣--help)"
129   exit 1
130 }
131 ...
```

Quelltext D.2: Ausschnitte aus `erspan/usr/local/bin/erspan-decapsule.sh`: Skript zur Konfiguration der ERSPAN-Interfaces auf dem Server [154, 156, 370–373]

Im Skript wird eine MAC-Adresse für das „ERSPAN-Haupt-Interface“ statisch hinterlegt. Dies hat den Grund, damit Arkime unter Hedgehog Linux bei einem Neustart das Interface finden. Die Interfaces sind virtuell und müssen nach jedem Bootvorgang neu angelegt werden.

Um ERSPAN-Interfaces zu definieren und den ERSPAN-Traffic zu entkapseln ist wie folgt vorzugehen:

- Einträge in `/etc/environment` hinterlegen
  - Name für das ERSPAN-Interface hinterlegen (hier `ERSPAN_PORT_TO_MIRROR=enp0s3`)
    Neu erstellte Interfaces erhalten denselben Namen mit einer Nummer als Suffix
  - `KEY`-Wert definieren
    (muss auf der anderen Seite derselbe Wert sein, hier `ERSPAN_KEY=10`)
- systemd-Service mittels `systemctl enable --now erspan-decapsule@innernet` aktivieren
  Das gewählte Interface (`innernet`) definiert wo der ERSPAN-Traffic eintrifft und analysiert wird

D.3. Zeek-Skript zur Scan-Detektion

```
...
global processed_url: set[string, bool];
...
global rfc9511probedescriptionfile = ".well-known/probing.txt";
...
# ------------------------------------------------------------------------
# CUSTOM EVENTS START
# ------------------------------------------------------------------------
event log_detected_scanner(noticetype: Notice::Type, c: connection, address:
    addr, host: string, name: string, datasource: string, method: string,
    intention: string) {
  ...
  # write to logfile opened in zeek_init()
  NOTICE([$note=noticetype,$uid=c$uid,$id=c$id,$identifier=fmt("%s/%s",
      intention, address),$msg=fmt("%s␣scanner␣on␣%s␣(%s):␣%s␣via␣%s,␣method:␣
      %s", intention, address, host, name, datasource, method)]);
  Log::write(ScannerDetection::LOG, info);
}
# checks if source address (ip and reverse resolved value) is in loaded tables
    (see zeek_init() and global variables)
event detect_scanner_in_table(c: connection) {
  if ( c$id$orig_h in known_scanners_ip ) {
    switch known_scanners_ip[c$id$orig_h]$intention {
      case "good":
        event log_detected_scanner(KnownIP_good, c, c$id$orig_h, unknownfqdn,
            known_scanners_ip[c$id$orig_h]$name, known_scanners_ip[c$id$orig_h
            ]$source, "ip␣address␣in␣known␣scanner␣file", known_scanners_ip[
            c$id$orig_h]$intention);
        break;
      case "bad":
        event log_detected_scanner(KnownIP_bad, c, c$id$orig_h, unknownfqdn,
            known_scanners_ip[c$id$orig_h]$name, known_scanners_ip[c$id$orig_h
            ]$source, "ip␣address␣in␣known␣scanner␣file", known_scanners_ip[
            c$id$orig_h]$intention);
        break;
    }
  }
  else { # LINE ADDED AFTER ANALYSIS (2025-02-21), see section 4.1.6
    # if ip is not known, check if ip is in subnet of knownscannerssubnet.
        table # LINE ADDED AFTER ANALYSIS (2025-02-21), see section 4.1.6
    if ( c$id$orig_h in known_scanners_subnet ) {
      switch known_scanners_subnet[c$id$orig_h]$intention { ... }
    }
  } # LINE ADDED AFTER ANALYSIS (2025-02-21), see section 4.1.6
  when [c] ( local fqdn = lookup_addr(c$id$orig_h) ) {
    if ( fqdn != unknownfqdn ) {
      if ( fqdn in known_scanners_fqdn ) {
        switch known_scanners_fqdn[fqdn]$intention { ... }
      }
      else {
        # check if domain of resolved fqdn is in knownscannersfqdn.table
        for ( entry in known_scanners_fqdn ) {
          if ( entry in fqdn ) {
            switch known_scanners_fqdn[entry]$intention { ... }
          }
        }
```

```
45        }
46      }
47    }
48  }
49  # tries to access the Probe Description URI of RFC 9511, if found logged with
      good intention
50  event detect_probing_url(c: connection, host: string, https: bool) {
51    local urlweb : string;
52    if ( https ) {
53      urlweb = fmt("https://%s/%s", host, rfc9511probedescriptionfile);
54    }
55    else {
56      urlweb = fmt("http://%s/%s",host, rfc9511probedescriptionfile);
57    }
58    # if already processed, don't process again and log again if entry was
        successfully detected
59    if ( [urlweb, T] in processed_url ) {
60      event log_detected_scanner(ProbeAttribution_OutOfBand, c, c$id$orig_h,
          host, "unknown", "Probe Description URI Out-of-Band", fmt("probing.txt
          found (%s, status: %s)", urlweb, "200"), "good");
61      return;
62    }
63    if ( [urlweb, F] in processed_url ) { return; }
64    # try to access .well-known/probing.txt and add to processed_url table
65    when [c,host,urlweb] ( local response = ActiveHTTP::request([$url=urlweb,
        $max_time=5sec]) )  {
66      if ( response$code == 200 ) {
67        # string "contact" (case insensitive) needs to be in answer to filter
            out false-positives
68        if ( "contact" in to_lower(response$body) ) {
69          add processed_url[urlweb, T];
70          event log_detected_scanner(ProbeAttribution_OutOfBand, c, c$id$orig_h,
              host, "unknown", "Probe Description URI Out-of-Band", fmt("
              probing.txt found (%s, status: %s)", urlweb, response$code), "good
              ");
71        }
72        else { add processed_url[urlweb, F]; }
73      }
74      else { add processed_url[urlweb, F]; }
75    }
76    timeout 5sec { add processed_url[urlweb, F]; }
77  }
78  event detect_rfc9511_inband(c: connection, payload: string, datasource: string
      , method: string) {
79    local urisfound : set[string];
80    # extract_email_addrs_set does not respect "mailto:" at beginning and has
        not changeable pattern, using find_all
81    # otherwise without "mailto:" for example cipher specs from SSH could match
82    local mailpattern : pattern = /mailto:[^<,:[:blank:]@]+"@"[^>,;[:blank:]]+/;
83    # Telephone regex (https://ihateregex.io/expr/phone/ modified)
84    local telpattern : pattern = /tel:[\+]?[(]?[0-9]{0,3}[)]?[-\s
        \.]?[(]?[0-9]{3}[)]?[-\s\.]?[0-9]{3}[-\s\.]?[0-9]{4,6}/;
85
86    # get all URIs
87    urisfound += find_all_urls(payload);
88    urisfound += find_all(payload, mailpattern);
89    urisfound += find_all(payload, telpattern);
90
```

```
91    # check found URIs
92    for ( uri in urisfound ) {
93      # RFC9511:
94      # The Probe Description URI must start at the first octet of the payload
            and must be terminated by an octet of 0x00, i.e., it must be null
            terminated.
95      # If the Probe Description URI cannot be placed at the beginning of the
            payload, then it must be preceded by an octet of 0x00.
96      uri = gsub(bytestring_to_hexstr(uri), /0{2}.*$/, "00"); # URI may contain
            other URI after \x00 - cut other URI off to not get doubles
97      local urisplit00 = split_string(uri,/00/);
98
99      for ( urisplit in urisplit00 ) {
100       local uritocheck = urisplit00[urisplit];
101       uritocheck = hexstr_to_bytestring(uritocheck);
102
103       # must start at 1st octet of payload, otherwise must be preceded by 0x00
104       if ( starts_with(payload, uritocheck) ) {
105         if ( |urisplit00| >= 1 ){
106           # if split has happened (if first substring smaller), then it was
                terminated by 0x00)
107           if ( (uritocheck != uri && uritocheck != payload) || (ends_with(
                payload,"00") && uritocheck == payload) ) {
108             if ( (rfc9511probedescriptionfile in uritocheck) || (|
                  extract_email_addrs_set(uritocheck)| == 1) || (|find_all(
                  uritocheck, telpattern)| == 1) ) {
109               event log_detected_scanner(ProbeAttribution_InBand, c,
                    c$id$orig_h, unknownfqdn, "unknown", fmt("Probe Description
                    URI In-Band %s", datasource), fmt("Probe Description URI
                    found in %s payload at beginning (%s)", method, uritocheck),
                     "good");
110             }
111           }
112         }
113       }
114       else {
115         # if uri found in splitted string, then preceding by 0x00 is done
116         local founduri = find_all_urls(uritocheck);
117         founduri += find_all(uritocheck, mailpattern);
118         founduri += find_all(uritocheck, telpattern);
119         if ( |founduri| == 1 ) {
120           for ( item in founduri ) {
121             if ( ((rfc9511probedescriptionfile in item) || (|
                    extract_email_addrs_set(item)| == 1) || (|find_all(item,
                    telpattern)| == 1)) && (count_substr(bytestring_to_hexstr(
                    payload),fmt("00%s", bytestring_to_hexstr(item))) == 1) ) {
122               event log_detected_scanner(ProbeAttribution_InBand, c,
                    c$id$orig_h, unknownfqdn, "unknown", fmt("Probe Description
                    URI In-Band %s", datasource), fmt("Probe Description URI
                    found in %s payload not at beginning (%s)", method, item), "
                    good");
123             }
124           }
125         }
126       }
127     }
128   }
129 }
```

```
130 # ---------------------------------------------------------------------------
131 # CUSTOM EVENTS END
132 # ---------------------------------------------------------------------------
133 # ZEEK EVENTS START
134 # ---------------------------------------------------------------------------
135
136 # executed at start of script
137 event zeek_init() {
138   # open log to scannerdetection logfile
139   Log::create_stream(ScannerDetection::LOG, [$columns = Info, $ev =
        log_scannerdetection, $path="scannerdetection", $policy=log_policy]);
140   # load data from knownscanners.table and recheck continually
141   Input::add_table([$source=fmt("%s/knownscannersip.table", @DIR), $name="
        known_scanners_ip", $idx=IdxIP, $val=Val, $destination=known_scanners_ip
        ,$mode=Input::REREAD]);
142   Input::add_table([$source=fmt("%s/knownscannerssubnet.table", @DIR), $name="
        known_scanners_subnet", $idx=IdxSubnet, $val=Val, $destination=
        known_scanners_subnet,$mode=Input::REREAD]);
143   Input::add_table([$source=fmt("%s/knownscannersfqdn.table", @DIR), $name="
        known_scanners_fqdn", $idx=IdxFQDN, $val=Val, $destination=
        known_scanners_fqdn,$mode=Input::REREAD]);
144 }
145
146 # fired on icmp4 and icmpv6 echo request
147 event icmp_echo_request(c: connection, info: icmp_info, id: count, seq: count,
      payload: string) {
148   event detect_rfc9511_inband(c, payload, "ICMP", "echo␣request");
149 }
150
151 # fired for as soon as connection is removed from memory
152 event connection_state_remove(c: connection) {
153   # check if address is in known scanner tables
154   event detect_scanner_in_table(c);
155   # check if well known url exists at ip address
156   event detect_probing_url(c, addr_to_uri(c$id$orig_h), F);
157   event detect_probing_url(c, addr_to_uri(c$id$orig_h), T);
158   # reverse lookup ip address and use fqdn
159   when [c] ( local fqdn = lookup_addr(c$id$orig_h) ) {
160     if ( fqdn != unknownfqdn ) {
161       event detect_probing_url(c, fqdn, F);
162       event detect_probing_url(c, fqdn, T);
163     }
164   }
165 }
166
167 # fired for all packets to zeek
168 event new_packet(c: connection, p: pkt_hdr) {
169   if ( p?$ip ) {
170     if ( p$ip$id == 54321 ) {
171       event log_detected_scanner(ZMap, c, c$id$orig_h, unknownfqdn, "unknown",
            "ZMap", "IPv4␣ID␣is␣set␣to␣54321,␣assuming␣traffic␣from␣ZMap", "
           good");
172     }
173   }
174 }
175
176
177
```

```
178 # fired on ipv6 packet with extension headers
179 # get data from extension headers (not ipv6 payload)
180 event ipv6_ext_headers(c: connection, p: pkt_hdr) {
181   for ( extheader in p$ip6$exts ) {
182     local ip6header = p$ip6$exts[extheader];
183     local ip6headerdata : vector of ip6_option;
184     local datatocheck : vector of string = vector();
185     local datatocheckfiltered : vector of string = vector();
186     ...
187
188     for ( data in datatocheck ) {
189       if ( |gsub(bytestring_to_hexstr(datatocheck[data]),/^0{3,}$/,"")| != 0 )
              { # lots of these data with just \x00 repeated in testing, exclude
             to lessen load
190         event detect_rfc9511_inband(c, datatocheck[data], fmt("IPv6␣Extension␣
              Header␣(%s)",ip6header$id), fmt("IPv6␣Extension␣Header␣(%s)",
              ip6header$id));
191       }
192     }
193   }
194 }
195
196 # fired on tcp traffic (see variable tcp_content_deliver_all_orig)
197 event tcp_contents(c: connection, is_orig: bool, seq: count, contents: string)
      {
198   event detect_rfc9511_inband(c, contents, "TCP", "TCP");
199 }
200
201 # fired on udp requests (see variable udp_content_deliver_all_orig)
202 event udp_contents(u: connection, is_orig: bool, contents: string) {
203   event detect_rfc9511_inband(u, contents, "UDP", "UDP");
204 }
205
206 # ---------------------------------------------------------------------------
207 # ZEEK EVENTS END
208 # ---------------------------------------------------------------------------
```

Quelltext D.3: Ausschnitte aus `zeek/scannerdetection.zeek`: Zeek-Skript zur Detektion von Scans [117, 164–167, 171, 380]

Ein Beschrieb der Funktionalität kann Kapitel 3.1.3 oder der zugehörigen Datei `README.md` (siehe Tabelle D.1 in Kapitel D) entnommen werden.

D.3.1. Tabellen-Anreicherung für Zeek-Skript

Die nachfolgenden Befehle werden für das Anreichern der Tabellen aus Kapitel 3.1.3 verwendet. Es gilt zu beachten, dass die Spalten mittels Tab[78] separiert sind und in jeder Tabellen-Datei (`*.table`) folgender Header als erste Zeile stehen muss. Diese Befehle werden in der Datei `zeek/loaddata.sh` zusammengeführt.

```
#fields name    address source  intention
```

Quelltext D.4: Header-Zeile einer Tabelle für das Zeek-Skript (Quelltext D.3)

```
git clone https://gitlab.com/mcollins_at_isi/acknowledged_scanners
acknowledged_scanners/bin/asutil.py -d acknowledged_scanners/data pmap \
| grep -E "(\.[0-9]+/32|\:[0-f]+/128)" \
| sed 's/\/128//g␣;␣s/\/32//g' | awk '{printf␣\
"%s\t%s\tgitlab.com/mcollins_at_isi/acknowledged_scanners\tgood\n",␣$2,␣$1}' \
>> knownscannersip.table
```

Quelltext D.5: Anreicherung bekannter Scanner IP-Adressen aus dem von Collins [18]

```
git clone https://gitlab.com/mcollins_at_isi/acknowledged_scanners
acknowledged_scanners/bin/asutil.py -d acknowledged_scanners/data pmap \
| grep -Ev "(\.[0-9]+/32|\:[0-f]+/128)" | grep -E "(\.[0-9]+\/|\:[0-f]\/)" \
| awk '{printf␣\
"%s\t%s\tgitlab.com/mcollins_at_isi/acknowledged_scanners\tgood\n",␣$2,␣$1}' \
>> knownscannerssubnet.table
```

Quelltext D.6: Anreicherung bekannter Scanner IP-Subnetze aus dem von Collins [18]

```
curl https://support.censys.io/hc/en-us/article_attachments/25644686434196 | \
awk '{printf␣"censys\t%s\tsupport.censys.io\tgood\n",␣$1}' \
>> knownscannerssubnet.table
```

Quelltext D.7: Anreicherung bekannter Scanner IP-Subnetze Censys [101]

```
git clone https://github.com/ShadowWhisperer/IPs.git
cat IPs/Other/Scanners | grep -E "(\.[0-9]+|\:[0-f]+)" | grep -v -F '#' \
| awk '{printf␣\
"shadowwhisperer_scanner\t%s\tgithub.com/ShadowWhisperer/IPs\tgood\n",␣$1}' \
>> knownscannersip.table
cat IPs/Malware/Hosting | grep -E "(\.[0-9]+|\:[0-f]+)" | grep -v -F '#' \
| awk '{printf␣\
"shadowwhisperer_malware_hosting\t%s\tgithub.com/ShadowWhisperer/IPs\tbad\n",␣
   $1}' \
>> knownscannersip.table
cat IPs/Malware/Hackers | grep -E "(\.[0-9]+|\:[0-f]+)" | grep -v -F '#' \
| awk '{printf␣\
"shadowwhisperer_malware_hackers\t%s\tgithub.com/ShadowWhisperer/IPs\tbad\n",␣
   $1}' \
>> knownscannersip.table
```

Quelltext D.8: Anreicherung bekannter IP-Subnetze (Scanner und Malware) von ShadowWhisperer [93]

[78] Teilweise wie folgt repräsentiert: `\t`

```
wget -qO - https://threatfox.abuse.ch/export/csv/domains/full/ | zcat \
| tail -n +10 | grep -F "\"domain\"" \
| awk -F',␣' '{gsub("\"","");␣printf␣\
"abuseiocid_%s\t%s\tthreatfox.abuse.ch/export/csv/domains/full\tbad\n",␣$2,␣$3
   }' \
>> knownscannersfqdn.table
wget -qO - https://threatfox.abuse.ch/export/csv/ip-port/full/ | zcat \
| tail -n +10 | grep -F "\"ip:port\"" \
| awk -F',␣' '{gsub("\"","");␣gsub("\:[0-9]+","");␣printf␣\
"abuseiocid_%s\t%s\tthreatfox.abuse.ch/export/csv/ip-port/full\tbad\n",␣$2,␣$3
   }' \
>> knownscannersip.table
```

Quelltext D.9: Anreicherung bekannter Bedrohungen von ThreatFox [94]

```
#!/bin/bash
FILE="knownscannersip.table"
NEXTURL=https://atlas.ripe.net/api/v2/probes/?format=json

while [[ ! -z $NEXTURL ]]; do
DATA=$(curl $NEXTURL)
NEXTURL=$(echo $DATA | jq .next | sed 's/\"//g')
echo $DATA \
| jq '.results[]␣|␣select(.status.id␣==␣1)␣|␣.address_v4,␣.address_v6' \
| grep -v null | sed 's/\"//g' | awk '{printf␣"ripeatlas_probe\t%s\tRIPE␣Atlas
   ␣API\tgood\n",␣$1}' >> $FILE
echo $NEXTURL
done
```

Quelltext D.10: Anreicherung bekannter Scanner von RIPE Atlas API [172, 173]

```
curl https://check.torproject.org/exit-addresses | paste -s - \
| sed 's/\tExitNode/\nExitNode/g' | awk '{printf␣\
"%s_%s\t%s\tTor␣Bulk␣Exit␣List\tbad\n",␣$1,␣$2,␣$10}' \
>> knownscannersip.table
```

Quelltext D.11: Anreicherung bekannter Tor Exit Nodes [174]

D.3.2. Verifikation mittels Generierung von Netzwerkpaketen

Die nachfolgenden Ausschnitte zeigen zusätzlich zur Tabelle 3.1 in Kapitel 3.1.4 Befehle zur Generierung von Netzwerkpaketen mit Scapy [200]. Die Netzwerkpaket-Payload ist meist in Form von Hexadezimal-Werten (`\x00`) definiert. Der lesbare Inhalt befindet sich bei der zugehörigen Befehlsbeschreibung.

Wird keine Quell-Adresse angegeben, wird automatisch die des sendenden Hosts eingetragen. Mit dem `IP`-Argument `src` kann die Quell-Adresse modifiziert werden. Bei der Ziel-Adresse handelt es sich um ein Scan-Ziel, das aktiv Port-Mirroring zum Sensor mit Hedgehog Linux betreibt.

Weiteres zur Detektion ist in Kapitel 3.1.3 sowie zur Verifikation in Kapitel 3.1.4 aufgeführt.

Tabelle D.2.: Weitere Verifikation des Zeek-Skripts mittels Test-Netzwerkpaketen [200–202] (Der gemäss RFC 9511 verwendete Hexadezimal-Wert `0x00` wird in der Payload-Beschreibung so beibehalten (siehe Liste in Kapitel 3.1.3))

Beschreibung und Befehl	Detektion ✅ = Erfolgreiche Detektion ❌ = Keine Detektion
ICMPv6 Echo-Request mit falsch angegebener `probing.txt`-URI, Telefonnummer und E-Mail-Adresse nicht zu Beginn der Payload `0x00` nur vor Telefonnummer und E-Mail-Adresse `(echo "from scapy.all import *"; echo 'sr(IPv6(dst="fe80::a00:27ff:fe95:bb91")/ICMPv6EchoRequest()/"\x68\x65\x6c\x6c\x6f\x20\x77\x6f\x72\x6c\x64\x68\x74\x74\x70\x3a\x2f\x2f\x74\x68\x69\x73\x2e\x69\x73\x2e\x75\x72\x6c\x2f\x2e\x77\x65\x6c\x6c\x2d\x6b\x6e\x6f\x77\x6e\x2f\x70\x72\x6f\x62\x69\x6e\x67\x2e\x74\x78\x74\x00\x74\x65\x6c\x3a\x2b\x31\x2d\x32\x30\x31\x2d\x35\x35\x35\x2d\x30\x31\x32\x33\x00\x6d\x61\x69\x6c\x74\x6f\x3a\x6d\x61\x69\x6c\x40\x65\x78\x61\x6d\x70\x6c\x65\x2e\x63\x6f\x6d", timeout=3)') | python3` Quelltext D.12: Test-Netzwerkpaket: ICMPv6 Echo-Request mit Payload `hello world http://this.is.url/.well-known/probing.txt0x00 tel:+1-201-555-01230x00mailto:mail@example.com`	In-Band Probe Attribution mit `probing.txt`-URI ❌ Nicht detektiert, da der Wert `0x00` nicht davor platziert ist Telefonnummer ✅ E-Mail-Adresse ✅ Zweifache Detektion
IPv6-Paket mit E-Mail-Adresse zu Beginn der PadN-Option des IPv6-„Hop-by-Hop“-Extension-Headers und `probing.txt`-URI nicht zu Beginn der PadN-Option des IPv6-„Destination-Options“-Headers Ersteres terminiert mit `0x00`, letzteres mit `0x00` vor Angabe `(echo "from scapy.all import *"; echo 'sr(IPv6(dst="fe80::a00:27ff:fe95:bb91")/IPv6ExtHdrHopByHop(options=PadN(optdata="\x6d\x61\x69\x6c\x74\x6f\x3a\x6d\x61\x69\x6c\x40\x65\x78\x61\x6d\x70\x6c\x65\x2e\x63\x6f\x6d\x00"))/IPv6ExtHdrDestOpt(options=PadN(optdata="\x70\x00\x68\x74\x74\x70\x3a\x2f\x2f\x65\x78\x61\x6d\x70\x6c\x65\x2e\x63\x6f\x6d\x2f\x2e\x77\x65\x6c\x6c\x2d\x6b\x6e\x6f\x77\x6e\x2f\x70\x72\x6f\x62\x69\x6e\x67\x2e\x74\x78\x74"))/UDP(), timeout=3)') | python3` Quelltext D.13: Test-Netzwerkpaket: IPv6-Paket mit Payload `mailto:mail@example.com0x00` in PadN-Option des IPv6-„Hop-by-Hop“-Extension-Headers und Payload `p0x00http://example.com/.well-known/probing.txt` in PadN-Option des IPv6-„Destination-Options“-Headers	In-Band Probe Attribution mit E-Mail-Adresse ✅ `probing.txt`-URI ✅ Zweifache Detektion

Weitere Verifikation des Zeek-Skripts mittels Test-Netzwerkpaketen Fortsetzung

Beschreibung und Befehl	Detektion ✅ = Erfolgreiche Detektion ❌ = Keine Detektion
TCP-Segment mit `probing.txt`-URI zu Beginn der Payload Terminiert mit `0x00` `(echo "from scapy.all import *"; echo 'sr(IP(dst ="10.0.2.22")/TCP(dport=56789)/"\x68\x74\x74\x70\ x3a\x2f\x2f\x65\x78\x61\x6d\x70\x6c\x65\x2e\x63\x6f \x6d\x2f\x2e\x77\x65\x6c\x6c\x2d\x6b\x6e\x6f\x77\ x6e\x2f\x70\x72\x6f\x62\x69\x6e\x67\x2e\x74\x78\x74 \x00", timeout=3)') | python3` Quelltext D.14: Test-Netzwerkpaket: TCP-Segment mit Payload `http://example.com/.well-known/probing.txt0x00`	In-Band Probe Attribution mit `probing.txt`-URI ✅
TCP-Segment mit falscher URI zu Beginn der Payload Terminiert mit `0x00`, jedoch keine URI mit `.well-known/probing.txt` `(echo "from scapy.all import *"; echo 'sr(IP(dst ="10.0.2.22")/TCP(dport=56789)/"\x68\x74\x74\x70\ x3a\x2f\x2f\x65\x78\x61\x6d\x70\x6c\x65\x2e\x63\x6f \x6d\x2f\x2e\x77\x65\x6c\x6c\x2d\x6b\x6e\x6f\x77\ x6e\x2f\x73\x65\x63\x75\x72\x69\x74\x79\x2e\x74\x78 \x74\x00", timeout=3)') | python3` Quelltext D.15: Test-Netzwerkpaket: TCP-Segment mit Payload `http://example.com/.well-known/security.txt0x00`	In-Band Probe Attribution mit `probing.txt`-URI ❌ Nicht detektiert, da die URI nicht `.well-known/probing.txt` beinhaltet

Anhang E. Zusätzliche Abbildungen

In diesem Kapitel werden zusätzliche Abbildungen aufgeführt.

E.1. Kontaktierte Ports

Abbildungen dieses Kapitels werden von Kapitel 4.4 referenziert.

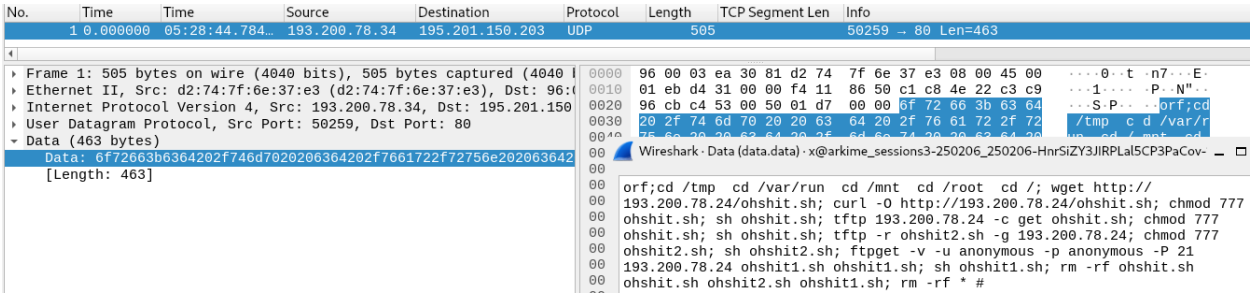


Abbildung E.1.: Shell-Skript in UDP-Payload zu Scan-Ziel `st003` auf Ziel-Port 80

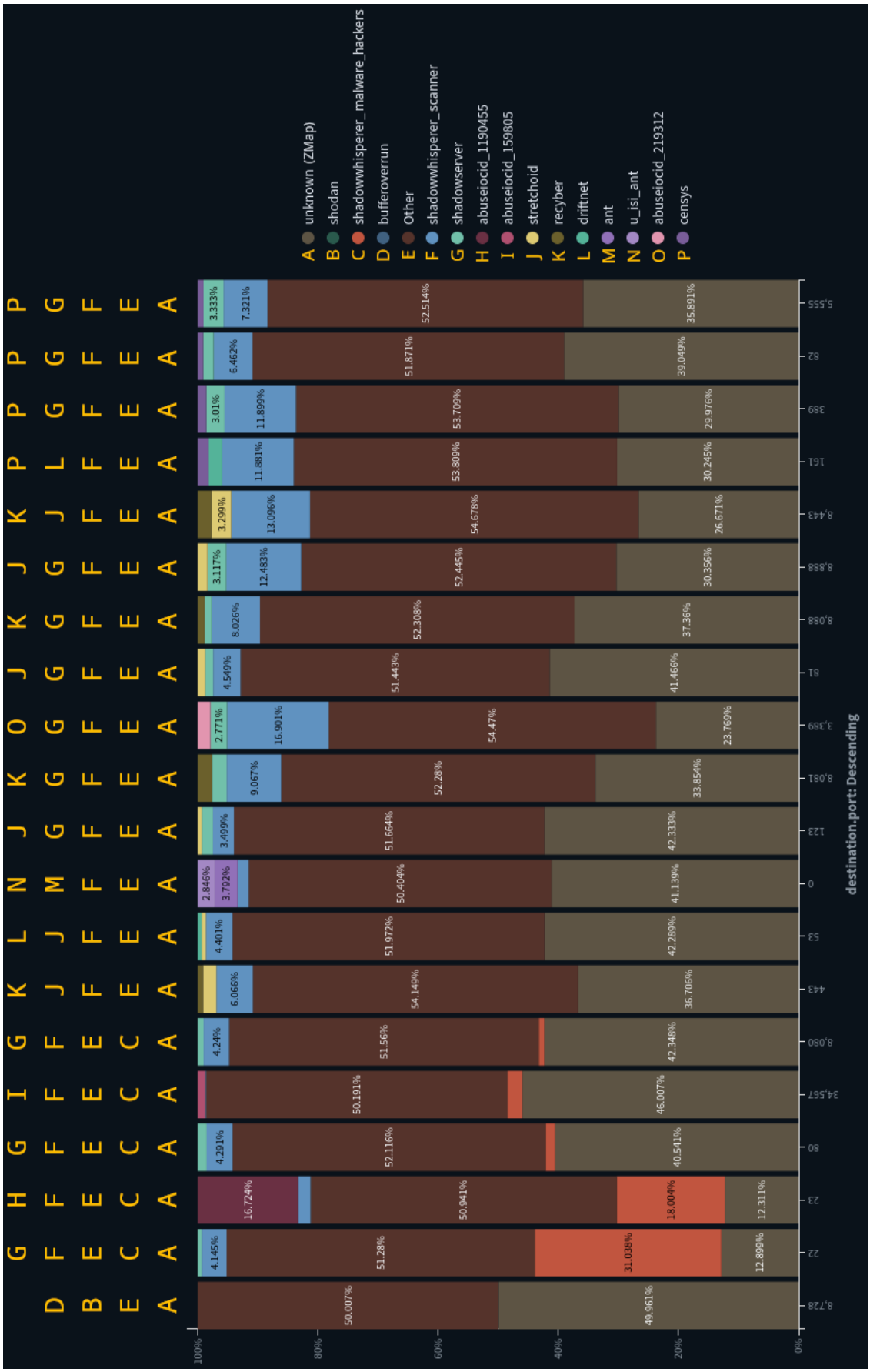


Abbildung E.2.: Anteile der detektierten Scanner (Top 4, Rest gebündelt in „Other“) an meist aufgerufenen Ports bei detektierten Scans (Visualisierung „Destination Ports Top 20 Scanner portions“, modifizierte Darstellung ohne Inhaltsanpassung)

E.2. Dashboard „Scanner Detection“

Die beiden nachfolgenden Abbildungen zeigen das im Rahmen dieser Arbeit erstellte Dashboard „Scanner Detection“ in OpenSearch Dashboards. Entsprechende Objekte werden mit Malcolm mitgeliefert oder sind dieser Arbeit beigelegt (siehe Kapitel D).

Abbildung E.3.: Dashboard „Scanner Detection“ in OpenSearch Dashboards (Teil 1 von 2)

Bei den beiden Spitzen in der Visualisierung „Scanner Detection Over Time“ oben rechts in Abbildung E.3 handelt es sich um Scan-Detektionen des Typs `KnownDomain_good`. Bei der ersten Spitze in der Mitte der Visualisierung handelt es sich um Anfragen von research-scanner [183] zum Scan-Ziel `st002` 🇳🇬. Die zweite Spitze gegen Ende der Visualisierung zeigt Anfragen von CyberOK [190] zu `st008` 🇷🇺. Hierbei scheint es sich um Scans mit TCP-SYN-Segmenten zu handeln, die viele verschiedene Ports in einer kurzen Zeit kontaktieren.

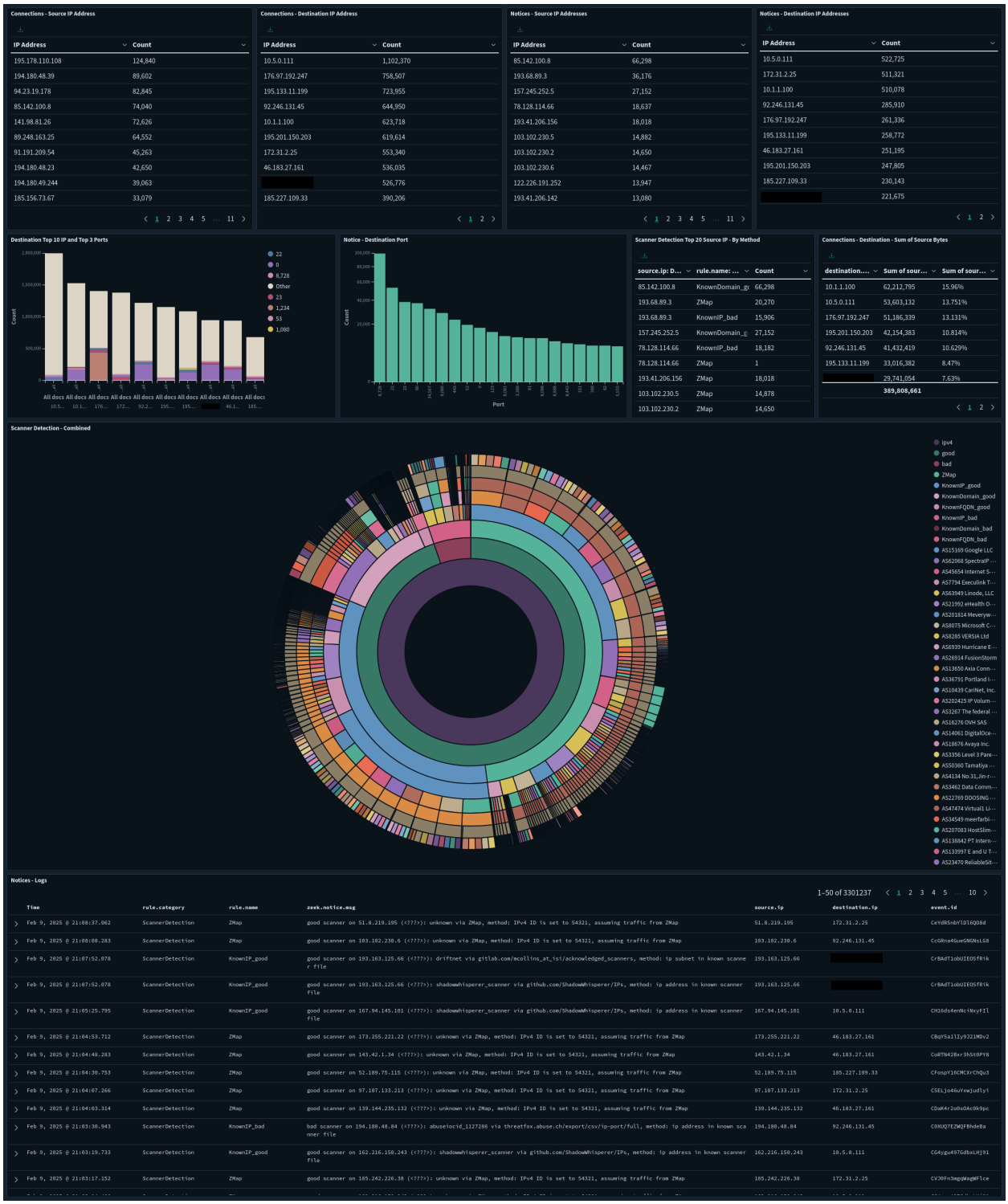


Abbildung E.4.: Dashboard „Scanner Detection“ in OpenSearch Dashboards (Teil 2 von 2)