HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Proxy Redirection with Fake C&C

Bachelorarbeit im Herbstsemester 2016

Fish Tank Suite

Autoren
Silvan Adrian
Fabian Binna

Betreuer
Ivan Bütler

Experte
Daniel Frei

Gegenleser
Prof. Dr. Andreas Rinkel

Abstract

Unter dem Begriff Advanced Persistent Threat (APT) versteht man gezielte und langfristig geplante Cyber Attacken, so wie es das Eidgenössische Departement für auswärtige Angelegenheiten (EDA) und die RUAG im Jahre 2016 erlebten. Wenn eine APT Attacke identifiziert wird, dann stehen zeitraubende Analysearbeiten an, während dessen der Angriff weiter fortschreitet. Es stellt sich die Frage, wie der Angriff verhindert oder zumindest verzögert werden kann, ohne dass die Täterschaft dieses Eingreifen bemerkt.

Diese Bachelor Arbeit stellt eine Methodik und Tool vor, wie der Schaden im Unternehmen reduziert und gleichzeitig die Chancen für ein Unerkannt bleiben bei der Täterschaft erhöht werden kann. Im Wesentlichen handelt es sich um ein Verfahren um Zeit zu gewinnen, damit die APT Attacke im Detail untersucht werden kann ohne weiteren Schaden zu riskieren.

Das Resultat der Arbeit umfasst eine Software, die es erlaubt, durch Trojaner (Malware) infizierte Clients im Unternehmensnetzwerk zu erkennen und stillzulegen. Die Malware ist aus Sicht des Angreifers funktionsfähig, in der Tat ist die Malware jedoch im Sleep-Mode, ohne dass dies vom Angreifer bemerkt wird. Somit gewinnt das betroffene Unternehmen Zeit, um den Fall im Detail zu klären und die Gefahr, die durch die Malware für das Unternehmen entsteht, abzuwenden.

Management Summary

1. Ausgangslage

Die Bedrohung von Industriespionage und Hackerangriffen durch Geheimdienste und organisierte Hackergruppen ist in den letzten Jahren gestiegen, da die Welt immer weiter vernetzt wird und so mehr Möglichkeiten für solche Angriffe bestehen. Gezielte Angriffe mit klaren Motivationen, wie die Industriespionage, setzen auf sogenannte Advanced Persistent Threat (APT) Attacken. Bei solchen Angriffen kann es Monate bis Jahre dauern, bis der Angreifer Zugriff auf das System erlangt. Der Sinn solch langwieriger Prozeduren ist die Verschleierung des Angriffs. Durch das sehr langsame Kompromittieren des Opfers ist die Wahrscheinlichkeit geringer, entdeckt zu werden. Bei der RUAG und dem Eidgenössischen Departement für auswärtige Angelegenheiten (EDA) wurde eine APT Attacke erst nach circa 14 Monaten erkannt, das zeigt wie schwierig es ist, solche Attacken zu erkennen. Wenn so eine Attacke aufgedeckt wird, fallen komplexe Analysearbeiten an, währenddessen schreitet der Angriff weiter fort. Es besteht somit Bedarf, die weitere Kompromittierung des Systems zu verhindern oder zumindest zu verzögern.

2. Vorgehen

Die Phase, bei dem das System unterwandert wird, um später Informationen zu exfiltrieren, läuft relativ simpel ab. Eine Software (Malware), die in eine Organisation eingeschleust wurde, sendet in zeitlich grossen Abständen (z.B. 90 Tage) Informationen über den Sicherheitszustand der Organisation an einen sogenannten Command & Control (C&C) Server. Dieser C&C Server entscheidet, ob die Organisation mit dem aktuellen Sicherheitszustand weiter kompromittiert werden kann. Falls dem so ist, antwortet der C&C Server mit Befehlen, die die Malware ausführt.

Das Ziel der Arbeit besteht darin, Zugriffe auf einen C&C Server zu erkennen und diese auf einen Fake C&C Server umzuleiten. Der Fake C&C Server kann den C&C Server des Angreifers sowie die Malware so täuschen, dass der Angreifer im Glauben gelassen wird, die Malware funktioniere noch. In Wirklichkeit wird aber keiner der Befehle des C&C Servers der Angreifer ausgeführt, da diese Befehle aus den Antworten des C&C Servers herausgefiltert werden.

3. Ergebnisse

Aus der Entwicklung von zwei Prototypen entstand die Fish Tank Suite. Sie erhielt ihren Namen durch die verschiedenen Systemelemente, denen Namen von Fischen zugeteilt wurden. Die Fish Tank Suite zeichnet den HTTP/HTTPS Verkehr auf und speichert ihn in eine Datenbank. Die gewonnenen Daten können mit weiteren Informationen angereichert werden. Es wird zum Beispiel nach verschlüsselten Datenpaketen gesucht, die dann entschlüsselt werden. Die Informationen (Schlüssel) um die

Datenpakete zu entschlüsselt müssen vorher durch ein Team aus Spezialisten durch eine Analyse des Angriffs erarbeitet werden. Die angereicherten Daten helfen dabei, Zugriffe auf C&C Server zu erkennen und umzuleiten. Für die Erkennung der C&C Zugriffe wurden Suchstrategien entwickelt, die auf spezifische Eigenschaften solcher C&C Zugriffe zielen. Diese Suchstrategien durchsuchen die Datenbank und setzen eine Umleitung anhand der Zieladresse des C&C Servers der Angreifer.

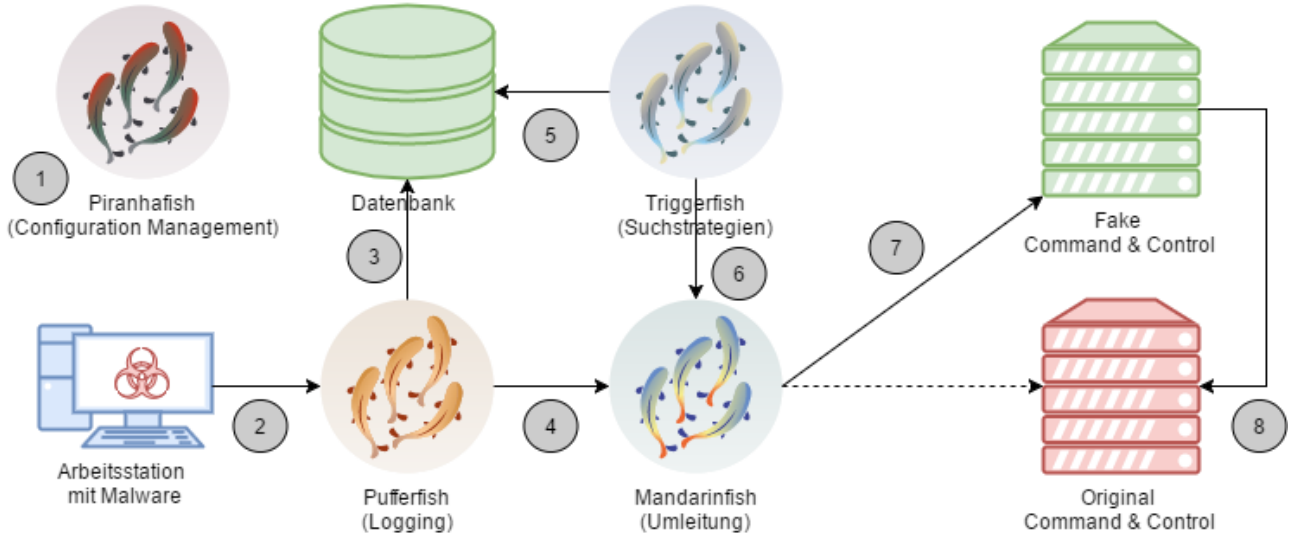


Abbildung 1.: Management Summary: Ergebnis Fish Tank Suite

1. Über den Piranhafish werden alle Instanzen konfiguriert, so dass die Malware erkannt und umgeleitet werden kann.
2. Die Malware möchte dem Original C&C Server Informationen über den Sicherheitszustand der Arbeitsstation, auf der er installiert ist, senden. Alle Datenpakete, die ins Internet gehen, müssen aber am Pufferfish vorbei.
3. Der Pufferfish speichert die Anfrage der Malware in einer Datenbank. In der Datenbank wird bei Bedarf der Inhalt des Datenpakets entschlüsselt.
4. Die Anfrage der Malware wird aber trotzdem weitergesendet, da noch nicht klar ist, ob es sich um ein ganz normales Datenpaket handelt.
5. Der Triggerfish durchsucht die Datenbank mit vordefinierten Suchstrategien.
6. Wenn der Triggerfish das Datenpaket der Malware erkannt hat, teilt er dem Mandarinfish mit, dass er die Verbindung der Malware umleiten soll.
7. Die Verbindung der Malware wird vom ursprünglichen Ziel, dem Original C&C Server, auf den Fake C&C Server umgeleitet.
8. Der Fake C&C Server leitet das Datenpaket der Malware an den Original C&C Server weiter, damit kein Verdacht geschöpft werden kann. Die Antwort des Original C&C Server kann vom Fake C&C Server so verändert werden, dass die Malware weiterschläft und somit keine gefährlichen Aktionen mehr durchführt.

Die Fish Tank Suite ist in der Lage eine Malware zu erkennen, umzuleiten und den Originalen C&C zu täuschen, somit ist der Bedarf, den weiteren Kompromittierungsprozess zu verhindern, erfüllt.

4. Ausblick

Die Entwicklung der Fish Tank Suite befindet sich in einem frühen Stadium. Diverse Anforderungen gehörten nicht zum Anforderungskatalog, die in einem realen Softwareprojekt dazu gehören. Trotzdem läuft das System stabil und eignet sich für eine Weiterentwicklung sehr. Die Suchstrategien könnten auch über ein öffentliches Repository ausgetauscht werden, so dass auch andere Organisationen, die mit denselben Problemen konfrontiert sind, davon profitieren können. Darüber hinaus wären auch präventive Installationen solcher Suchstrategien denkbar.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

HS2016

BA

September 30., 2016

Document Name:	Aufgabenstellung_BA_Fake_C_AND_C.docx
Version:	v1.0
Author	Ivan Buetler
Date of Delivery:	September 30., 2016
Classification:	BA

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

Table of Content

HSR Hochschule für Technik
Postfach 1475
CH-8640 Rapperswil
www.hsr.ch

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

1 Ausgangslage

1.1 APT Angriffe

Im Kontext von APT versuchen Cyber Kriminelle und Intelligence Services geheime Informationen aus dem Firmennetz ins Internet zu exfiltrieren. Dies wird oft auch als Command und Control (C&C) Verbindung bezeichnet und bei professionellen Angreifern mehrschichtig eingesetzt. Das zwingt die Unternehmen, sich mit dem Verhalten des Netzwerkverkehrs von innen nach aussen auseinander zu setzen und sich zu überlegen, wie man bösartigen Traffic erkennt und damit umgeht.

Ein möglicher Ansatz für die Erkennung von C&C Verkehr besteht in der intelligenten Korrelation von Log Daten. Mit Software wie Splunk oder Elasticsearch stehen Tools zur Verfügung, die sich hierfür anbieten. Grundsätzlich geht es darum die echten Log Daten des Unternehmen (DNS, Proxy, Firewall, DHCP) mit Indikatoren aus dem Internet (IP Reputation, Virustotal, Malware.lu, Realtime Blacklist) abzugleichen und die Fähigkeit zu erwerben, mit geeigneten Splunk Abfragen die Malware zu erkennen.

Das ist aber noch nicht genug. Wie geht man damit um, wenn eine Malware erkannt wird? Wie kann man die infizierten Maschinen im Glauben lassen, dass der Angriff noch nicht erkannt wurde? Wie sieht eine Lösung aus, welche die Umleitung von C&C Traffic auf einen Fake Server ermöglicht?

1.2 Auftrag

Im Rahmen von dieser HSR Bachelor Arbeit soll eine Methodik mit Tool entwickelt werden, um bei Befall von APT besser reagieren zu können. Das Tool soll den C&C Traffic analysieren und an einen Fake C&C weiterleiten können. Einige weiterführenden Details und Anforderungen findet man im nächsten Kapitel.

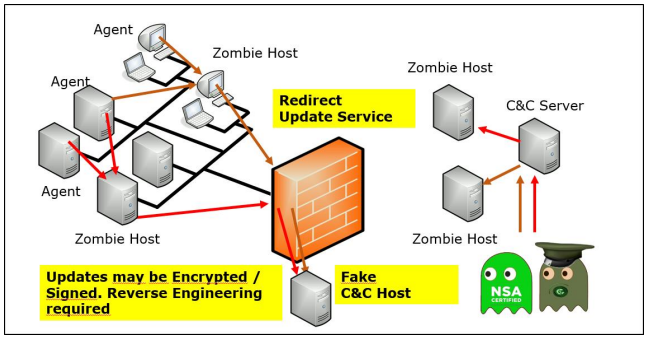
1.3 Voraussetzung

Das System soll in einem Umfeld eingesetzt werden, welche einen sogenannten SSL Splitting Proxy betreibt. Das sind spezielle Content Filter Proxies (Bluecoat, WebWasher, u.ä) welche den SSL Verkehr zwischen Client und einer SSL/TLS Anwendung analysieren um den Inhalt auf Virus und Trojaner zu untersuchen. Das System funktioniert analog einem Inspection Proxy wie ZAP, Burp oder auch dem Linux mitm Tool.

HSR Hochschule für Technik
Postfach 1475
CH-8640 Rapperswil
www.hsr.ch

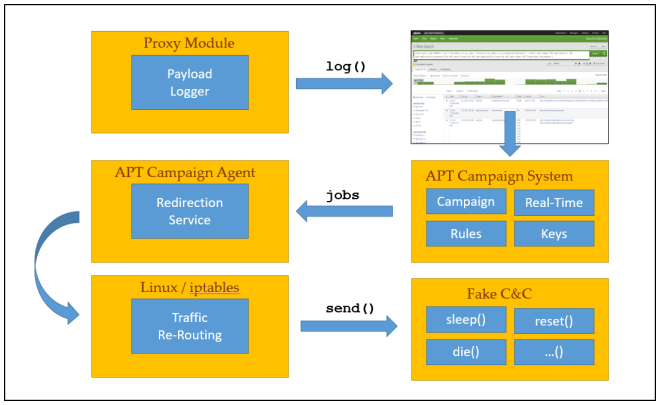
HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

1.4 Beispiel



Im Beispiel oben versuchen die infiszierten Clients via Firewall (SSL Splitting Proxy) auf den C&C im Internet zuzugreifen. Da die C&C Verbindung eine gewisse Charakteristik aufweist, sollen die Verbindungen zum C&C zum Fake C&C rerouted werden. Alle anderen Verbindungen vom Client in das Internet sollen nicht über den Fake C&C laufen.

1.5 Möglicher System Ansatz



Der Payload Logger schickt die Payloads der Verbindungen zu einem Splunk ähnlichen System. Das APT Campaign System analysiert diese in Real-time und versucht den C&C Traffic zu entschlüsseln. Hierzu kann der Kampagne Schlüssel und Algorithmen hinzugefügt werden. Aus der Analyse der Payloads generieren sich Jobs, welche beim APT Campaign Agent zu einer Redirection führen.

HSR Hochschule für Technik
Postfach 1475
CH-8640 Rapperswil
www.hsr.ch

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

1.6 Mögliche Herausforderungen

Die Verbindungen zwischen infisziertem Client und dem C&C können verschlüsselt sein. Möglicherweise ist der Payload auch mehrfach und mit unterschiedlichen Algorithmen verschlüsselt. Um eine Redirection zu initiieren, sind die TCP/IP 3-Way Handshakes als auch die SSL Verbindungen zu berücksichtigen, die allenfalls schon aufgebaut oder neu initiiert werden müssen. Der Fake C&C muss die Logik des C&C kennen, um diesen imittieren zu können.

1.7 Erwartetes Ergebnis

Die Bachelor-Arbeit soll im theoretischen Teil analytisch beleuchten und analysieren, wie die Lösung einer Fake C&C Umgebung umgesetzt werden könnte, unter Berücksichtigung der obig dargestellten Ausgangslage. Hierbei wird eine wissenschaftliche Vorgehensweise und Dokumentation erwartet.

Im praktischen Teil soll eine funktionierende PoC Lösung entwickelt, getestet und bewertet werden. Diese soll im Kontext einer bereits vorliegenden Malware mit C&C einsatzbereit sein. Die Malware selbst mit C&C ist demnach nicht Bestandteil der Aufgabe. Die Test-Malware kann über einen HTTP-Proxy arbeiten und hat für sich selbst auch Payload Encryption aktiviert. Die Malware ist für Microsoft Windows ausgelegt.

Die Testing Ergebnisse müssen für den Leser der Bachelor Arbeit nachvollziehbar sein. Das heisst, der Setup der Tests, die Details des Tests, die Darstellung von erwarteten und getesteten Testpunkten müssen ersichtlich und nachvollziehbar sein. Idealerweise sind die Tests mit den Use-Cases der Lösung verlinkt.

Darüber hinaus gelten die von der HSR publizierten Anforderungen an die Dokumentation einer BA-Arbeit.

1.8 Vertraulichkeit

Sämtliche Informationen über die Malware sind als vertraulich gekennzeichnet und dürfen zu keiner Zeit öffentlich oder Dritten zugänglich gemacht werden. Die Ergebnisse der BA-Arbeit mit dem theoretischen und praktischen Teil sind nicht vertraulich. Sämtliche Dokumentation im Zusammenhang mit der Malware (Testing, Fake C&C) sind als vertraulich zu erachten und dürfen nicht öffentlich gemacht werden.

HSR Hochschule für Technik
Postfach 1475
CH-8640 Rapperswil
www.hsr.ch

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quellcodeverzeichnis

Teil I.

Technischer Bericht

1. Einleitung

Diese Bachelorarbeit ist das Resultat eines Abschlussprojekts für den Bachelor of Science in Computer Science an der Hochschule für Technik Rapperswil (HSR). Dieses Projekt wurde in Zusammenarbeit mit Ivan Bütler und Compass Security durchgeführt. Compass Security ist ein auf Security Assessments und forensische Untersuchungen spezialisiertes Unternehmen.
Diese Abschlussarbeit wurde durch Ivan Bütler angestossen, um ein mögliches Proof of Concept anhand der Aufgabenstellung zu konzipieren und zu programmieren. Dieses Kapitel bietet eine Einleitung in die Aufgabenstellung und den Inhalt dieser Arbeit.

1.1. Kontext

Mitte 2016[12] wurde über einen Hacker Angriff bei der RUAG berichtet, welcher sowohl die RUAG selbst als auch das Eidgenössische Departement für auswärtige Angelegenheiten (EDA) betraf. Dieser Angriff war jedoch speziell in der Hinsicht, da er während fast 14 Monaten nicht entdeckt wurde. Bei diesem Angriff handelte es sich um einen Advanced Persistent Threat (APT) Angriff, welcher über sehr lange Zeit lief. Dabei wurde die Malware über ein aus dem Internet heruntergeladenes Spiel in das Firmennetzwerk eingeschleust. Dieses Spiel hat dann sporadisch immer wieder einen Command & Control (C&C) Server angefragt, ob nun ein 0-Day Bug existiere für den vorhandenen Patchlevel, OS etc., sobald ein 0-Day gefunden wird, kann der C&C Server seine Befehle an den Client senden. Das erste Ziel ist dabei vielmals Administratorrechte auf dem installierten PC zu erlangen und danach unentdeckt Daten weiter zu senden.

1.2. Problembeschreibung

Die Arbeit setzt ihren Fokus auf die C&C Phase einer APT Attacke. Voraussetzung dafür ist ein infiziertes System, das bereits einem Reverse Engineering unterzogen wurde. Die Malware und deren Funktionsweise ist somit bekannt. Nun besteht Bedarf, die nächste Phase der APT Attacke zu verzögern, um weitere Schritte gegen das Unterfangen des Angreifers zu unternehmen. Diese Arbeit soll sich damit befassen, ein System zu entwickeln, das ein weiteres Kompromittieren der Infrastruktur durch den C&C Server des Angreifers verhindert oder zumindest verzögert. Das System soll kein Analyse Tool sein, es dient lediglich der Zeitgewinnung.

1.3. Projektziel

Der Kernpunkt der Arbeit besteht darin, aus dem Netzwerkverkehr einer Organisation einen Zugriff auf einen C&C Server zu erkennen und diesen auf einen Fake C&C Server umzuleiten. Die Aufgabe kann in folgende Fragen aufgeteilt werden:

1. Wie funktioniert eine APT Attacke und welche Schritte beinhaltet sie? - Kapitel 2, Abschnitt 2.1
2. Welche Möglichkeiten bestehen, um HTTPS Verkehr unterbrechen zu können? - Kapitel 2, Abschnitt 2.2.3
3. Wie kann ein einzelner Netzwerkstrom, separiert von allen anderen, auf einen beliebigen Endpunkt dynamisch umgeleitet werden? - Kapitel 2, Abschnitt 2.3
4. Wie kann der HTTP/HTTPS Verkehr mitsamt Payload sauber geloggt werden? - Kapitel 2, Abschnitt 2.4
5. Wie kann der HTTP/HTTPS Verkehr für die Analyse gespeichert werden? - Kapitel 2, Abschnitt 2.5
6. Wie kann die Malware anhand der gespeicherten Netzwerkpakete erkannt werden? - Kapitel 2, Abschnitt 2.6
7. Wie können die Angreifer und die Malware getäuscht werden? - Kapitel 2, Abschnitt 2.8
8. Wie würde eine Systemarchitektur einer Software aussehen, die die Problemstellung des Auftrags erfüllt? - Kapitel 3
9. Ist eine praxistaugliche Software, die die Problemstellung des Auftrags erfüllt, möglich? - Kapitel 4, 5, 6
10. Wie würde eine Software aussehen, die für den Einsatz in einem realen Szenario geeignet ist? - Kapitel 7
11. Was sind die Vor- und Nachteile der erarbeiteten Lösung? - Kapitel 8

1.4. Vorgehen

Analyse
Die Analyse hat sich über das ganze Projekt gezogen und wurde stetig erweitert, sie befindet sich im Kapitel 2
Architektur
Anhand der Analyse wurden mögliche Architekturen gefunden und beschrieben, sie befinden sich im Kapitel 3.
Design und Implementation Proof of Concept
Anhand der gefundenen Architekturen wurden Prototypen erstellt, welche die möglichen Ansätze aufzeigen
Proof of Concept validieren
Das Proof of Concept wurde validiert, die Ergebnisse dazu befinden sich im Kapitel 6.
Fish Tank Suite
Aus den Prototypen entstand die Fish Tank Suite, welche im Kapitel 7 beschrieben ist.
Schlussfolgerung
Die Schlussfolgerung zum Projekt und das Resultat befinden sich im Kapitel 8

2. Analyse

In diesem Kapitel werden folgende Fragen beantwortet:

- *"Wie funktioniert eine Advanced Persistent Threat (APT) Attacke und welche Schritte beinhaltet diese?"* - Abschnitt 2.1
- *"Was für Möglichkeiten bestehen, um HTTPS Verkehr unterbrechen zu können?"* - Abschnitt 2.2.3
- *"Wie kann ein einzelner Netzwerkstrom, separiert von allen anderen, auf einen beliebigen Endpunkt dynamisch umgeleitet werden?"* - Abschnitt 2.3
- *"Wie kann der HTTP/HTTPS Verkehr mitsamt Payload sauber geloggt werden?"* - Abschnitt 2.4
- *"Wie kann der HTTP/HTTPS Verkehr für die Analyse gespeichert werden?"* - Abschnitt 2.5
- *"Wie kann die Malware anhand der gespeicherten Netzwerkpakete erkannt werden?"* - Abschnitt 2.5
- *"Wie können die Angreifer und die Malware getäuscht werden?"* - Abschnitt 2.8

2.1. Command & Control

2.1.1. Begrifflichkeit

Der Begriff APT[26] bezeichnet einen komplexen, zielgerichteten und effektiven Angriff auf kritische IT-Infrastrukturen. Bei solchen Angriffen gehen die Angreifer sehr zielgerichtet vor und nehmen grossen Aufwand auf sich. Das Ziel eines APT ist es, möglichst lange unentdeckt zu bleiben, um über längeren Zeitraum sensible Informationen auszuspähen oder Schaden anzurichten. Die Erkennung und Analyse dieser Angriffe gestalten sich als sehr schwierig. Nur das Sammeln, Analysieren und Korrelieren von Sicherheitsinformationen kann Hinweise zur Erkennung geben.

2.1.2. Die 6 Schritte einer APT Attacke

Die folgenden Punkte[7] beschreiben eine von vielen möglichen APT Attacken. Es sind natürlich diverse andere Abläufe mit verschiedenen Zielen möglich.

1. Der Angreifer schleust über einen beliebigen Attack Vector eine Malware in das Netzwerk einer Organisation ein. Das Netzwerk ist "compromised".

2. Die Malware sucht nach weiteren Vulnerabilities oder kommuniziert mit command-and-control Servern, um Befehle oder Schadcode zu erhalten.

3. Die Malware versucht Backdoors zu installieren, um den Angriff fortzusetzen, falls eine der Backdoors geschlossen wird.

4. Wenn der Angreifer einen zuverlässigen Zugriff auf das Netzwerk der Organisation aufgebaut hat, können Benutzernamen und Passwörter gesammelt werden. Nun hat der Angreifer Zugriff auf die Systeme der Organisation.

5. Die Malware sammelt Daten auf einem Staging Server und schleust die Daten dann aus dem Netzwerk der Organisation raus. Das Netzwerk ist "breached".

6. Am Ende werden alle Beweise des Angriffs entfernt. Das Netzwerk bleibt jedoch "compromised", um den Angriff zu einem späteren Zeitpunkt fortzusetzen.

2.2. Proxy

2.2.1. HTTP Proxy

Ein HTTP Proxy[29] erlaubt, den ein- und ausgehenden HTTP/HTTPS Verkehr über einen zentralen Punkt zu führen, das kann z.B. zum Filtern oder Sperren von nicht erlaubten Webseiten in einem Firmennetz interessant sein. Der HTTP Proxy wird dabei nur über HTTP angesprochen, falls ein Client eine HTTPS Verbindung benötigt, wird dies über einen "Proxy Connect" [18] gemacht, der eine Secure Socket Layer (SSL) Verbindung zur Zielseite aufbaut. Die HTTPS Verbindung vom Client zum Server kann dabei nicht im Detail betrachtet werden, man sieht nur den Header, aber nicht den Body. Bei einer reinen HTTP Verbindung ist ein "Proxy Connect" nicht nötig.

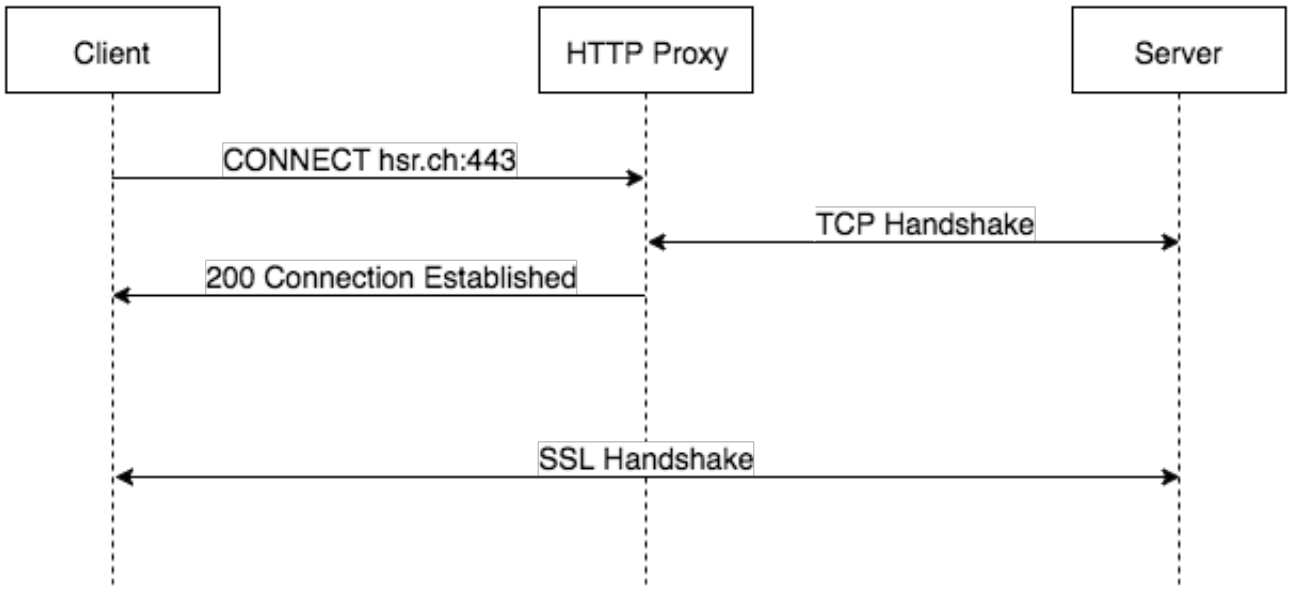


Abbildung 2.1.: Analyse: Proxy Connect

2.2.2. Proxy Framework

Ein Proxy Framework bietet die Möglichkeit, einen eigenen Proxy und damit eine eigene Logik programmieren zu können, wie auf einen Request oder Response reagiert werden soll. Die Proxy Frameworks sind oft zum Debuggen von Web Traffic gedacht, sie können aber auch für etwas ganz anderes verwendet werden, da die Logik selbst programmiert und angepasst werden kann. Bekanntere Proxy Frameworks sind hierbei der MITMProxy[1] oder der FiddlerCore[2], beides sind Frameworks und können direkt in einem Software Projekt verwendet werden.
Die Evaluation der beiden Proxy Frameworks wurde im Abschnitt 2.2.4 behandelt.

[1] https://mitmproxy.org/
[2] http://www.telerik.com/fiddler

2.2.3. SSL Splitting

SSL Splitting[20] ermöglicht die Einsicht in den Payload verschlüsselter Pakete.

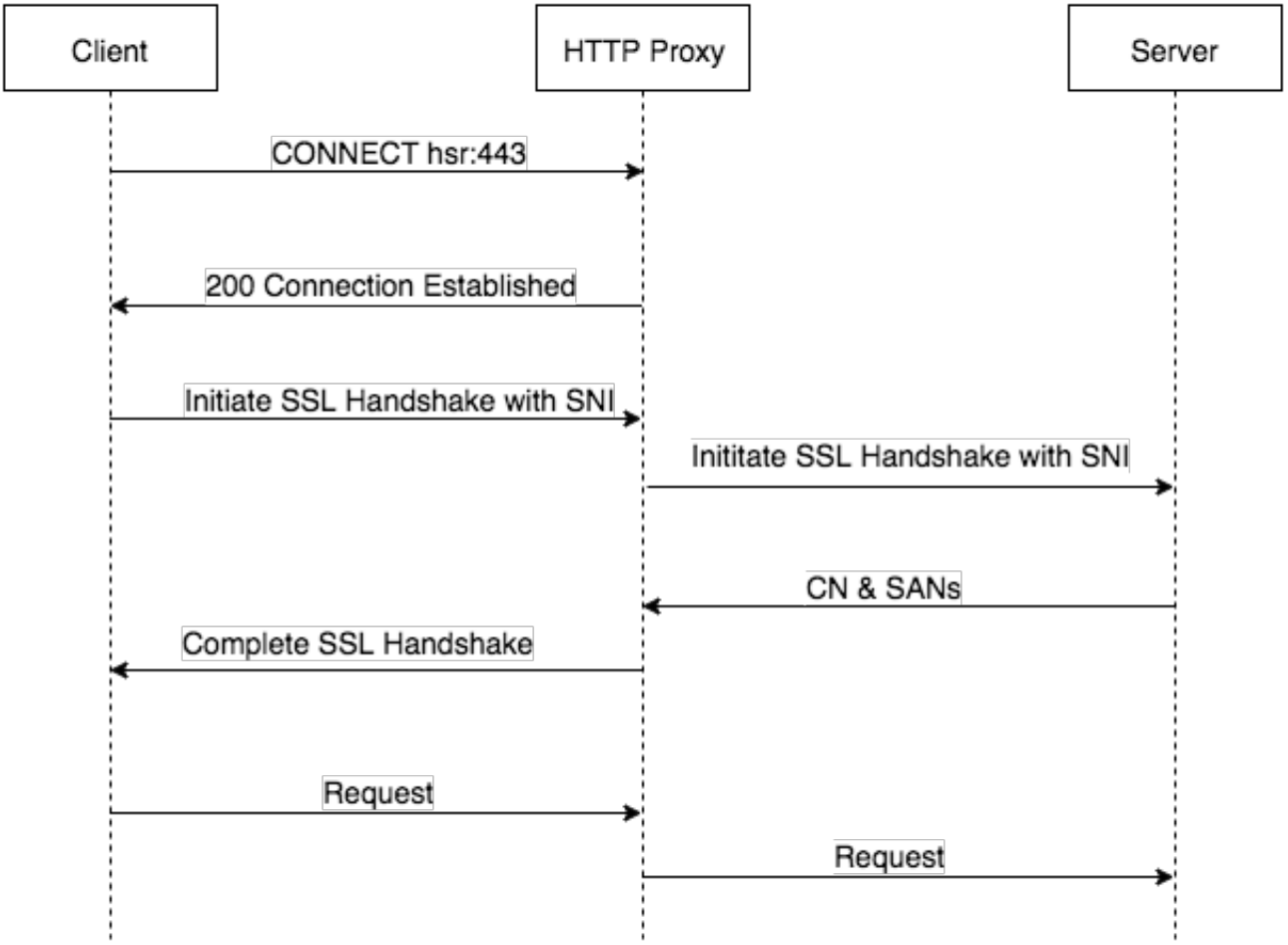


Abbildung 2.2.: Analyse: SSL Splitting

Das Einsehen von verschlüsseltem Traffic ist jedoch fragwürdig, da dadurch auch die Möglichkeit besteht, privaten Content zu protokollieren z.B. Passwörter.

2.2.4. Evaluation Proxy Framework

MITMProxy

Feature	Beschreibung	Bewertung
Pattern Matching	Zugriff auf Payload und Header.	OK
HTTP/S Verkehr umleiten	Durch Umschreiben der Destination auf Layer 7 möglich.	OK
SSL Splitting	Wird unterstützt.	OK
Dokumentation	Gut geschriebene Doku und einige Beispiele, welche allerdings nicht alle funktionieren.	Genügend
Entwicklungsstand	Regelmässig neue Releases.	OK
Performance	Wird als interaktiver Proxy zu Debugzwecken und nicht als Proxy Library beworben.	Ungenügend

Tabelle 2.2.: Analyse: Prototyp V1 Evaluation MITMProxy

FiddlerCore

Feature	Beschreibung	Bewertung
Pattern Matching	Zugriff auf Payload und Header	OK
HTTP/S Verkehr umleiten	Durch Umschreiben der Destination auf Layer 7 möglich	OK
SSL Splitting	Wird unterstützt.	OK
Dokumentation	Doku nur zu Klassen verfügbar, Beispiele sind meistens nur für die Skriptengine verfügbar, funktionieren jedoch auch mit FiddlerCore	Genügend
Entwicklungsstand	Regelmässig neue Releases.	OK
Performance	Wird als Proxy Library für eigene Softwareprojekte beworben.	OK

Tabelle 2.4.: Analyse: Prototyp V1 Evaluation FiddlerCore

Schlussfolgerung

Anhand der Evaluation in 2.2.4 haben wir uns für die Proxy Library FiddlerCore entschieden, um den Prototyp V1 zu programmieren.

2.2.5. Evaluation HTTP Proxy

Beim HTTP Proxy wurden zwar bei der Analyse verschiedene Proxies (Privoxy [3], Polipo[4]) angeschaut, jedoch bietet nur der Squid Proxy [5] von Haus aus einen Internet Content Adaptation Protocol (ICAP) Client an (siehe 2.4.5).

Schlussfolgerung

Wegen den in 2.2 genannten Gründen hatten wir uns für den Squid Proxy entschieden, um den Prototyp V2 umzusetzen.

2.3. Redirect

2.3.1. Iptables

Iptables[24][28] ist eine Linux Applikation, die es ermöglicht, diverse Rules für die Linux Kernel Firewall zu definieren. Mit diesen Rules ist es möglich, anhand der Destination Adressen ein Paket auf eine andere Adresse umzuleiten. Dies geschieht auf Layer 3 und ist somit protokollunabhängig, was die darüber liegenden Layer bzw. Protokolle betrifft. Die Umleitung auf Layer 3 vereinfacht das System, da nicht auf Layer 7 diverse Ausnahmen, wie z.B. HTTP oder HTTPS, beachtet werden müssen.

Es gibt zwei mögliche Architekturen. Bei der ersten Variante wird die Iptable auf dem selben Host, auf dem auch der Proxy betrieben wird, gesetzt. Bei der zweiten Variante, befindet sich die Iptable auf einem anderen Host, der sich jedoch hinter dem Proxy befinden muss.

[3] https://www.privoxy.org/
[4] https://www.irif.fr/~jch/software/polipo/
[5] http://www.squid-cache.org/

Variante 1: Proxy und Iptables auf dem selben Host

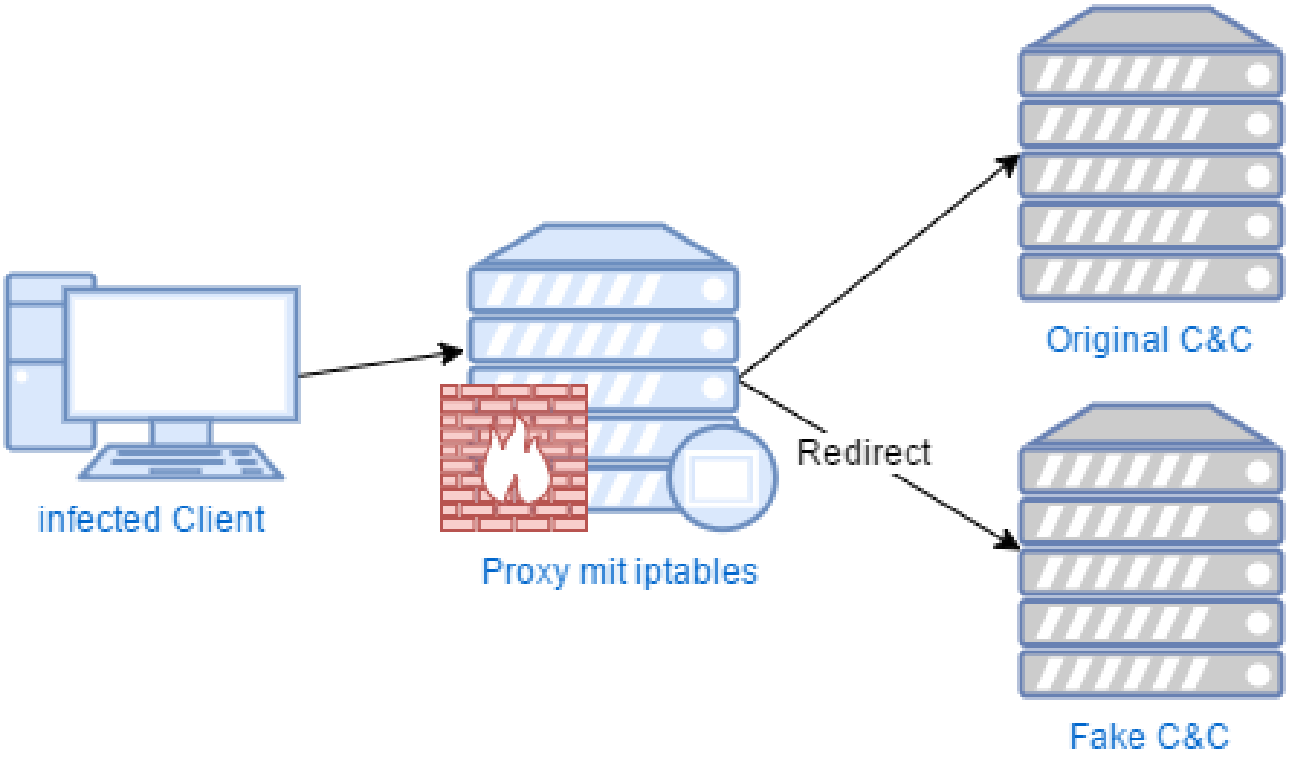


Abbildung 2.3.: Analyse: Systemübersicht Variante 1

Zu beachten ist, dass die Pakete bei der INPUT Chain ankommen, da die Destination Adresse der Proxy ist. Das heisst, die Pakete werden wieder über die OUTPUT Chain ausgegeben. Die Iptable kann also wie unten beschrieben gesetzt werden um eine gewisse Destination Adresse auf eine andere umzuleiten. Bei dieser Architektur ist jedoch kein Proxy Chaining möglich, da bei einem Chaining die Destination immer die des nächsten Proxys ist.

```
1 Iptables -t nat -A OUTPUT --destination [Original IP] -j DNAT --to-destination [Redirect
  ↪  IP]
```

Auflistung 1: Analyse: Iptable Variante 1

Variante 2: Proxy und Iptables auf separatem Host

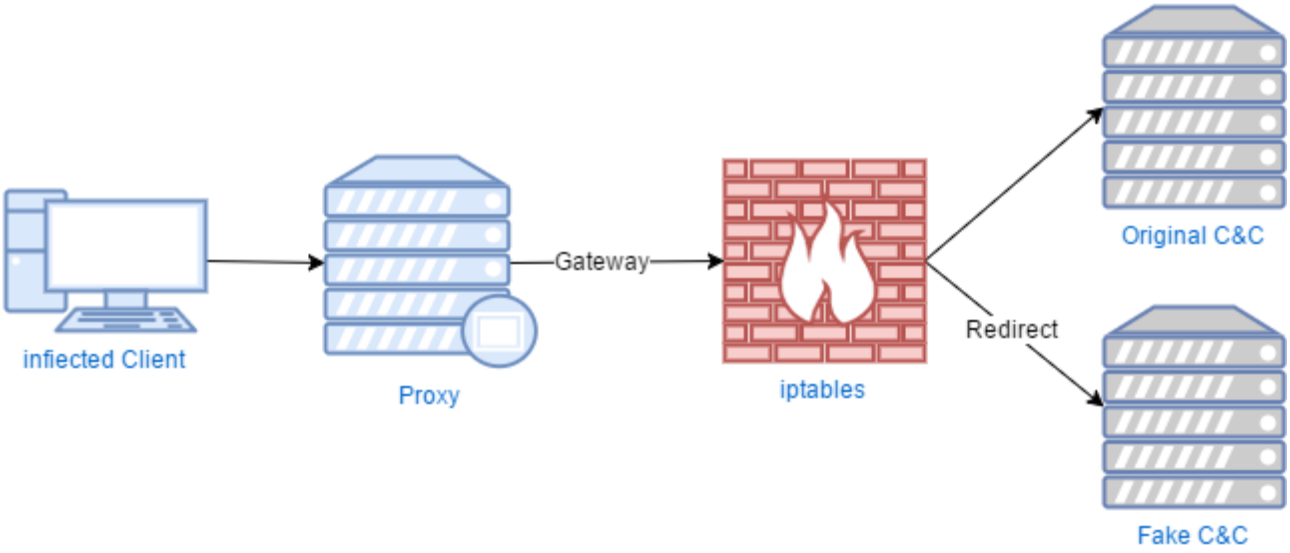


Abbildung 2.4.: Analyse: Systemübersicht Variante 2

Diese Architektur ist nur möglich, wenn der Host mit den Iptables als Gateway konfiguriert ist, da nur so die Pakete in der PREROUTING Chain ankommen. Das Masquerading wird benötigt, um die ursprüngliche Destination Adresse beim Response in die Source Adresse einzutragen, ansonsten lehnt der Client, der den Request gesendet hat, die Verbindung ab. Das Proxy Chaining kann umgangen werden, wenn die Iptables hinter oder auf dem letzten Proxy installiert werden.

```
1 Iptables -t nat -A PREROUTING --destination [Original IP] -j DNAT --to-destination
   ↪ [Redirect IP]
2
3 Iptables -t nat -A POSTROUTING -j MASQUERADE
```

Auflistung 2: Analyse: Iptable Variante 2

Schlussfolgerung

Für die Realisierung des Prototyps V2 und der Fish Tank Suite fällt der Entscheid auf Variante 1, da sowohl Konfiguration als auch das Setup simpler ist. Mit Variante 2 ist es jedoch möglich, das Proxy Chaining zu umgehen, das in grösseren Firmennetzwerken unter Umständen notwendig ist.

2.3.2. Layer 7 Redirect

Bei einem Redirect auf Layer 7 wird der Header des HTTP/HTTPS Requests umgeschrieben, dabei wird ein Proxy Framework verwendet, welches diese Funktion im Normalfall anbietet. Es ist aber auch durch ein Rewrite-Programm[6] für einen HTTP Proxy möglich.

[6] http://www.squid-cache.org/Doc/config/url_rewrite_program/

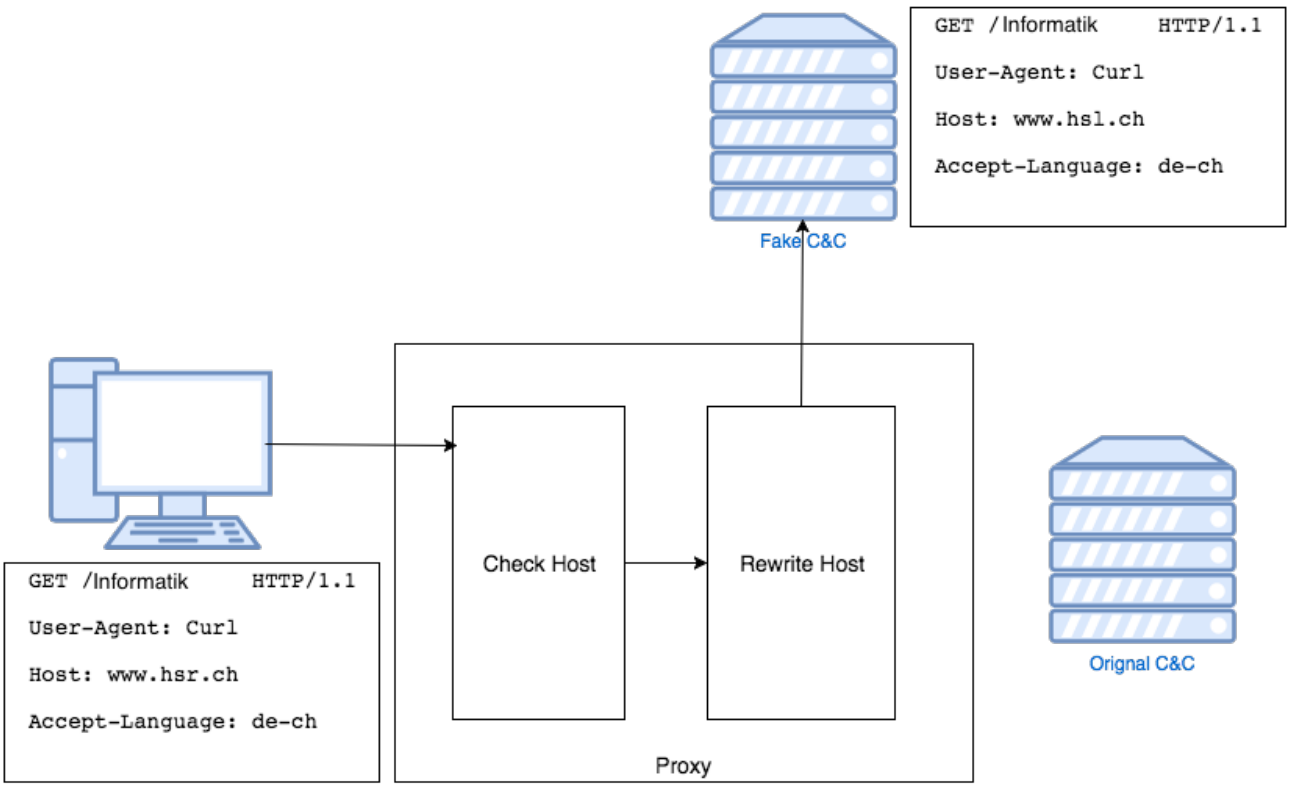


Abbildung 2.5.: Analyse: Layer 7 Redirect

Schlussfolgerung

Für die Realisierung des Prototyps V1 ist eine sofortige Umleitung vorgesehen, die nur mit dieser Methode möglich ist. Deshalb wird beim Prototyp V1 auf einen Layer 7 Redirect gesetzt.

2.4. Logging

2.4.1. HTTP Proxy Logging

HTTP Proxies[22] besitzen zwar Logging Möglichkeiten, jedoch hört dies bei Access Logs schon auf. Ganze Requests zu loggen gehört nicht zu den Logging-Möglichkeiten eines HTTP Proxies.

2.4.2. Packetbeat

Packetbeat[3] [7] ist ein Produkt von Elastic[8] und bietet das einfache Loggen von Netzwerk Paketen an. Jedoch wird HTTPS als Protokoll nicht unterstützt und müsste durch eine eigene Implementation erweitert werden.

[7] https://www.elastic.co/products/beats/packetbeat
[8] https://www.elastic.co

2.4.3. TCPDump

TCPDump[15] [9] ist ein Programm um auf einem Port oder Interface zu horchen und alles was ankommt kann als PCAP Datei gespeichert werden, womit der ganze TCP Stream nachvollzogen werden kann. Allerdings ist die Lösung nur File-basiert und verschlüsselter Datenaustausch kann nicht inspiziert werden.

2.4.4. SSLDump

SSLDump[19] [10] ist eine Erweiterung zu TCPDump und erlaubt das Entschlüsseln von verschlüsselten Paketen.

2.4.5. ICAP

ICAP[27] wird verwendet, um HTTP Requests und Responses an einen Server weiterzuleiten, wo gewisse Änderungen, Malware Scans oder generelle Anpassungen gemacht werden können. Dabei werden die bestehenden HTTP Requests und Responses in ICAP Requests verpackt und als Body mitgesendet, falls nun SSL Splitting aktiviert ist, kann auch der Payload von verschlüsseltem Traffic auch auf einem ICAP Server angepasst werden.

REQMOD

Unter REQMOD versteht man die Request Modification, welche der ICAP Client an den ICAP Server sendet, dabei wird der ursprüngliche Request als Body in einen ICAP Request eingebettet. Der Server kann dann Modifikationen vornehmen und sendet die Requests zurück an den ICAP Client.

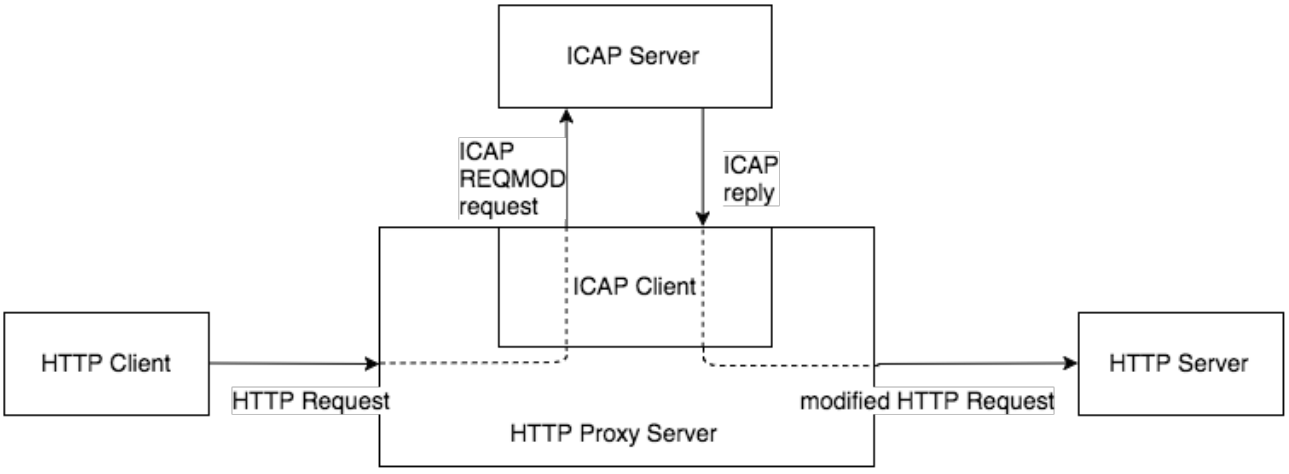


Abbildung 2.6.: Analyse: REQMOD Ablauf

[9] http://www.tcpdump.org/tcpdump_man.html
[10] http://ssldump.sourceforge.net/

RESPMOD

Unter RESPMOD versteht man die Response Modification, welche gleich wie REQMOD funktioniert, nur dass hierbei Responses modifiziert werden.

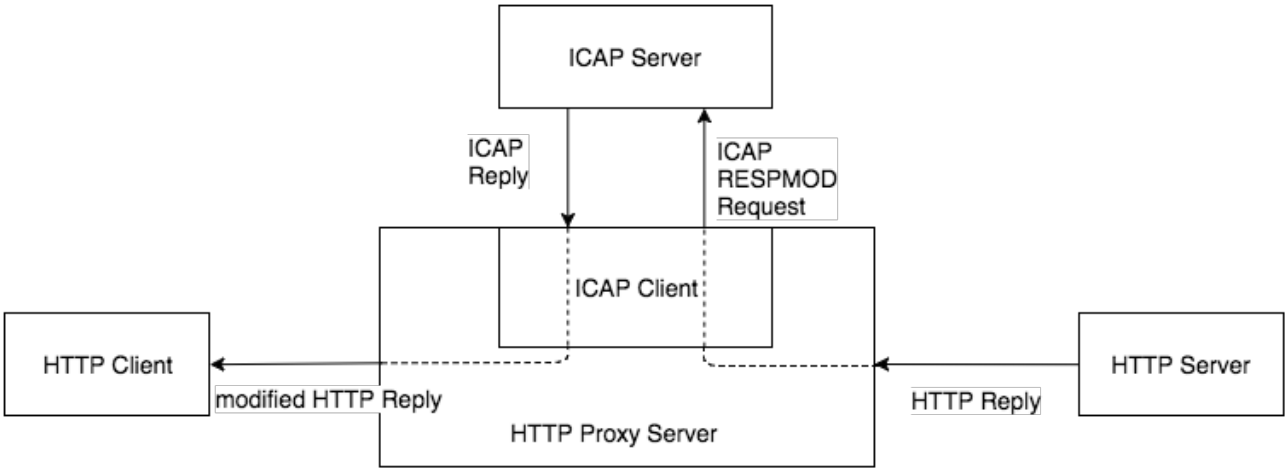


Abbildung 2.7.: Analyse: RESPMOD Ablauf

ICAP Client

Der ICAP Client kann ganze Request und Responses an den ICAP Server senden oder aber er sendet nur eine Preview (bei Virus Checks kann dies reichen). Der Client sendet dazu eine OPTIONS Anfrage an den Server und bekommt zurück was er alles an den Server senden kann. Die RESPMOD OPTIONS und REQMOD OPTIONS müssen einzeln abgefragt werden und können auch auf verschiedenen Servern aufgeschaltet werden.

```
1 OPTIONS icap://icap.example.com/log_requests ICAP/1.0
2 Host: icap.example.com
```

Auflistung 3: Analyse: ICAP Options Request

Anhand der ICAP Responses auf die OPTIONS Anfrage, entscheidet der Client, ob er den ganzen Response bzw. Request sendet oder nur eine Preview. Es kann auch eingeschränkt werden welche Methoden vom Server unterstützt sind.

```
ICAP/1.0 200 OK
Date: Mon, 10 Nov 2016  09:55:21 GMT
Methods: RESPMOD
Service: ICAP Server
Encapsulated: null-body=0
Max-Connections: 100
Options-TTL: 3600
Transfer-Complete:
Transfer-Ignore: jpg
```

Auflistung 4: Analyse: ICAP Options Response

ICAP Server

Da der ICAP Server keine Anpassungen am Body der Requests und Responses machen muss, können diese wieder ganz zurückgesendet werden, oder man sendet den Response 204 (No Modification Needed) an den ICAP Client zurück. Dabei kann auf dem ICAP Server der ganze Request oder Response angeschaut und bei Bedarf auch irgendwohin geloggt werden zur weiteren Aufbereitung.

2.4.6. Performance

Weil beim Logging die Performance wichtig ist, da relativ viele Daten anfallen können, muss das Logging wenn möglich auf einen anderen Service ausgelagert werden, damit der Proxy seine Arbeit verrichten kann.

2.4.7. Schlussfolgerung

Da die Performance einer der wichtigsten Aspekte ist haben wir uns beim Logging für ICAP entschieden, was eine Proxy unabhängige Lösung ist mit der Möglichkeit den Logging-Service woanders zu hosten.

2.5. Datenbank

2.5.1. Elasticsearch

Bei Elasticsearch[9][11] handelt es sich um ein Full Text Search (FTS) System, wodurch Suchen nach bestimmten Suchbegriffen oder Patterns in längeren Texten mit kleinem Zeitaufwand möglich ist.

Die Analyse der Datenhaltung ist nicht Teil der Arbeit. Es ist nur zu beachten, dass die gewählte Datenbank skalierbar ist, was auf Elasticsearch zutrifft.

[11] https://www.elastic.co/products/elasticsearch

2.5.2. Kibana

Kibana[5][12] ist ein Grafisches Dashboard für Elasticsearch, worauf man alle Einträge von Elasticsearch durchsuchen und visualisieren kann.

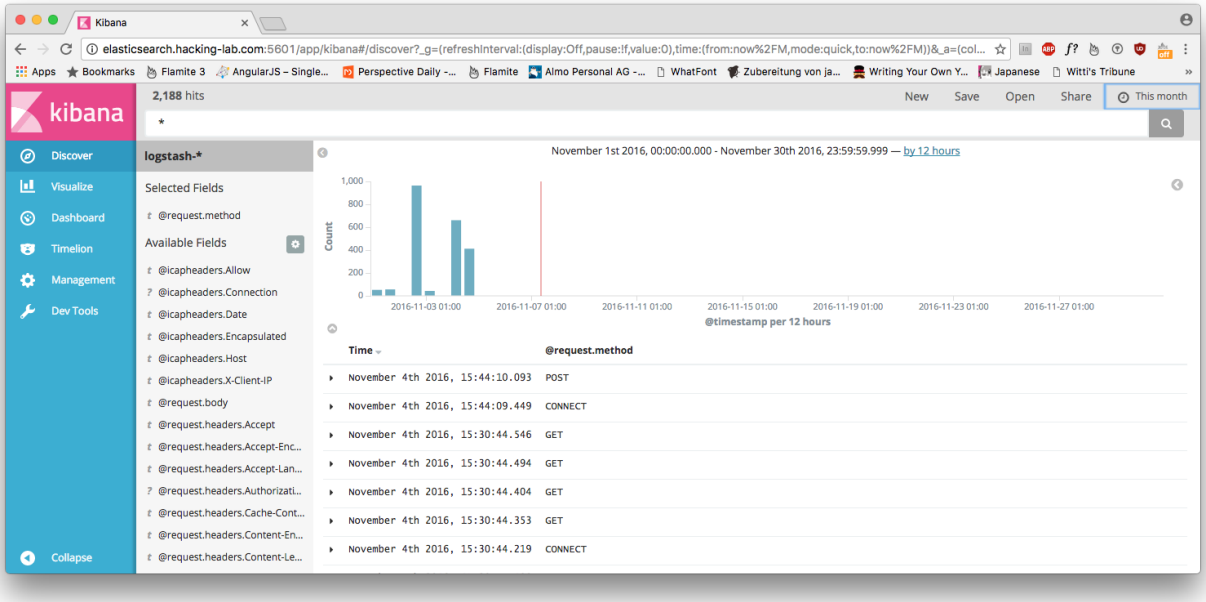


Abbildung 2.8.: Analyse: Kibana Dashboard

2.6. Malware Detection

Die Erkennung der Malware kann auf verschiedenste Weise gelöst werden. Eine einfache Variante ist die Suche nach Schlüsselwörtern im Header und Payload. Es ist auch denkbar das Zeitverhalten beziehungsweise den Intervall von Paketen zu untersuchen und so Schlüsse auf Malware-Aktivitäten zu ziehen. Sicher ist, dass jede Malware seine ganz eigenen Eigenschaften besitzt, und nach diesen muss gezielt gesucht werden, das heisst eine generische Suche wäre zu ungenau.

2.6.1. Pattern Matching im Payload

Die Suche nach Schlüsselwörtern oder Patterns ist wie oben schon genannt die denkbar einfachste und auch präziseste Lösung. Es gibt jedoch ein grosses Problem bei diesem Verfahren. Der Payload könnte verschlüsselt sein. Falls es sich um eine symmetrische Verschlüsselung handelt kann der Key ermittelt werden, bei asymmetrischen Verschlüsselungen ist dies nicht so leicht.

Schlussfolgerung

Diese Methode ist also nur möglich wenn der Payload im Klartext gelesen werden kann. Deswegen sind auch andere Erkennungsverfahren, bei denen der Payload keine Rolle spielt interessant.

[12] https://www.elastic.co/products/kibana

2.6.2. Zeitliches Verhalten

Eine weitere Möglichkeit stellt das analysieren des zeitlichen Verhaltens gewisser Pakete dar. Dabei soll untersucht werden, ob eine Malware in gewissen abständen Pakete sendet.

In der Praxis stellt sich dieses Verfahren als sehr unpräzise dar, wenn man annimmt, dass nur das Zeitverhalten betrachtet werden darf und praktisch keine Informationen des Pakets zur Verfügung stehen. Das Problem dabei ist, dass bei einer Datenbankabfrage die Pakete im zeitlichen Abstand von 10 Sekunden liefern soll beinahe immer mehr als ein Paket pro Intervall zurückgeliefert wird. Das eigentliche Problem dabei ist nun die Assoziation der zusammengehörigen Pakete in den Intervalls, was schlichtweg unmöglich ist ohne weitere Parameter in den Algorithmus zu nehmen.

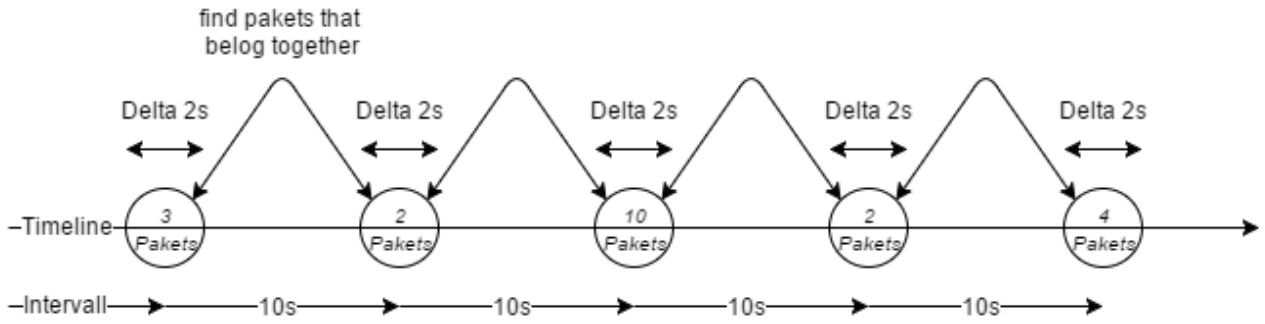


Abbildung 2.9.: Analyse: Intervall Search

Schlussfolgerung

Die Analyse ergab, dass es zu viele False-Positives gibt, somit ist dieses Verfahren eher ungeeignet.

2.6.3. URL Sequenz

Dieses Verfahren baut auf der Analyse des zeitliche Verhaltens auf und versucht dessen Probleme zu verringern. Es geht hier darum durch eine Liste von URLs und die dazwischenliegenden ungefähren Zeitabstände das Verhalten einer Malware zu erkennen. Beim nachfolgenden Beispiel wird zuerst nach einem "register", und dann nach einem "hello"das 10 Sekunden später eingetroffen sein muss, gesucht.

1. Suche nach Register-URI.
2. Zeitpunkt der Register-URI merken.
3. Suche nach der Hello-URI mit zeitlichem Offset nach dem Zeitpunkt der Register-URI.
4. Falls die URIs im korrekten Abstand zueinander vorliegen handelt es sich um die Malware.

Schlussfolgerung

Dieses Vorgehen ist um einiges genauer als nur der zeitliche Intervall. Es sind natürlich komplexere Abläufe denkbar, für die zur Verfügung gestellte Malware reicht das aber schon.

2.7. Entschlüsselung

Die Entschlüsselung ist in 3 Schritte aufgeteilt. Diese Schritte sind notwendig, um den verschlüsselten Payload in ein lesbares Format zu bringen. Die Schritte können bei anderen Verschlüsselungen anders ausfallen und können deshalb nicht generisch implementiert werden. Das heisst für jede Malware muss auch ein separater Algorithmus entwickelt werden.

Da die Malware für die die Entschlüsselungen entwickelt wurde vertraulich ist, können keine genauen Details über deren Funktionsweise dokumentiert werden.

2.7.1. XOR und AES128

1. **EXTRACT:** Der Payload wird von unnötigen Zeichen befreit.
2. **DECODE:** Der Text wird dekodiert (Base64).
3. **DECRYPT:** Hier findet die eigentliche Entschlüsselung statt.

2.7.2. Logstash

Bei Logstash[2][13] handelt es sich um eine Software, die das Zusammenfügen von Logs von verschiedenen Quellen sehr vereinfacht. Dabei durchlaufen alle Logs im Normalfall die gleichen 3 Schritte.

1. **INPUT:** Als Input können verschiedene Protokolle verwendet werden, an welche die Log Quellen ihre Logs senden.
2. **FILTER:** Ein Filter wird zum Anreichern oder Anpassen der gesendeten Log Daten verwendet.
3. **OUTPUT:** Zum Abschluss werden die Daten an einen Output gesendet, hier können ebenfalls verschiedene Protokolle verwendet werden oder eine Datenbank, wie zum Beispiel Elasticsearch.

2.7.3. Performance

Da manchmal neue Entschlüsselungsalgorithmen entwickelt werden, muss Logstash nach der Aktualisierung neu gestartet werden. Damit kein Unterbruch stattfindet, bis die Änderungen aufgeschaltet sind, bietet sich eine Queue an, die auch persistent bleibt, selbst wenn Logstash mal neu gestartet wird.

2.7.4. Schlussfolgerung

Da die Entschlüsselung von verschlüsseltem Payload zeitreibend ist, bietet es sich sehr an, diese auf einem eigenen Service vorzunehmen, um so allen anderen Systemen nicht in die Quere zu kommen. Die Malware kann sich immer wieder ändern und dabei muss der Filter von Logstash upgedatet werden, daher wurde entschieden, eine RabbitMQ[4] Queue zwischen Log Quelle und Logstash zu setzen, damit keiner der Requests oder Responses verloren geht.

[13] https://www.elastic.co/products/logstash

2.8. Fake Command & Control

Der Fake Command & Control (C&C) befindet sich sobald eine Malware entdeckt wird immer wieder im Kontakt mit dem Original C&C, da dadurch sichergestellt werden kann, dass die Betreiber des Originalen C&C weniger schnell dahinter kommen, ob die Malware entdeckt wurde. Die Malware sendet immer wieder Updates. Wenn nun der Redirect gesetzt ist, nimmt der Fake C&C den Request entgegen und leitet diesen an den Original C&C weiter. Falls nun der Original C&C eine Antwort sendet entscheidet der Fake C&C, ob die Antwort weitergeleitet werden darf oder nicht. Kurz gesagt der Fake C&C wird zur "Fake Malware" und behält den Original C&C bei Laune mit den weitergeleiteten Requests, zusätzlich wird der Malware vorgegaukelt, dass alles ok ist.

Abbildung 2.10.: Analyse: Fake C&C Ablauf Request von Maleware

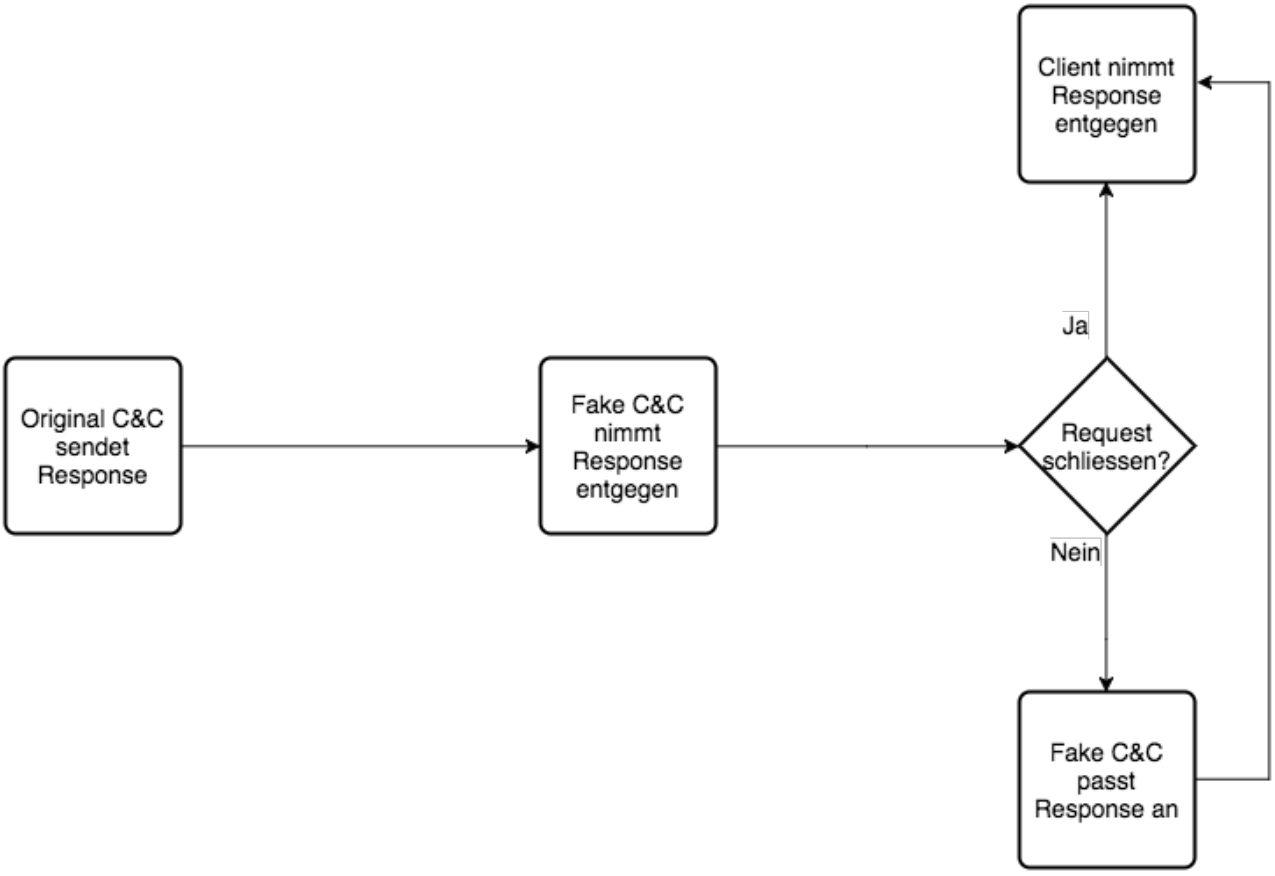


Abbildung 2.11.: Analyse: Fake C&C Ablauf Response von Original C&C

2.8.1. Schlussfolgerung

Mit diesem Verfahren bleibt der Original C&C immer aktuell und kein Verdacht tritt auf, dass die Requests der Malware über einen Fake C&C geleitet wurden. Der Fake C&C kann so ebenfalls die Responses des Originalen C&C anpassen und bearbeiten.

3. Systemarchitekturen

In diesem Kapitel wird folgende Frage beantwortet: *"Wie würde eine Systemarchitektur einer Software aussehen, die die Problemstellung des Auftrags erfüllt?"*.

Die Architekturen zeigen verschiedene Lösungsansätze, die mit Hilfe der Analysen im Kapitel 2 entwickelt wurden.

Es gibt zwei grundsätzlich verschiedene Lösungsansätze. Hierbei stellt sich die Frage, ob ein sofortiges Eingreifen bei Erkennen einer Malware notwendig ist. Eine Erkennung in Echtzeit bedeutet, dass jedes Netzwerkpaket erst nach einer Überprüfung passieren kann. Bei einer zeitversetzten Erkennung kann es vorkommen, dass ein Paket der Malware aus dem System gelangt, da die Überprüfung erst später erfolgt.

3.1. Echtzeit Erkennung

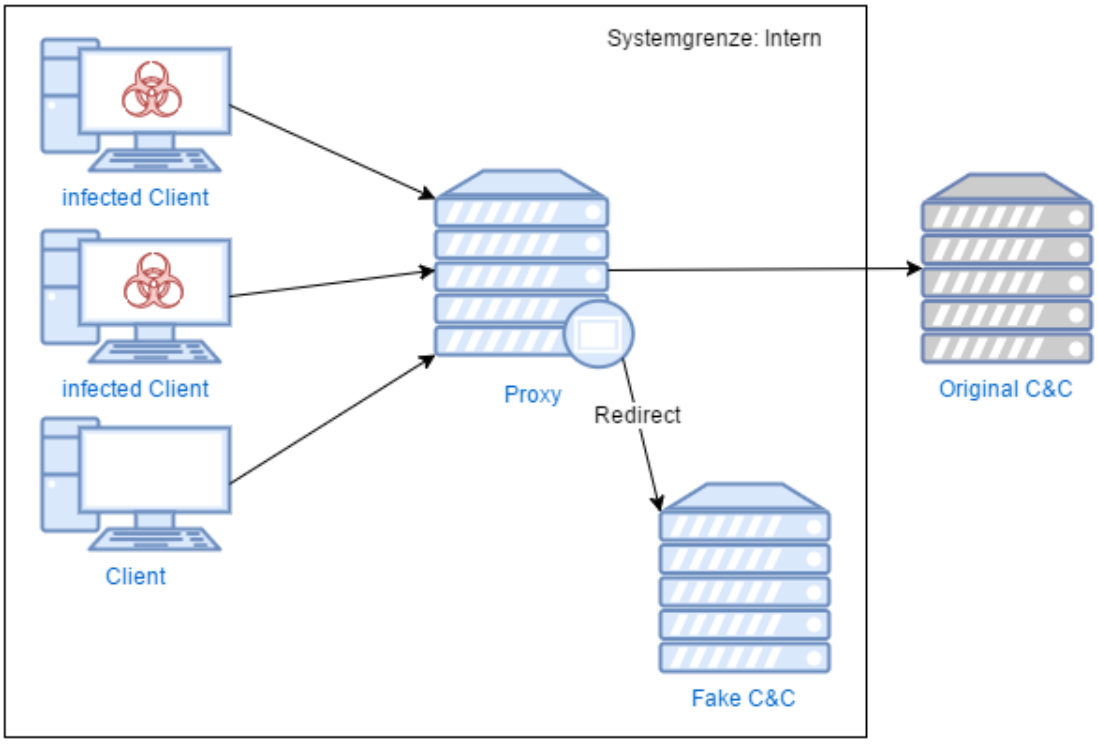


Abbildung 3.1.: Systemarchitekturen: Echtzeit Erkennung: Systemarchitektur

Bei der Echtzeit Erkennung befindet sich die gesamte Logik auf dem Proxy. Das Loggen der Pakete fällt weg, weil die Pakete direkt nach Malware Eigenschaften überprüft werden. Falls ein Paket Malware Eigenschaften aufweist, wird es direkt umgeleitet. Diese Methode kann enorme Anforderungen an die Performance stellen, da jedes einzelne Paket überprüft werden muss, bevor es den Proxy passieren darf. Vorteil dieser Architektur ist das sofortige Umleiten, das heisst, dass kein Paket, das Malware Eigenschaften aufweist, jemals den Command & Control (C&C) Server des Angreifers erreichen wird.

3.1.1. Malware Detection

Die Erkennung der Malware Aktivitäten wird durch ein Pattern Matching auf dem Proxy gelöst. Das setzt voraus, dass der Proxy eine Open Source Software ist, oder auf andere Weise das Erweitern ermöglicht.

3.1.2. Redirect

Die Umleitung muss auf Layer 7 geschehen, da die darunter liegenden Layer (z.B. Layer 3) erst beim Versenden durch den Proxy erstellt werden. Eine Umleitung mittels Iptables ist somit nicht möglich. Das erschwert die Implementation dieser Lösung, da diverse Layer 7 Protokolleigenschaften beachtet werden müssen.

3.2. Zeitversetzte Erkennung

Bei der Zeitversetzten Erkennung werden alle Pakete in einer Datenbank gespeichert und dann analysiert, damit man eine Datenbasis zur Analyse besitzt. Die Kommunikation zwischen Malware und C&C wird erst bei Erkennen des Patterns umgeleitet.

3.2.1. Variante A: Single Proxy

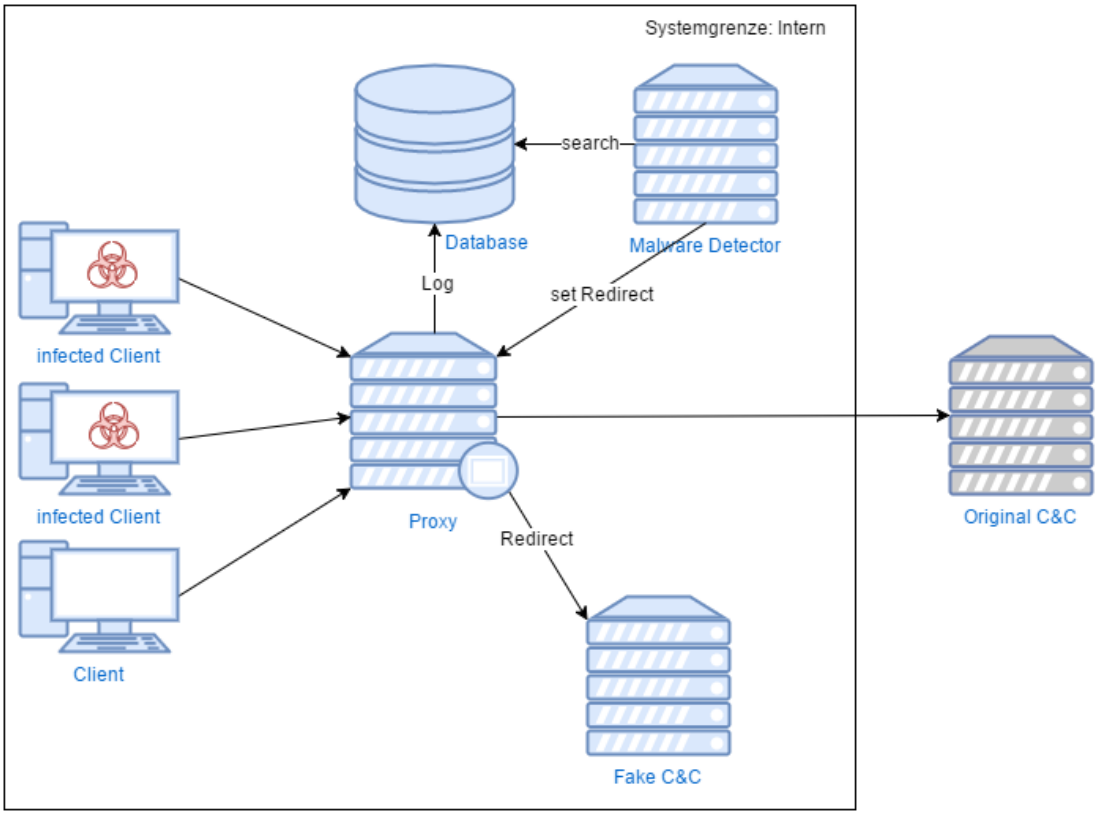


Abbildung 3.2.: Systemarchitekturen: Zeitversetzte Erkennung Variante A: Systemarchitektur

Um die Performance Probleme der Echtzeit Erkennung zu beheben, werden bei der Zeitversetzten Erkennung die Pakete in eine Datenbank geloggt. Das führt dazu, dass die Pakete direkt weitergeleitet werden können. Daraus entsteht jedoch der Nachteil, dass die Pakete einer noch nicht erkannten Malware zum C&C Server gelangen.

Logging

Die geläufigsten Proxies unterstützen Internet Content Adaptation Protocol (ICAP), dass das Loggen des Netzwerkverkehrs auf eine Datenbank ermöglicht. Die Unterstützung von ICAP ist bei dieser Variante eine zwingende Anforderung.

Malware Detection

Ein Malware Detector iteriert durch die Datenbank, bei Erkennen einer Malware wird ein Redirect angefordert. Die Suche nach der Malware kann auf diverse Arten gelöst werden. Eine Möglichkeit ist das Loggen auf eine Elasticsearch Datenbank. Mit Hilfe von Full Text Search können dann Pakete mit Malware Eigenschaften gefunden werden.

Redirect

Die Umleitung lässt sich hier am besten durch Iptables lösen. Die Iptables können direkt auf dem Host der Proxys definiert werden.

3.2.2. Variante B: Proxy Chaining

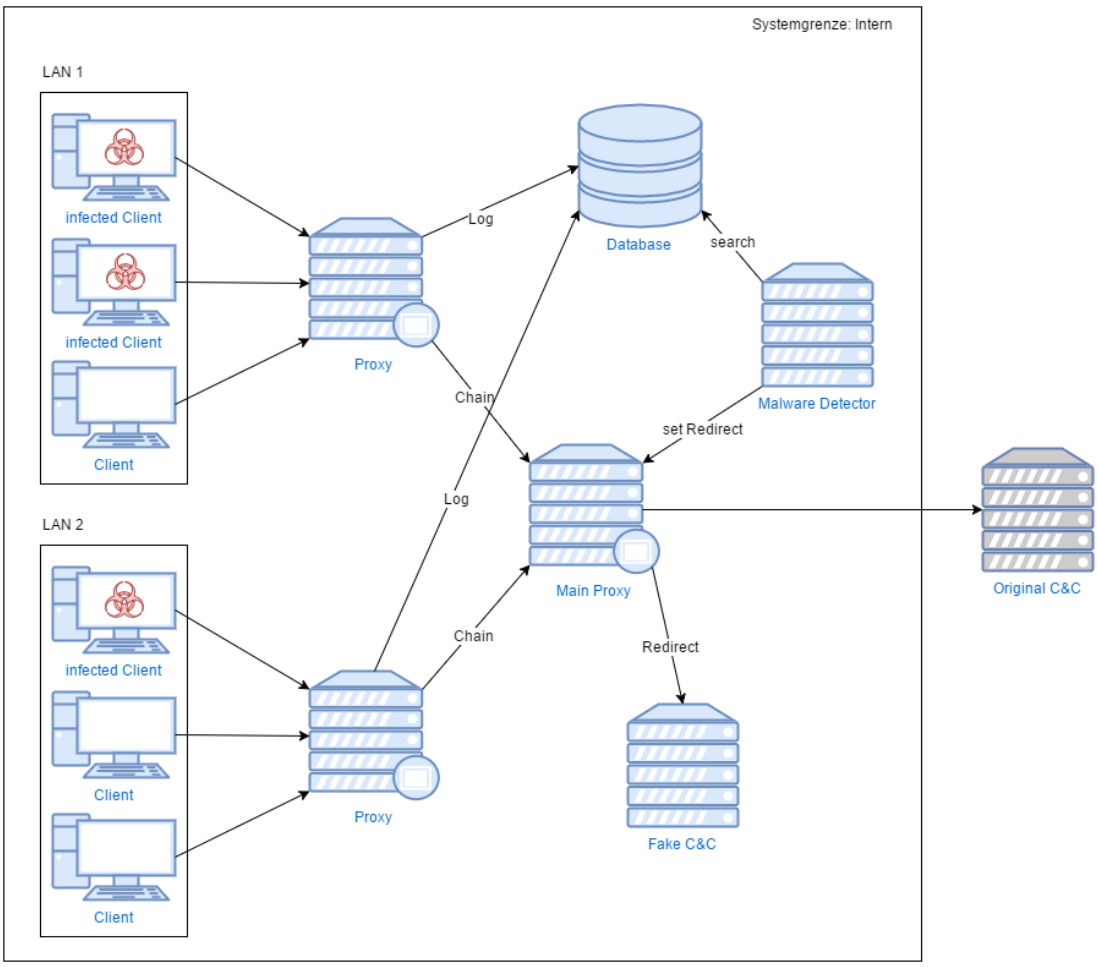


Abbildung 3.3.: Systemarchitekturen: Zeitversetzte Erkennung Variante B: Systemarchitektur

Grössere Firmen besitzen meisstens einen Proxy. Falls der bestehende Proxy keine ICAP Unterstützung aufweist oder ein Logging aus anderen Gründen nicht möglich oder gewünscht ist, bietet diese Architektur eine Lösung.

Logging

Da ein Proxy für das Logging der Pakete benötigt wird, kommt ein Proxy Chaining zum Einsatz. Es können auch mehrere Proxys für verschiedene LANs oder Abteilungen, wie im Diagramm ersichtlich, eingesetzt werden. Diese Proxys loggen die Pakete in eine zentrale Datenbank.

Malware Detection

Die Erkennung der Malware ist gleich wie bei Variante A.

Redirect

Bei dieser Architektur sind verschiedene Arten des Redirects möglich. Die einfachste Lösung ist eine Iptable hinter oder auf dem letzten Proxy. Im Diagramm wäre das der Main Proxy. Es wäre auch denkbar, den Redirect direkt auf einem der Proxys zu machen, das weist in der Praxis aber Probleme auf.

Secure Socket Layer (SSL) Bump Problem

Wenn der Squid Proxy SSL Bump aktiviert hat ist dieser nicht fähig nochmals einen CONNECT Request zu einem Upstream Proxy zu machen. Daher muss der Upstream Proxy bei aktiviertem SSL Bump HTTPS unterstützen.

4. Proof of Concept Prototype V1

In diesem Kapitel wird folgende Frage beantwortet: *"Ist eine praxistaugliche Software, die die Problemstellung des Auftrags erfüllt, möglich?"*.

Dieser Prototyp V1 setzt auf die Architektur der Echtzeit Erkennung.

4.1. Anforderungen

ANF-Nummer	Beschreibung
P1-ANF01	Der Proxy durchsucht die Pakete nach Malware Eigenschaften.
P1-ANF02	Der Proxy leitet Pakete mit Malware Eigenschaften auf einen Fake C&C Server um.
P1-ANF03	Der Proxy ist ein SSL Splitting Proxy, um HTTPS Pakete zu entschlüsseln.
P1-ANF04	Der Client sendet Pakete mit Malware Eigenschaften.
P1-ANF05	Der Client unterstützt HTTPS.
P1-ANF06	Der Fake C&C Server unterstützt HTTPS.
P1-ANF07	Der Redirect wird gesetzt, bevor das Paket den Proxy verlässt, so dass das Paket sofort umgeleitet wird.

Tabelle 4.2.: Prototyp V1: Anforderungen

4.2. Nicht Funktionale Anforderungen

NFR-Nummer	Beschreibung
P1-NFR01	Der Benutzer darf keine Verschlechterung der Performance durch das System beim Benutzen des Netzwerks bemerken.

Tabelle 4.4.: Prototyp V1: Nicht Funktionale Anforderungen

4.3. Use Cases

Die Use Cases sind nur für den Prototyp V1 gültig und sind deshalb mit dem Pattern "P1"gekennzeichnet.

4.3.1. P1-UC01: Pattern erkennen

Use-Case-Name	**Pattern erkennen**
Umfang	Fiddlercore Proxy
Ebene	Unterfunktionsebene
Primärakteur	Fiddlercore Proxy
Vorbedingungen	Fiddlercore Proxy als zentraler Proxy konfiguriert
Nachbedingungen	Umleitung gesetzt (P1-UC02)
Standardablauf	1. Netzwerkpaket trifft beim Fiddlercore Proxy ein. 2. Paket wird auf Pattern geprüft. 3. Command & Control Pattern erkannt. 4. IP Adresse umleiten.
Offene Fragen	▪ Wie kann der verschlüsselte Payload von der Malware erkannt und entschlüsselt werden?

Tabelle 4.6.: Prototyp V1: P1-UC01 Pattern erkennen

4.3.2. P1-UC02: Umleitung setzen

Use-Case-Name	**Umleitung setzen**
Umfang	Fiddlercore Proxy
Ebene	Unterfunktionsebene
Primärakteur	Fiddlercore Proxy
Vorbedingungen	Fiddlercore Proxy als zentraler Proxy konfiguriert
Nachbedingungen	IP Adresse für Umleitung gespeichert
Standardablauf	1. IP Adressen speichern. 2. Bei Eintreffen eines Pakets mit der gespeicherten IP wird das Paket auf den Fake Command & Control Server umgeleitet.

Tabelle 4.8.: Prototyp V1: P1-UC02 Umleitung setzen

4.4. Design

Der Prototyp V1 wurde mit dem Framework Fiddler Core[25] [14] [6] implementiert, der eine grosse Bandbreite an Funktionen hat. Bei dem Framework handelt es sich um eine .NET Bibliothek, daher wurde der Fake C&C und der Client ebenfalls in C#[17] geschrieben.

4.4.1. Deployment Diagramm

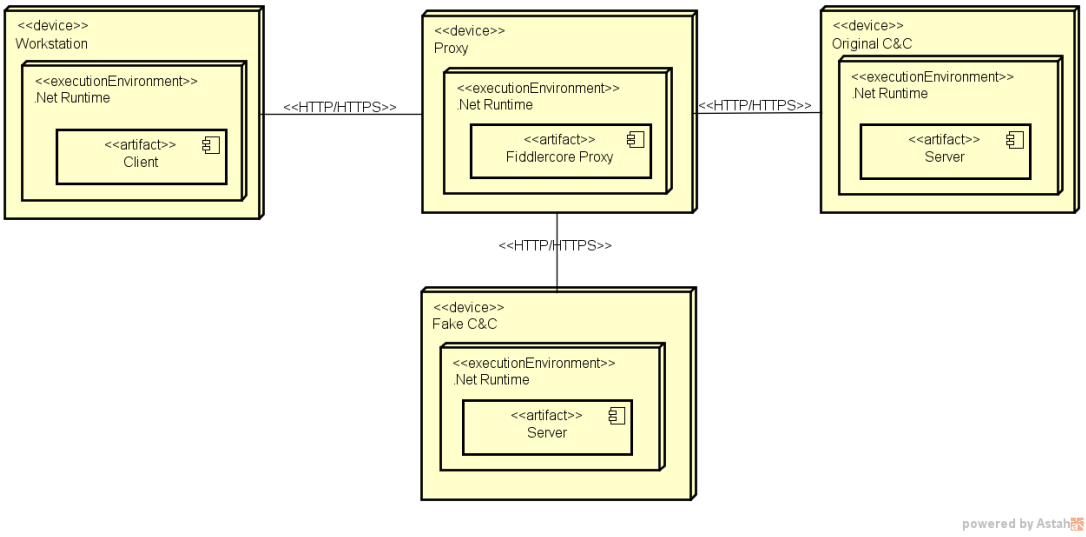


Abbildung 4.1.: Prototyp V1: Deployment Diagramm

4.4.2. Proxy

Der Proxy überprüft das Paket, das vom Client gesendet wird, auf einen vorhandenen Trojan-Header.

```
1 //Redirect according to Trojan-Header
2 if (oS.oRequest.headers.Exists("Trojan")) oS.host = "127.0.0.1:9443";
```

Auflistung 5: Prototyp V1: Beispielcode für Umleitung in Fiddler

Bei der Umleitung muss beachtet werden, ob es sich um eine HTTP oder HTTPS Verbindung handelt, damit sie auf den richtigen Port umgeleitet werden kann.

4.4.3. Fake C&C

Der Fake C&C ist ein einfacher HTTP/HTTPS Server, der auf die Requests des Clients mit einem einfachen 200 OK Response antwortet.

4.4.4. Client

Der Client sendet zufällige HTTP und HTTPS Requests an den Proxy und fügt einzelnen Requests einen Trojan-Header hinzu, nach welchem beim Proxy gefiltert wird.

4.4.5. System Sequenz Diagramm eines Redirects

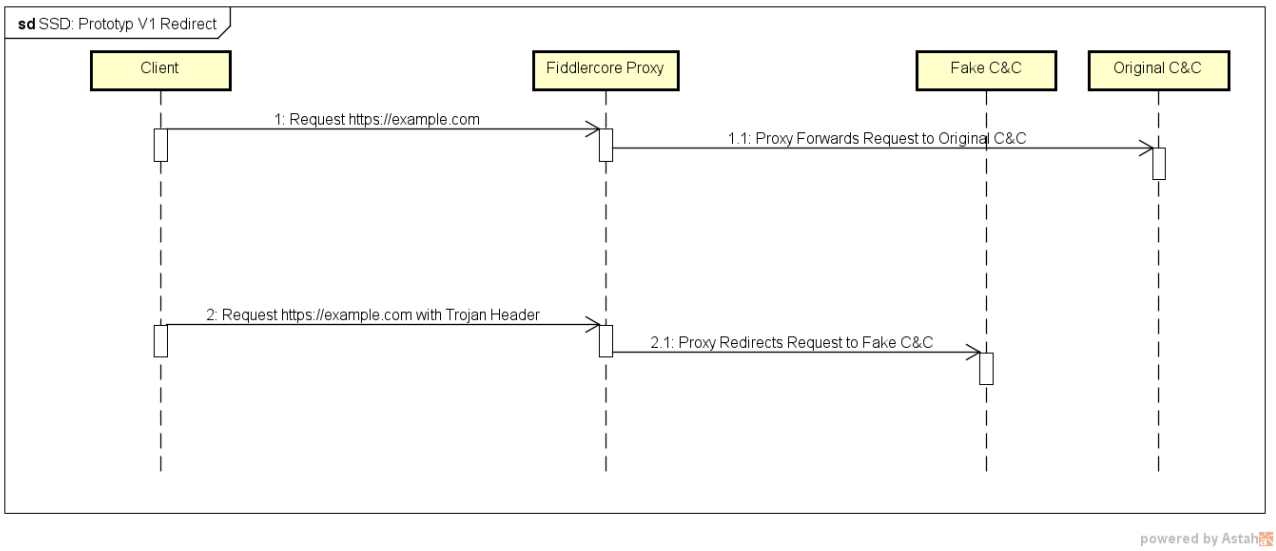


Abbildung 4.2.: Prototyp V1: System Sequenz Diagramm Redirect

4.5. Schlussfolgerung

Der Proxy besitzt viel Logik, was zu Problemen bei Performance und Wiederverwendbarkeit führt. Wenn das System in einer Firma integriert werden soll, die keine Änderungen an ihrem Netzwerk vornehmen will, dann ist eine Integration nicht möglich. Hier sollte wenn möglich alles aufgetrennt werden, damit nicht eine einzige Stelle so viele Verantwortlichkeiten besitzt (Separation of Concern). Im Abschnitt 6.1 wurde die Validation des Prototyp V1 durchgeführt, auf welcher die nachfolgende Entscheidung aufbaut.

4.5.1. Entscheidung

Aus genannten Gründen wird nach einer bessere aufgetrennten Lösung gesucht, und da der Host zu Beginn (falls HTTPS) bekannt sein muss, wird es nicht möglich sein, alle Requests in Real Time umleiten zu können. Das hat uns veranlasst, eine Zeitversetzte Erkennung zu entwickeln und dabei die Zuständigkeiten aufzutrennen, um bessere Performance und Wiederverwendbarkeit zu erhalten.

5. Proof of Concept Prototype V2

In diesem Kapitel wird folgende Frage beantwortet: *"Ist eine praxistaugliche Software, die die Problemstellung des Auftrags erfüllt, möglich?"*.

Der Prototyp V2 setzt auf die Architektur der Zeitversetzten Erkennung, Variante A.

5.1. Anforderungen

ANF-Nummer	Beschreibung
P2-ANF01	Der Proxy ist ein SSL Splitting Proxy, um HTTPS Pakete zu entschlüsseln.
P2-ANF02	Eine Schnittstelle ermöglicht es, den bestehenden Proxy einer Firma für das Loggen der Pakete zu verwenden.
P2-ANF03	Die Netzwerkpakete werden in einer Datenbank persistiert, um sie nach Paketen mit Malware Eigenschaften zu durchsuchen.
P2-ANF04	Der Client sendet Pakete mit Malware Eigenschaften.
P1-ANF05	Der Client unterstützt HTTPS.
P1-ANF06	Der Fake C&C Server unterstützt HTTPS.

Tabelle 5.2.: Prototyp V2: Anforderungen

5.2. Nicht funktionale Anforderungen

NFR-Nummer	Beschreibung
P2-NFR01	Der Benutzer darf keine Verschlechterung der Performance durch das System beim Benutzen des Netzwerks bemerken.
P2-NFR02	Vom Loggen bis zum Setzen der Umleitung dürfen maximal 10 Sekunden verstreichen.
P2-NFR03	Die Integration des Prototypen stellt möglichst geringe Anforderungen an eine bestehende Infrastruktur.
P2-NFR04	Die Umleitung der Pakete mit Malware Eigenschaften ist möglichst protokollunabhängig, um die Komplexität des Systems zu verringern
P2-NFR05	Das System ist in einzelne Instanzen aufgeteilt um dadurch eine bessere Wiederverwendbarkeit zu erhalten.

Tabelle 5.4.: Prototyp V2: Nicht Funktionale Anforderungen

5.3. Use Cases

Die Use Cases sind nur für den Prototyp V2 gültig und sind deshalb mit dem Pattern "P2"gekennzeichnet.

5.3.1. P2-UC01: Request loggen

Use-Case-Name	**Request loggen**
Umfang	Squid Proxy
Ebene	Unterfunktionsebene
Primärakteur	Squid Proxy
Vorbedingungen	Squid Proxy als zentraler Proxy konfiguriert
Nachbedingungen	Request in Elsaticsearch gespeichert
Standardablauf	1. Netzwerkpaket trifft beim Squid Proxy ein. 2. Squid Proxy sendet Paket via ICAP an ICAP Server (Pufferfish Logger) 3. ICAP Server (Pufferfish Logger) loggt ganzen Body in Elasticsearch

Tabelle 5.6.: Prototyp V2: P2-UC01 Request loggen

5.3.2. P2-UC02: Pattern erkennen

Use-Case-Name	**Pattern erkennen**
Umfang	Triggerfish Agent
Ebene	Unterfunktionsebene
Primärakteur	Triggerfish Agent
Vorbedingungen	Elasticsearch hat Logging Einträge
Nachbedingungen	Umleitung gesetzt (P2-UC03)
Standardablauf	1. Search Agent (Triggerfish Agent) findet Requests anhand von Pattern auf Elasticsearch 2. Search Agent (Triggerfish Agent) sendet Meldung an den Mandarinfish Router für Umleitung
Offene Fragen	Wie kann der verschlüsselte Payload von der Malware entschlüsselt und erkannt werden.

Tabelle 5.8.: Prototyp V2: P2-UC02 Pattern erkennen

5.3.3. P2-UC03: Umleitung setzen

Use-Case-Name	**Umleitung setzen**
Umfang	Mandarinfish Router
Ebene	Unterfunktionsebene
Primärakteur	Mandarinfish Router
Vorbedingungen	Triggerfish hat Meldung für Umleitung versendet
Nachbedingungen	Iptable wurde gesetzt
Standardablauf	1. Meldung für Umleitung wird empfangen 2. Meldung wir interpretiert und eine Iptable gesetzt

Tabelle 5.10.: Prototyp V2: P2-UC03 Umleitung setzen

5.4. Design

5.4.1. Systemübersicht

Die Systemübersicht ist die logische Sicht auf das System, sie soll auf einfache Weise einen Überblick auf das Gesamtsystem ermöglichen.

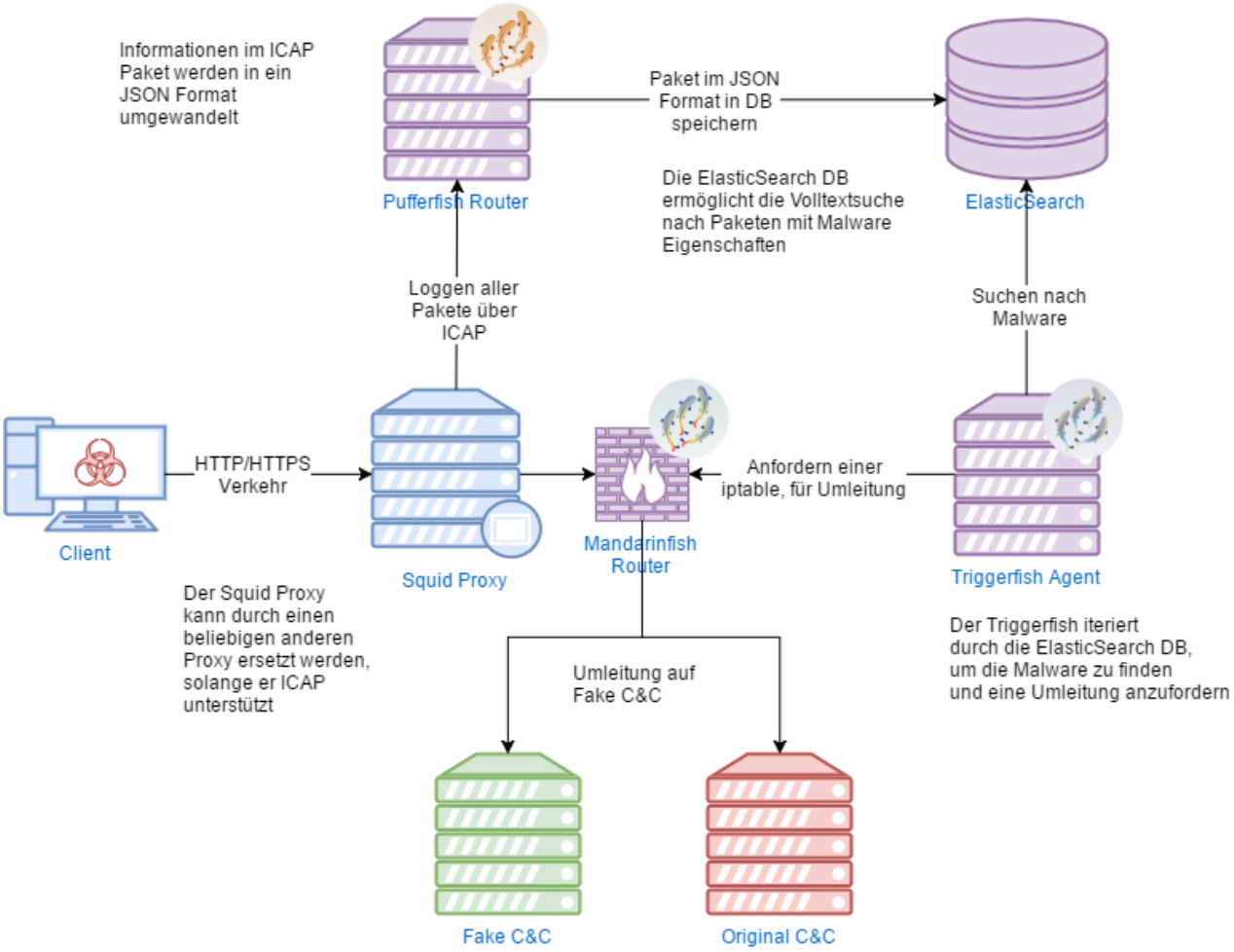


Abbildung 5.1.: Prototyp V2: Systemübersicht

5.4.2. Mandarinfish Router

Der Mandarinfish Router verfügt über eine Representational State Transfer (REST) Application Programming Interface (API), die es ermöglicht, einen Redirect anzufordern, der mit Hilfe einer Iptable festgelegt wird.

Titel	Setze Redirect
URL	/v1/redirect
Method	POST

Data Params	{
	original: [string],
	redirect: [string]
	}

Success Repsonse	**Code:** 202 **Content:** -
Error Response	**Code:** 500 **Content:** {error: 'Iptables error'} **Code:** 400 **Content:** {error: 'Malformed Syntax'}
Kommentar	Beim Original handelt sich um die IP des Original C&C und bei Redirect um die IP des Fake C&C Servers

Tabelle 5.12.: Prototyp V2: Setze Redirect API

Bespiel: Setzen eines Redirects

```
curl -XPOST -d "{"original": "[Original IP]" , "redirect": "[Redirect IP]"}"
 ↪  http://mandarin/v1/redirect
```

Auflistung 6: Prototyp V2: Beispiel für Redirect Request

5.4.3. Squid Proxy

Die Squid Proxy Konfiguration wurde so auch in die Fish Tank Suite übernommen 7.4.4.

5.4.4. Pufferfish Logger

Der Pufferfish loggt alle Requests in die Elasticsearch, dazu wird der vom Squid Proxy gesendete ICAP REQMOD geparst und an Logstash über TCP gesendet. Der Squid Proxy bekommt vom Pufferfish für jeden REQMOD eine 204 (No modifications needed) Response zurück.

Beispiel: ICAP REQMOD mit Body

```
REQMOD icap://127.0.0.1:1344 ICAP/1.0
Host: 127.0.0.1:1344
Date: Mon, 31 Oct 2016 12:09:53 GMT
Encapsulated: req-hdr=0, req-body=164
Allow: 204 # No modifications needed
X-Client-IP: [Client IP] # Ursprünglicher Client

POST https://example.com/ HTTP/1.1
User-Agent: curl/7.43.0
Accept: */*
Content-Length: 14
Content-Type: application/x-www-form-urlencoded
Host: example.com

e # Content Length
{"count": "0"}
0 # End of Content
```

Auflistung 7: Prototyp V2: Beispiel für ICAP REQMOD von Squid Proxy

Beispiel: JSON an Elasticsearch

```
{
  "message": {
    "@icapheaders": { //ICAP REQMOD Headers
      "Host": "127.0.0.1:1344",
      "Date": "Thu, 27 Oct 2016 12:28:36 GMT",
      "Encapsulated": "req-hdr=0, req-body=160",
      "Allow": "204",
      "X-Client-IP": "[Client IP]"
    },
    "@request": { // HTTP Request
      "method": "POST",
      "uri": "https://example.com/",
      "headers": {
        "User-Agent": "curl/7.43.0",
        "Accept": "application/json",
        "Content-Type": "application/json",
        "Content-Length": "10",
        "Host": "example.com"
      },
      "body": "e\r\n{\"count\":0}\r\n0\r\n\r\n"
    }
  }
}
```

Auflistung 8: Prototyp V2: Beispiel für ein JSON an Elasticsearch

5.4.5. Logstash

Logstash bekommt die Logs über TCP, bringt diese mit einem JSON Filter in die richtige Form und speichert diese in der Elasticsearch.

5.4.6. Triggerfish Agent

Der Triggerfish führt Suchanfragen anhand eines Patterns über die Elasticsearch REST API aus und setzt über die Mandarinfish REST API die nötigen Iptables für die gefundenen C&C Hosts.

5.4.7. Client

Der Client erstellt Requests und sendet diese an den Original C&C. Im Body dieser Requests befindet sich ein Count Feld, das vom C&C Server inkrementiert wird. Bei einer gewissen Höhe, die beim Triggerfish definiert wird, soll ein Redirect gesetzt werden.

Beispiel: Count Request

```
curl -XPOST -d "{"count": 0}" http://original/v1/count
```

Auflistung 9: Prototyp V2: Beispiel für Count Request

5.4.8. Fake C&C und Original C&C

Beim Fake C&C und Original C&C handelt es sich um den gleichen Server, welcher die Requests vom Client annimmt und darauf antwortet. Zu Demonstrationszwecken zählt der Originale C&C den Count um 1 hoch, der Fake C&C hingegen um 100.

Titel	Zähle Counter hoch
URL	/v1/count
Method	POST

Data Params	{
	count: [int]
	}

Success Repsonse	**Code:** 200 **Content:** {count: [int]}
Error Response	**Code:** 400 **Content:** Bad Request
Kommentar	Der gesendete Wert von Count wird beim Original C&C um 1 erhöht und beim Fake C&C um 100.

Tabelle 5.14.: Prototyp V2: Count Request

5.4.9. Deployment

Da es nicht notwendig ist, für jede Teilsoftware einen separaten Server zu verwenden, wurden mehrere Teile des Systems zusammengenommen. Der Squid Proxy, Mandarinfish und Pufferfish laufen auf dem gleichen Server. Die Elasticsearch benötigt einen eigenen Server, da der Arbeitsspeicherbedarf relativ hoch ist.

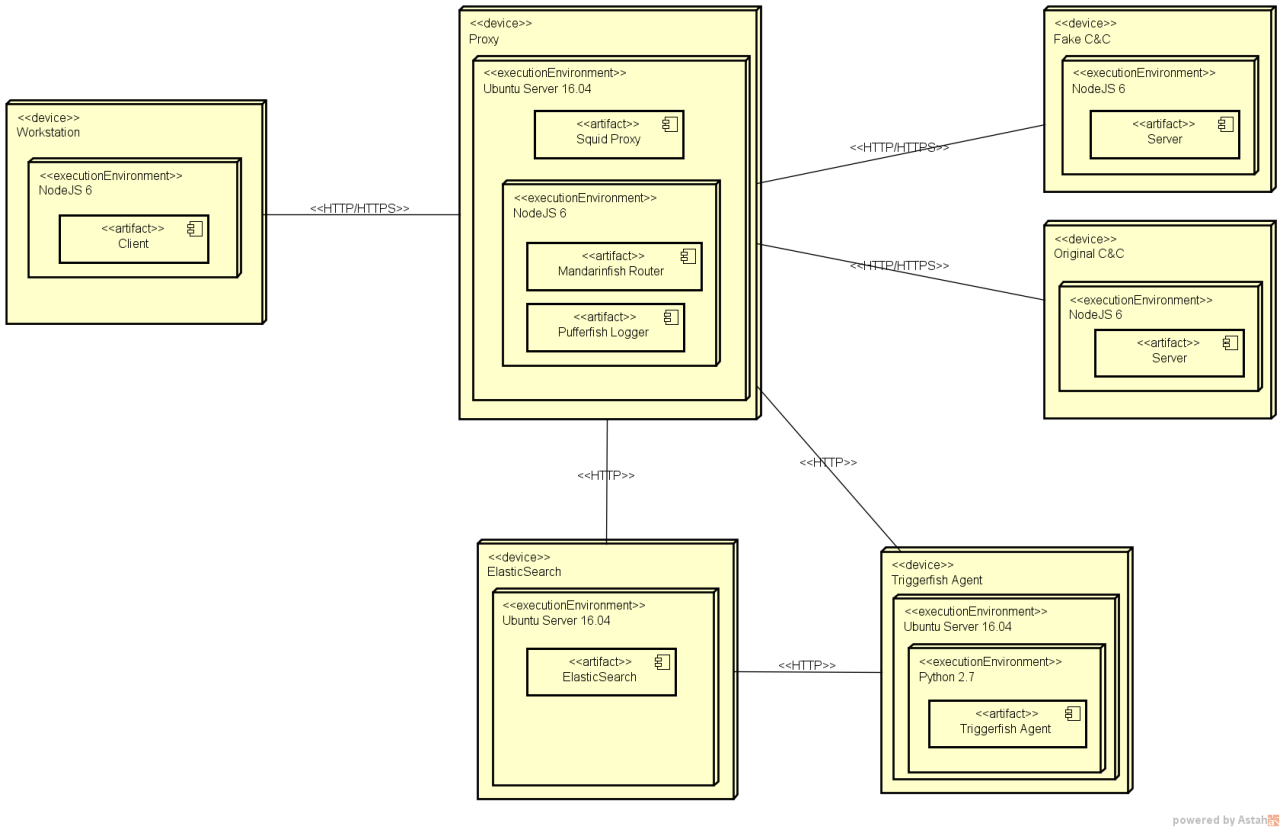


Abbildung 5.2.: Prototyp V2: Deployment Diagramm

System Sequenz Diagramm

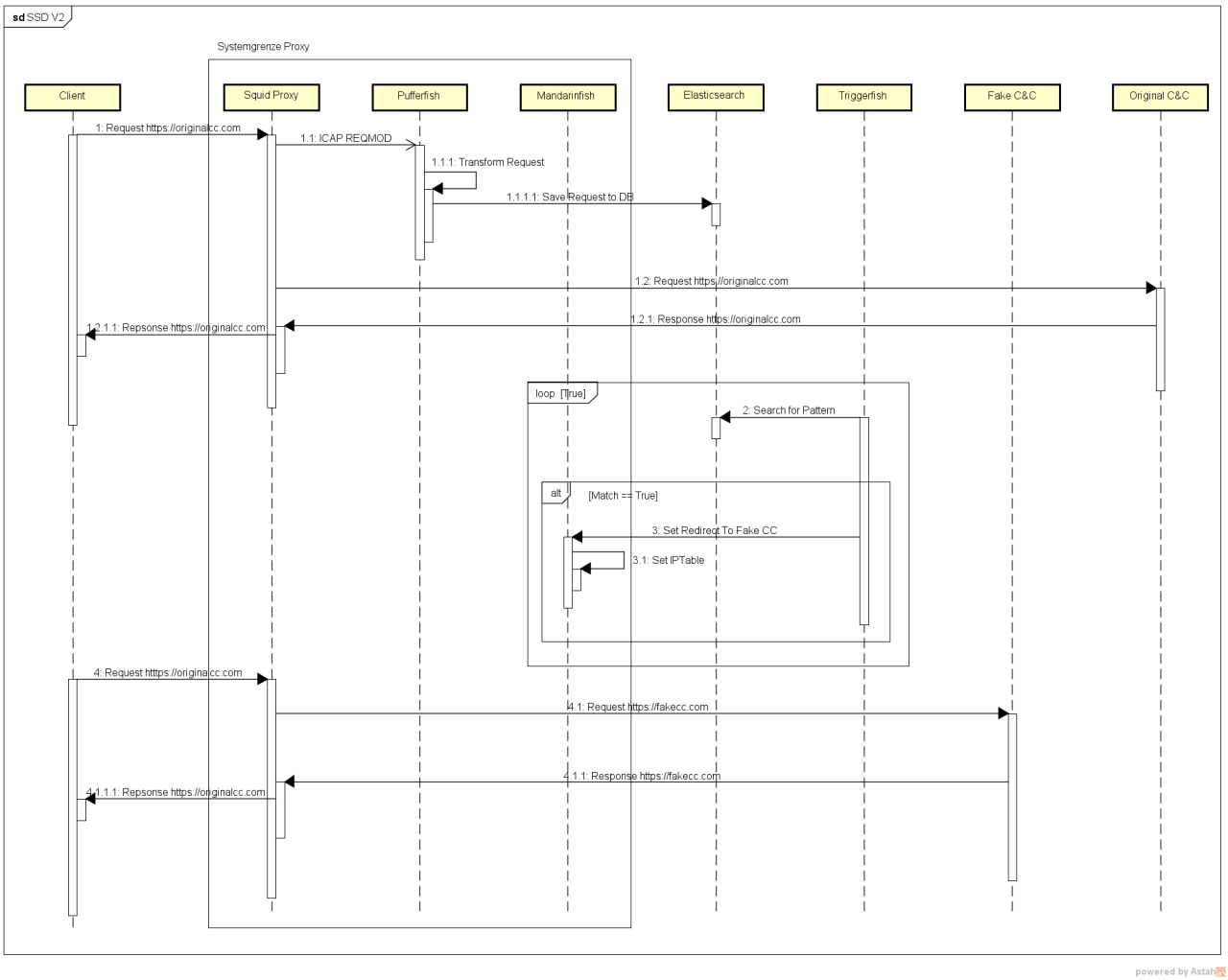


Abbildung 5.3.: Prototyp V2: System Sequenz Diagramm

5.5. Schlussfolgerung

Durch das Aufteilen auf verschiedene Programme konnten die Zuständigkeiten auf mehrere Teilsysteme verteilt werden. Das erhöht die Performance und verbessert die Erweiterbarkeit, ausserdem sind nun verschiedene Architekturen denkbar, was einigen Firmen die Integration erleichtern wird. Das Einsetzen von Elasticsearch erlaubt nun auch das Betrachten des Zeitverhaltens von Requests, zudem existiert eine Datenbasis, auf die jederzeit wieder zurückgegriffen werden kann. Das Umleiten mit Iptables vereinfacht die Behandlung diverser Protokolle, das heisst die Unterscheidung zwischen HTTP und HTTPS sowie anderen Protokollen entfällt.

Das Gesamtsystem ist zwar komplexer geworden, die gewonnenen Vorteile überwiegen jedoch eindeutig.
Im Abschnitt 6.2 wurde der Prototyp V2 getestet und hat ebenfalls zur nachfolgenden Entscheidung beigetragen.

Entscheidung

Die Probleme des Prototyp V1 konnten behoben werden. Der Prototyp V2 erlaubt durch die Verteilung der Logik, das Erweitern des Systems auf vielfältige Weise. Es wurde beschlossen, mit dem Prototyp V2 fortzufahren.

6. Proof of Concept Validation

In diesem Kapitel wird folgende Frage beantwortet: *"Ist eine praxistaugliche Software, die die Problemstellung des Auftrags erfüllt, möglich?"*.

Das Kapitel ist dabei in die Teile Validation des Prototyps V1 im Abschnitt 6.1 und die Validation des Prototyps V2 im Abschnitt 6.2 unterteilt.

6.1. Testing Prototyp V1

In diesem Abschnitt wurde der Prototyp V1 getestet, ob dieser die Anforderungen abdeckt.

6.1.1. Infrastruktur

Die Software wird lokal ausgeführt, und alle Services werden automatisch gestartet.

6.1.2. Testfälle

Die Testfälle werden mit dem Pattern P1 (für Prototyp V1) gekennzeichnet.

P1-T1: Software ausführen

ID/Bezeichnung	P1-T01 - Software starten
Beschreibung	
Testvoraussetzung	Prototyp V1 aufgesetzt, lauffähig und alle benötigen Libraries installiert.
Testschritte	1. Software starten 2. Ausgabe auf Kommandozeile interpretieren
Erwartetes Ergebnis	Die Software startet und sendet dabei HTTP und HTTPS Request an verschiedene Ziele. Requests, die einen Malware Header beinhalten, werden dabei auf den Fake C&C HTTP/HTTPS Server umgeleitet.

Tabelle 6.2.: Validation: Prototyp V1 Testfall 1

6.1.3. Testprotokoll

Protokoll Testfall P1-T1

Software hat gestartet und angefangen, HTTP und HTTPS Requests zu machen, dabei werden einem alle offenen Sessions angezeigt. Dabei sind vor allem die Responses vom Localhost interessant, der Fake C&C bekommt also umgeleitete Requests und antwortet auf diese, somit scheint der Layer 7 redirect zu funktionieren.

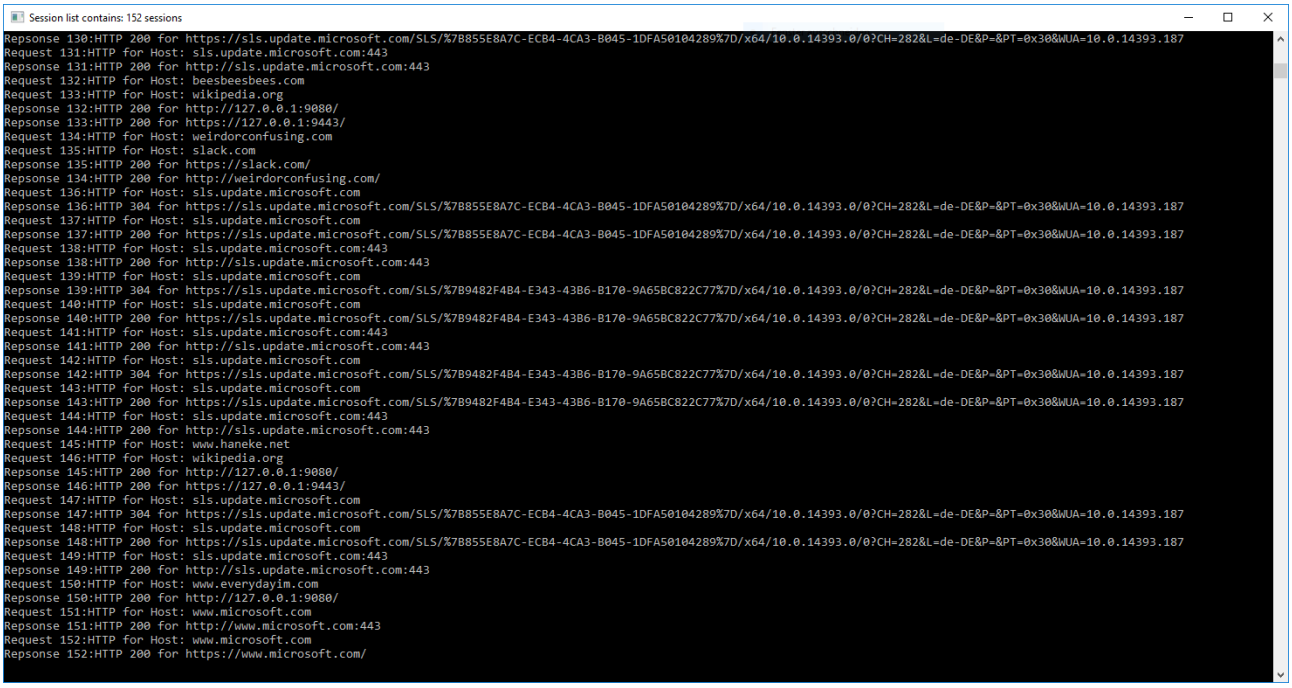


Abbildung 6.1.: Validation: Session List von FiddlerCore

HTTP Connect Host Da bei jeder HTTPS Verbindung ein HTTP Connect Request gesendet wird, welcher nur den Host beinhaltet, kann nicht anhand des Headers umgeleitet werden, die nachfolgenden Requests schon. Somit müsste der Host bekannt sein, den man weiterleiten möchte, ohne diesen geht es nicht.

```
1 CONNECT 127.0.0.1:8877 HTTP/1.1
2 Host: wikipedia.org
3 Proxy-Connection: Keep-Alive
```

Auflistung 10: Validation: HTTP Connect Beispiel

6.1.4. Zusammenfassung

Layer 7 Redirect funktioniert zwar, aber der CONNECT Request kann so oder so nicht direkt abgefangen werden, es sei denn, man kennt den Host bevor man umleitet. Durch diese Erkenntnis

entschieden wir uns, einen weiteren Prototypen zu erstellen, der auf eine zeitversetzte Erkennung setzt.

6.2. Testing Prototyp V2

In diesem Abschnitt wurde der Prototyp V2 getestet.

6.2.1. Infrastruktur

Der Prototyp V2 wurde auf der Testumgebung getestet, und um eine Malware nachzubilden, wurde eine einfache Client-Server Applikation geschrieben.

6.2.2. Testfälle

Die Testfälle werden mit dem Pattern P2 (für Prototyp V2) gekennzeichnet.

P2-T1: Über Squid Proxy Browsen

ID/Bezeichnung	P2-T01 - Über Squid Proxy Browsen
Beschreibung	
Testvoraussetzung	Prototyp V2 aufgesetzt und konfiguriert. Test Client im gleichen Netz wie Proxy.
Testschritte	1. CURL GET auf Example.com 2. CURL POST auf Example.com mit Body 3. CURL PUT auf Example.com mit Body 4. Auf Kibana überprüfen, ob Requests geloggt wurde
Erwartetes Ergebnis	Alle gemachten Requests mit CURL werden in Kibana angezeigt und somit durch den Prototyp V2 geloggt.

Tabelle 6.4.: Validation: Prototyp V2 Testfall 1

P2-T2: Redirect provozieren

ID/Bezeichnung	P2-T01 - Redirect provozieren
Beschreibung	
Testvoraussetzung	Prototyp V2 aufgesetzt und konfiguriert Test Client im gleichen Netz wie Proxy. Keine Iptable gesetzt, die bereits umleitet.
Testschritte	1. CURL POST mit Body, der einem Suchpattern entspricht 2. Auf Kibana prüfen,ob geloggt wurde 3. Nochmaliger CURL POST auf gleichen Destination Host 4. Rückmeldung nun vom Fake C&C Server
Erwartetes Ergebnis	Der Request triggert den Triggerfish und dieser setzt über den Mandarinfish eine Iptable, dadurch wird der nächste Request an den gleichen Host auf den Fake C&C umgeleitet.

Tabelle 6.6.: Validation: Prototyp V2 Testfall 2

P2-T3: Pseudo Malware

ID/Bezeichnung	P2-T01 - Pseudo Malware
Beschreibung	
Testvoraussetzung	Prototyp V2 aufgesetzt und konfiguriert Test Client im gleichen Netz, wie Proxy. Keine Iptable gesetzt, die bereits umleitet. "Original C&C" erreichbar.
Testschritte	1. Client starten und abwarten 2. Kibana prüfen, ob geloggt wird 3. Rückgaben auf Kommandozeile überprüfen
Erwartetes Ergebnis	Der Client sendet fortlaufend Requests, welche durch den "Original C&C" bestätigt werden, auf der Kommandozeile kann das nachvollzogen werden, sobald die Umschaltung auf den Fake C&C stattfand.

Tabelle 6.8.: Validation: Prototyp V2 Testfall 3

6.2.3. Testprotokoll

P2-T1: Über Squid Proxy Browsen

Es sollen nun 3 Requests über den Squid Proxy gemacht werden, und diese müssen vom Pufferfish Logger geloggt werden und in die Elasticsearch gespeichert werden.

GET

```
1 curl http://example.com --proxy mandarin:3128
```

Auflistung 11: Validation: CURL GET über Proxy

POST

```
1 curl http://example.com -X POST -d "{"count":4000}" --proxy mandarin:3128
```

Auflistung 12: Validation: CURL POST über Proxy

PUT

```
curl http://example.com -X PUT -d "{"count":4000}" --proxy mandarin:3128
```

Auflistung 13: Validation: CURL PUT über Proxy

Sobald alle Requests durchgeführt wurden, sollte dies nun im Kibana auftauchen und dank Elasticsearch durchsuchbar sein.

@request.method	@request.body	@request.headers.Host
PUT	c {count:4000} 0	example.com
POST	c {count:4000} 0	example.com
GET		example.com

Abbildung 6.2.: Validation: Requests in Kibana

P2-T2: Redirect provozieren

Der Triggerfish sucht nach dem Pattern "100", sobald 1 Request gemacht wird, der dieses Pattern beinhaltet, muss eine Iptable auf dem Mandarinfish existieren, welche den Destination Host auf den Fake C&C umleitet.

```
curl http://example.com -X POST -d "{"count":100}" --proxy mandarin:3128
```

Auflistung 14: Validation: Request mit POST 100

Nach den Requests muss also nach kurzer Zeit eine Iptable auf dem Mandarin Host existieren, die alle Requests auf den Fake C&C umleitet.

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DNAT       all  --  anywhere             [Destination IP]        to:[FakeCC IP]

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
```

Auflistung 15: Validation: Iptables List

Sobald nochmals ein neuer Request gemacht wird auf den gleichen Host, kommt der Response vom Fake C&C.

```
curl http://example.com -X POST -d "{"count":100}" --proxy mandarin:3128
#Fake C&C Repsonse
Cannot POST /
```

Auflistung 16: Validation: Erneuter Request an Example.com

P2-T3: Pseudo Malware

Nun soll das ganze dynamisch getestet werden, dazu wurde eine Software programmiert, welche einen einfachen Counter hoch zählt. Sobald dieser 100 erreicht hat, muss ein Redirect gesetzt werden.

Der Fake C&C zählt dabei um 100 hoch, man sieht auf der Konsole, sobald die Umschaltung stattgefunden hat.

```
14 Second
send: 100
{"count":101}
16 Second
send: 102
{"count":103}
17 Second
send: 104
{"count":105}
18 Second
send: 106
{"count":107}
19 Second
send: 108
{"count":109}
20 Second
send: 110
{"count":210}
22 Second
send: 211
{"count":311}
```

Auflistung 17: Validation: Client Requests an den Server

Die Umleitung benötigte knappe 6 Sekunden, danach wurde auf den Fake C&C Server umgeleitet.

6.2.4. Zusammenfassung

Der Redirect funktioniert, allerdings müsste die Reaktionszeit noch verbessert werden, da momentan noch nicht entschlüsselt werden muss. Dies ist aber dank der Separierung der Systeme relativ gut möglich, so kann jeder Service in der Performance verbessert werden.

6.3. Schlussfolgerung

Der Prototyp V1 hat nicht vollständig so funktioniert wie geplant, da eine Echtzeit Analyse nicht so einfach zu bewerkstelligen ist. Der Prototyp V2 hat allerdings gute Erfolgsaussichten, alleine schon durch die Aufteilung in verschiedene eigene Systeme.

7. Fish Tank Suite

In diesem Kapitel wird folgende Frage beantwortet: *"Wie würde eine Software aussehen, die für den Einsatz in einem realen Szenario geeignet ist?"*.

Bei der Fish Tank Suite handelt es sich um eine Software, die aus den Erkenntnissen der Analyse und Prototypen hervorging. Diese Software soll eine praxistaugliche Lösung für die Probleme der Ausgabenstellung darstellen.

7.1. Akteure

Rolle	**Beschreibung**
Auftraggeber	Der Auftraggeber möchte einen lauffähigen Prototypen, der die Machbarkeit des Projekts beweist.
Entwickler	Der Entwickler ist für die Entwicklung der Software zuständig.
IT-Security Engineer	Anwender der Software.

Tabelle 7.2.: Fish Tank Suite: Akteure

7.2. User Stories

7.2.1. Sprint 3

Epic 3: Release Candidate 1

Ende Sprint soll ein erster Release Candidate erstellt sein.

Story 3.1: Chunked Data

Als Entwickler will ich chunked Data speichern können.

Akzeptanzkriterien

- Ganze Requests und Responses können gespeichert werden.
- Begin und End of Data von Internet Content Adaptation Protocol (ICAP) werden nicht mehr geloggt.

Story 3.2: Log Daten

Als Entwickler will ich eine performante Analyse der Log Daten.

Akzeptanzkriterien

- Das Durchsuchen der Elasticsearch wurde im Vergleich zum Prototyp effizienter.
- Das Loggen von Daten wurde effizienter.

Story 3.3: Verschlüsselter Payload

Als Security Engineer will ich verschlüsselten Payload analysieren können.

Akzeptanzkriterien

- Payload Entschlüsselung möglich.
- Verschlüsselter Payload wird geloggt.

Story 3.4: Verschlüsselter Payload erkennen

Als Entwickler will ich erkennen, ob der Payload verschlüsselt ist.

Akzeptanzkriterien

- Anhand von Keywords möglich zu testen, ob entschlüsselt.

Story 3.5: Fake Command & Control (C&C)

Als Entwickler will ich einen Fake C&C zum Testen.

Akzeptanzkriterien

- Requests können weitergeleitet werden.
- Responses, die bösartige Malware beinhalten, können nicht weiter senden.

Story 3.6: Campaign

Als Security Engineer will ich eine Campaign definieren.

Akzeptanzkriterien

- Campaign JSON mit möglichen Optionen definiert.

Story 3.7: Projektdokumentation

Als Entwickler will ich eine aktuelle Dokumentation.

Akzeptanzkriterien

- Dokumentation wird laufend erweitert.

7.2.2. Sprint 4

Epic 4: Release Candidate 2

Ende Sprint soll ein zweiter Release Candidate erstellt sein.

Story 4.1: Projektdokumentation

Als Entwickler will ich eine aktuelle Dokumentation.

Akzeptanzkriterien

- Dokumentation wird laufend erweitert.

Story 4.2: Zentrale Verwaltung

Als Security Engineer will ich mehrere Systeme über einen zentralen Punkt verwalten.

Akzeptanzkriterien

- Campaign an zentralem Punkt gespeichert.
- Änderungen für andere Systeme werden übernommen.

Story 4.3: AES128 Payload

Als Auftraggeber will ich Payload mit AES128 entschlüsseln können.

Akzeptanzkriterien

- AES128 Entschlüsselung möglich.

Story 4.4: Fake C&C

Als Entwickler will ich mehrere Fake C&C betreiben

Akzeptanzkriterien

- Der Fake C&C kann auf verschiedenen Ports gestartet werden.
- Umleitung funktioniert für verschiedene Ports.

Story 4.5: Elasticsearch Filter

Als Entwickler will auf der Elasticsearch entschlüsseln können.

Akzeptanzkriterien

- Entschlüsselung über Elasticsearch Filter möglich.

7.3. Nicht Funktionale Anforderungen

NFR-Nummer	Beschreibung
P2-NFR01	Der Benutzer darf keine Verschlechterung der Performance durch das System beim Benutzen des Netzwerks bemerken.
P2-NFR02	Vom Loggen bis zum Setzen der Umleitung dürfen maximal 10 Sekunden verstreichen.
P2-NFR03	Die Integration des Prototypen stellt möglichst geringe Anforderungen an eine bestehende Infrastruktur.
P2-NFR04	Die Umleitung der Pakete mit Malware Eigenschaften ist protokollunabhängig, um die Komplexität des Systems zu verringern
P2-NFR05	Das System ist in einzelne Instanzen aufgeteilt, um dadurch eine bessere Wiederverwendbarkeit zu erhalten.

Tabelle 7.4.: Fish Tank Suite: Nicht Funktionale Anforderungen

7.4. Design

7.4.1. Systemübersicht

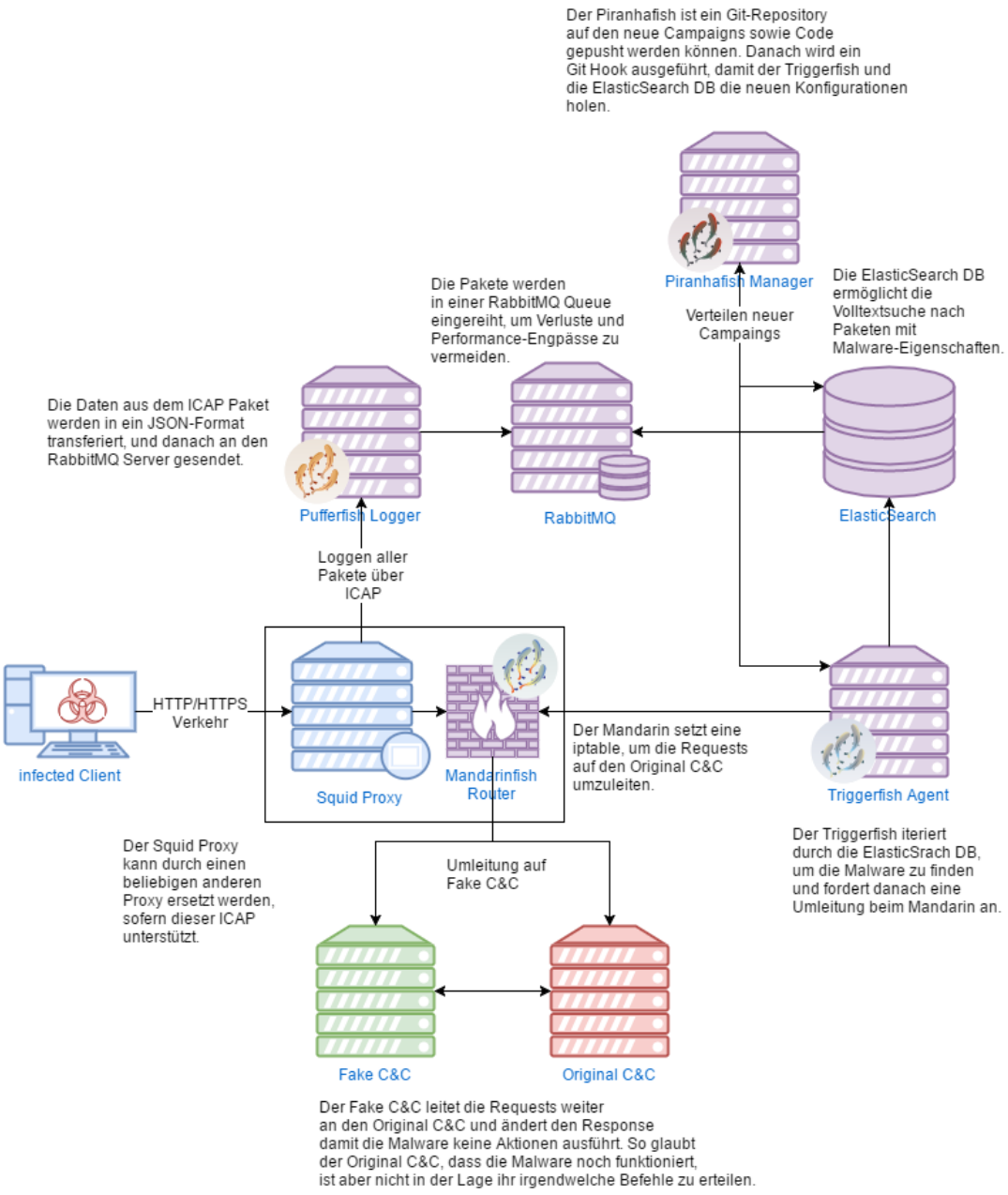


Abbildung 7.1.: Fish Tank Suite: Systemübersicht

7.4.2. Deployment Model

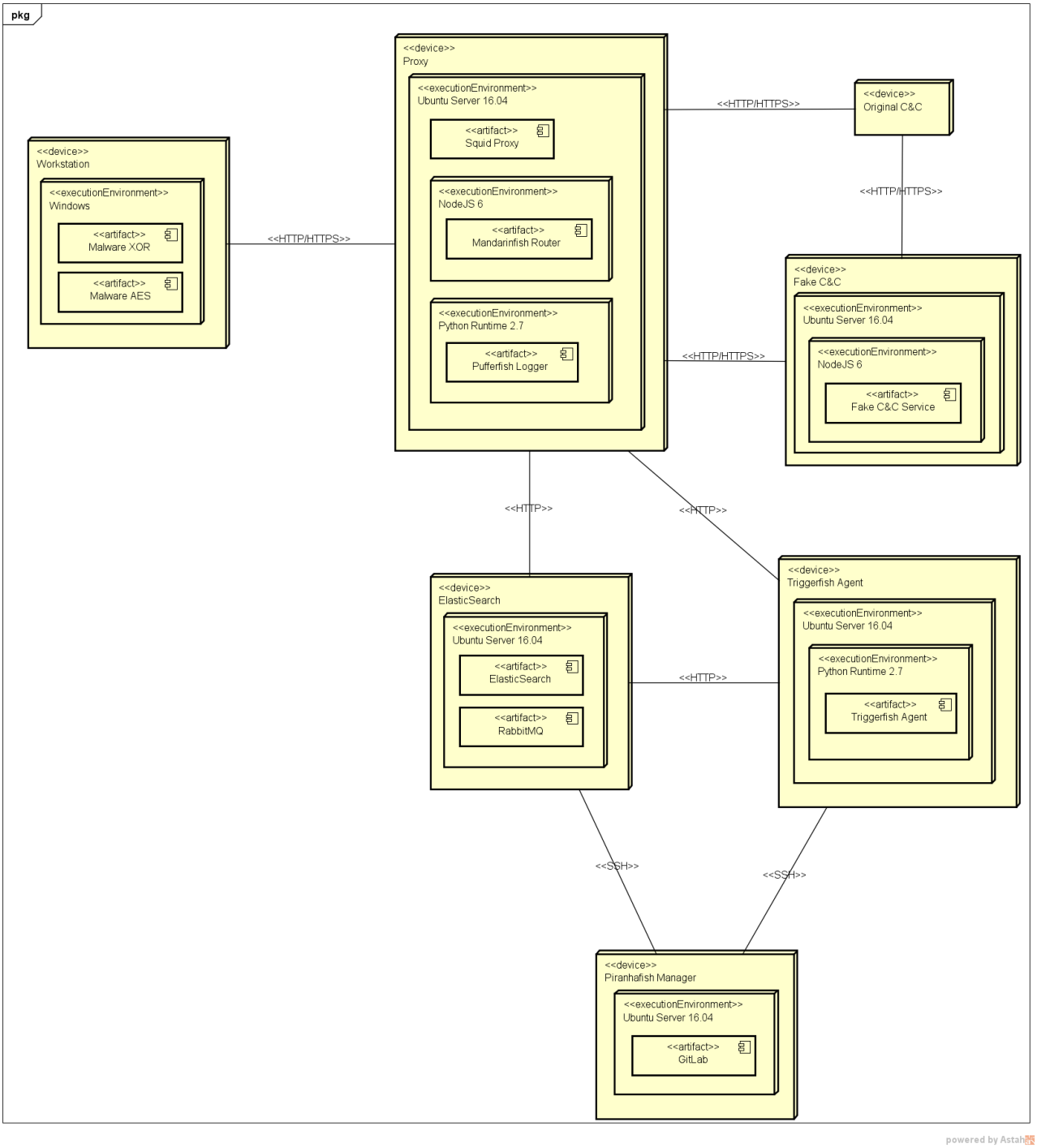


Abbildung 7.2.: Fish Tank Suite: Deployment Model

7.4.3. System Sequence Diagram

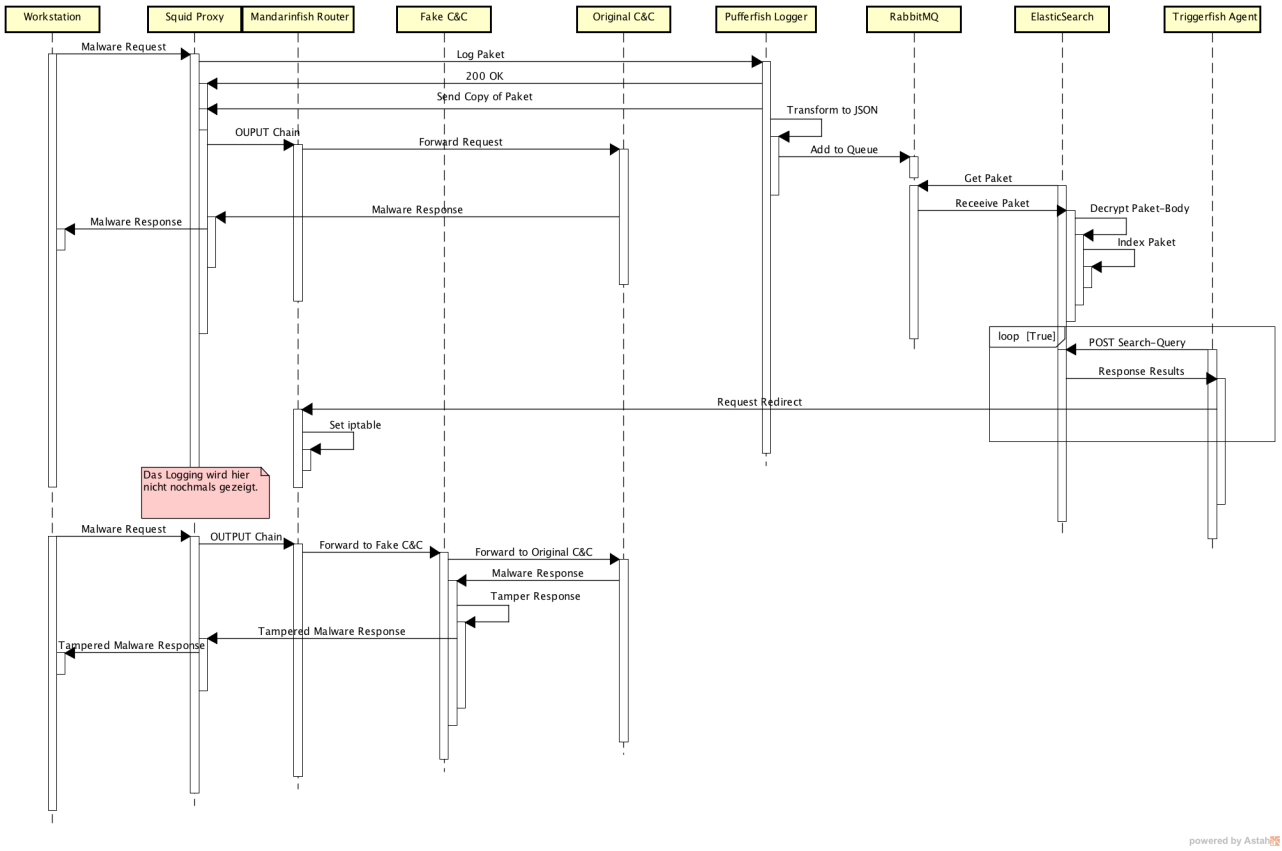


Abbildung 7.3.: Fish Tank Suite Sequence Diagram

7.4.4. Squid Proxy

Der Squid Proxy ist die zentrale Schnittstelle, da alle HTTP Requests und Responses über ihn laufen und er entscheidet, ob gewisse Requests oder Responses geloggt werden sollen oder ob der HTTPS Verkehr von gewissen Domains nicht entschlüsselt werden soll.

Access Control List (ACL)

Um gewisse Domains vom Entschlüsseln oder Loggen auszunehmen, kann eine ACL für die jeweilige Option (Secure Socket Layer (SSL) Splitting, ICAP Logging) erstellt werden.

```
1 acl icap_bypass_to_localnet dst 10.0.0.0/8
2 acl icap_bypass_to_localnet dst 172.16.0.0/12
3 acl icap_bypass_to_localnet dst 192.168.0.0/16
```

Auflistung 18: Fish Tank Suite: Squid Proxy ACL Beispiel

SSL Splitting

Um auch verschlüsselten Verkehr einsehen zu können, müssen alle HTTPS Verbindungen "gebumpt" werden, ausser man will Ausnahmen für gewisse Domains, dies wäre über eine ACL möglich. Zusätzlich wird ein Zertifikat benötigt, welches dann an die Clients ausgeliefert wird.

```
ssl_bump stare all
ssl_bump bump all
http_port 3128 ssl-bump generate-host-certificates=on dynamic_cert_mem_cache_size=4MB
 ↪  cert=/etc/squid/certs/myca.pem
```

Auflistung 19: Fish Tank Suite: Squid Proxy SSL Bump

Die Option "generate-host-certificates" setzt die jeweilige aufgerufene Domain als CN für das generierte Zertifikat.

ICAP Client

Der Squid Proxy beinhaltet den ICAP Client, welcher in regelmässigen Abständen den Pufferfish Logger anfragt, ob sich was an den ICAP Options geändert hat und sendet jeden Request oder Response, der nicht durch eine ACL ausgenommen wurde, an den Pufferfish Logger. Dabei müssen REQMOD und RESPMOD als einzelne Services konfiguriert sein, sehr wichtig ist das "bypass=0", welches das Umgehen des ICAP Servers nicht erlaubt, es sei denn, es gibt eine ACL dafür.

```
# Pufferfish ICAP Service
icap_service pufferfish1 reqmod_precache icap://127.0.0.1:1344/log_requests bypass=0
 ↪  on-overload=wait
icap_service pufferfish2 respmod_precache icap://127.0.0.1:1344/log_responses bypass=0
 ↪  on-overload=wait
```

Auflistung 20: Fish Tank Suite: Squid Proxy ICAP Server Config

```
# Adaptation Access
adaptation_access pufferfish1 deny icap_bypass_to_localnet
adaptation_access pufferfish2 deny icap_bypass_to_localnet
```

Auflistung 21: Fish Tank Suite: Squid Proxy Adaptation Access Config

Da der Pufferfish den ganzen Request oder Response benötigt, muss vom Squid Proxy auch der ganze gesendet werden und nicht bloss die Preview (einige Bytes des Requests oder Responses).

```
# Disable Preview
icap_preview_enable off
```

Auflistung 22: Squid Proxy Preview Konfiguration

7.4.5. Pufferfish Logger

Der Pufferfish Logger loggt jeden Request oder Response inklusive dessen Body. Beim Prototyp V2 wurde noch darauf verzichtet die Responses zu loggen, welche nun ebenfalls geloggt werden. Die Logs werden dabei an eine RabbitMQ gesendet, wo diese zwischengespeichert werden bis sie durch Logstash abgearbeitet werden.

REQMOD

Die Requests müssen für die Elasticsearch in ein verständliches Format gebracht werden, falls der Body mal leer sein sollte, wird ein leerer String gesetzt, damit bei der späteren Bearbeitung nicht ein Fehler bezüglich fehlendem Body auftritt.

```
{
  "method" : "CONNECT",
  "message": "REQMOD",
  "uri" : "hsr.ch:443",
  "headers" : {
           "host" : ["hsr.ch:443"],
           "user-agent" : ["curl"]
   },
  "body" : "",
  "level" : "INFO",
  "type" : "logstash",
  "host" : "mandarin",
  "logger_name" : "python-logstash-logger",
  "protocol_version": "HTTP/1.1",
  "path": "/home/ccproxy/Git/pufferfish-logger/__main__.py"
}
```

Auflistung 23: Fish Tank Suite: REQMOD logging Beispiel

RESPMOD

Dasselbe beim Response, hier wird der ganze Body mitgesendet, ausser es handelt sich um einen Dateityp, der im "Transfer-Ignore" gesetzt ist.

```
{
   "headers": {
     "date": ["Fri, 25 Nov 2016 17:31:12 GMT"],
     "content-length": ["1270"],
     "server": ["EOS (lax004/28A3)"],
     "last-modified": ["Fri, 09 Aug 2013 23:54:35 GMT"],
     "expires": ["Fri, 02 Dec 2016 17:31:12 GMT"],
     "etag": ["\"359670651\""],
     "content-type": ["text/html"],
     "accept-ranges": ["bytes"],
     "cache-control": ["max-age=604800"]
   },
   "path": "/home/ccproxy/Git/pufferfish-logger/__main__.py",
   "level": "INFO",
   "host": "mandarin",
   "logger_name": "python-logstash-logger",
   "message": "RESPMOD",
   "body": "<html>...",
   "type": "logstash",
   "tags": [],
   "status": ["HTTP/1.1","200","OK"]
}
```

Auflistung 24: RESPMOD logging Beispiel

7.4.6. RabbitMQ

Der Pufferfish erstellt beim Loggen (falls noch nicht vorhanden) einen neuen Exchange, der auf "Durable" gesetzt wird, damit dieser einen Neustart des RabbitMQ Servers überlebt.

Der Exchange wird als Fanout Exchange definiert, damit jede Queue, die sich mit dem Exchange verbindet, alle Messages bekommt. Routing Keys spielen keine Rolle.

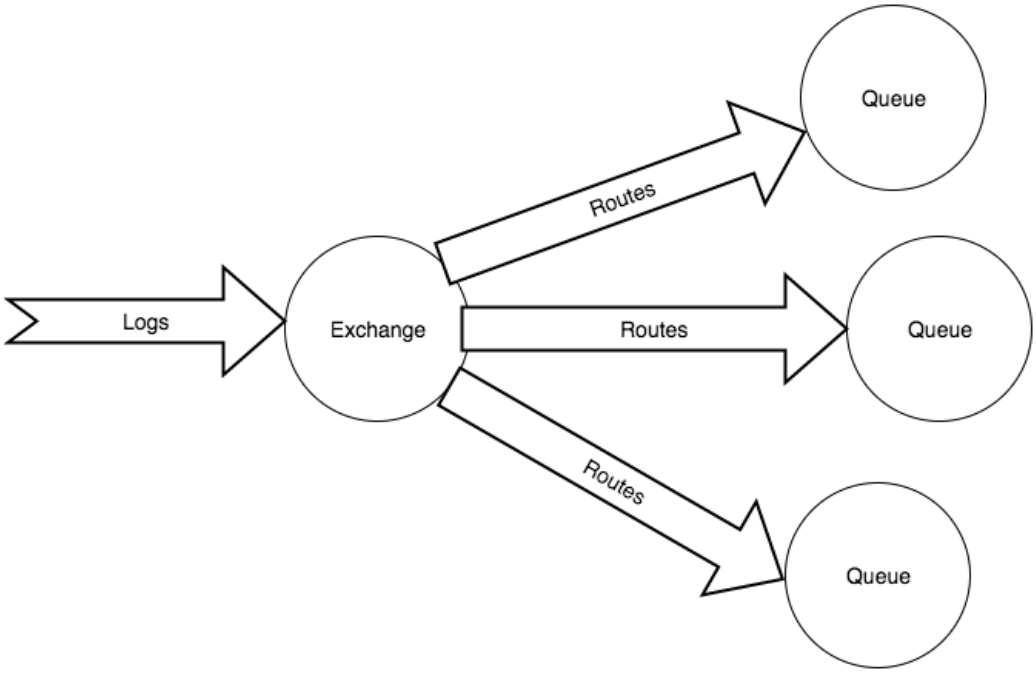


Abbildung 7.4.: Fish Tank Suite: Fanout Exchange

7.4.7. Logstash

Logstash durchläuft 3 Schritte für jede entgegengenommene Message aus der RabbitMQ.

Input Als erstes nimmt Logstash die Message aus der Queue entgegen und gibt sie im Plain Format weiter an den nächsten Schritt.

Filter

Beim Filter handelt es sich nun um eine speziell für die vorhandene Malware geschriebenen Software, die verschlüsselte Payloads (XOR oder AES) entschlüsseln kann und markiert die Entschlüsselten mit Campaign Name und Cipher Art.

Domain Modell

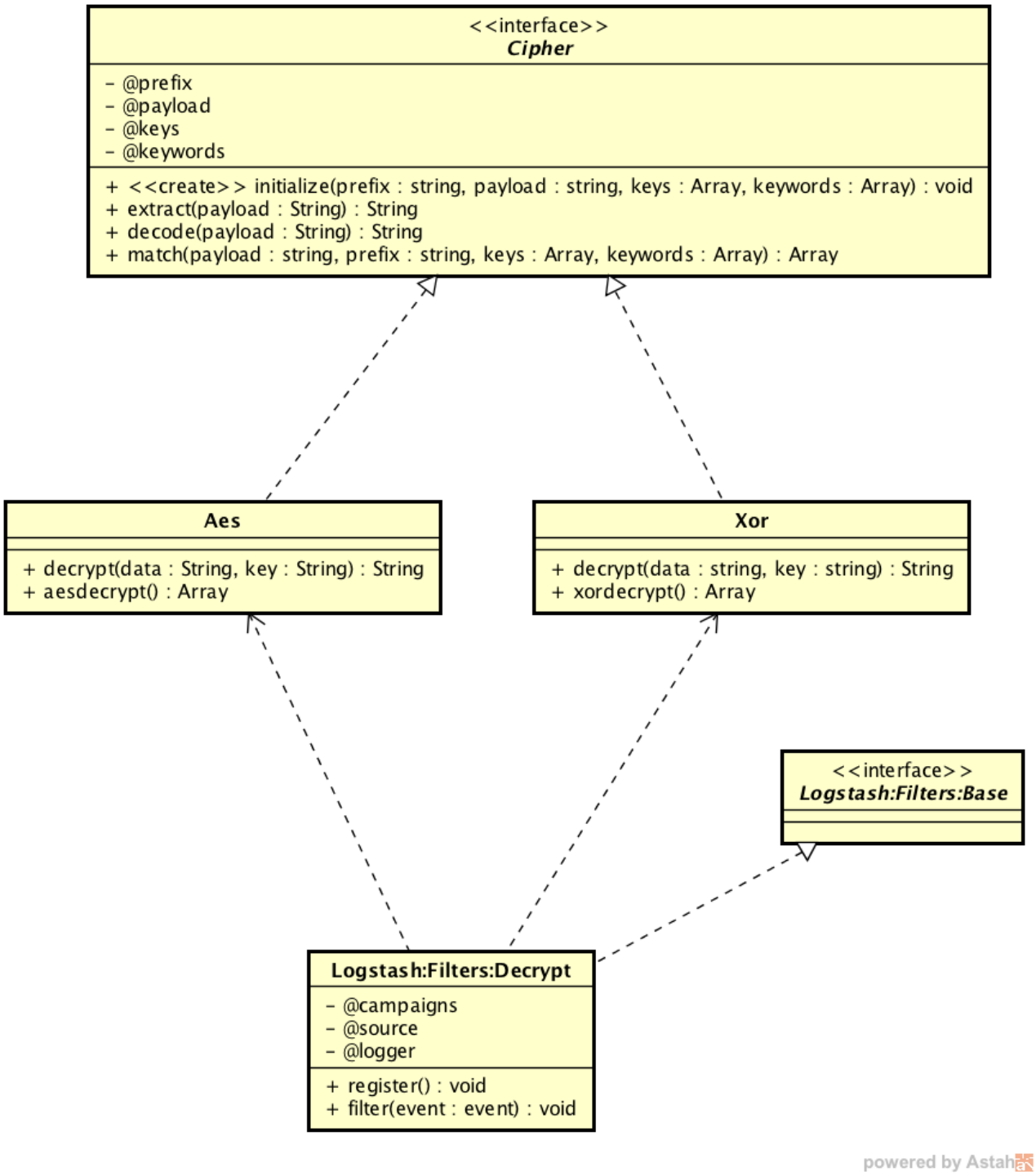


Abbildung 7.5.: Fish Tank Suite: Logstash Decrypt Filter Klassendiagramm

Ablauf

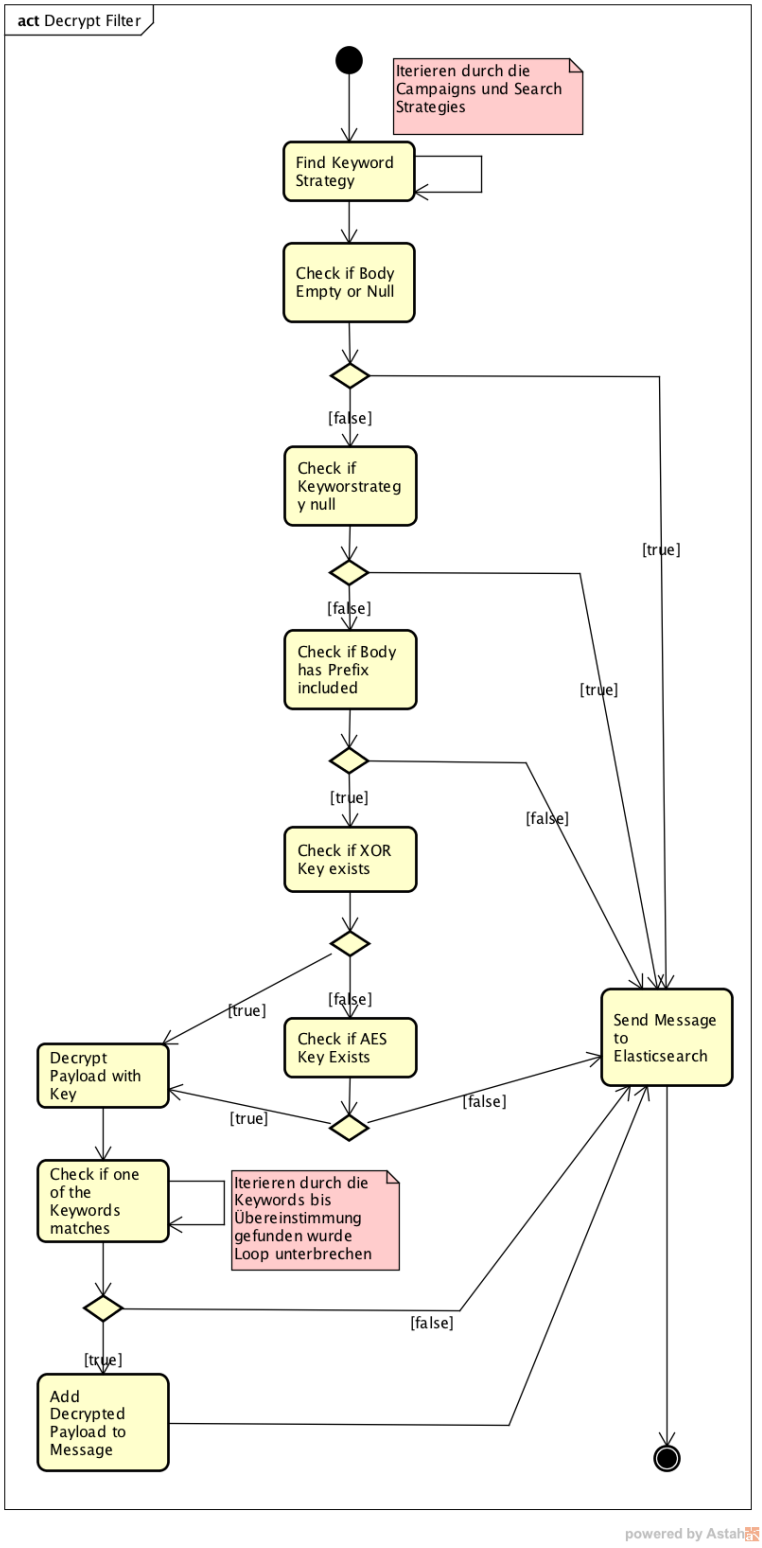


Abbildung 7.6.: Fish Tank Suite: Logstash Decrypt Filter Ablauf

Output Zm Abschluss wird die bearbeitete Message aus der Queue in eine Elasticsearch gespeichert, dabei werden die Messages alle unter dem gleichen Index abgespeichert.

```
1 logstash-YYYY.MM.DD
```

Auflistung 25: Fish Tank Suite: Elasticsearch Index Pattern

7.4.8. Campaign

Bei der Campaign handelt es sich um die zentrale Konfiguration, um eine bekannte Malware zu erkennen.
Die Campaign bietet dabei folgende Konfigurationsmöglichkeiten:

name Name der Campaign, dieser Name wird auch beim Logstash Decrypt Filter zum taggen verwendet.

redirectIP IP des Fake C&C, auf welchen weitergeleitet werden soll.

prefix Möglicher Prefix des Payloads, welcher nicht zum Entschlüsseln benötigt wird.

portRedirects Für welche Ports ein Redirect gesetzt werden soll. Der *orignalPort* ist der Port auf dem Original C&C. Der *redirectPort* ist der Port vom Fake C&C.

SearchStrategies Beinhaltet alle möglichen Strategien, um eine Malware zu erkennen. Für die Fish Tank Suite existieren 2 mögliche Strategien, nämlich die KeywordStrategy und die UriSequence-Strategy.

encryption Die Encryption Arten, die auftreten könnten. Für die Fish Tank Suite sind das XOR oder AES und deren Schlüssel.

```
{
  "name": "C1",
  "redirectIp": "ip",
  "portRedirects": [
    {
      "originalPort": "port",
      "redirectPort": "port"
    },
    {
      "originalPort": "port",
      "redirectPort": "port"
    }
  ],
  "SearchStrategies": [
    {
      "name": "uri",
      "type": "UriSequenceStrategy",
      "sequence": {
        "entry": {"uri": "pattern"},
        "next": {"uri": "pattern", "timeSpan": "seconds"}
      }
    },
    {
      "name": "payload",
      "type": "KeywordStrategy",
      "prefix" : "prefix",
      "keywords": [
        "keyword"
      ]
    }
  ],
  "encryption" : {
          "xor" : ["key"],
          "aes" : ["key"]
  }
}
```

Auflistung 26: Fish Tank Suite: Campaign Beispiel mit allen Optionen

7.4.9. Triggerfish Agent

Der Triggerfish Agent sucht anhand von SearchStrategies nach der Malware und setzt über den Mandarinfish einen Redirect. Die Funktionsweise der verschiedenen Strategies wird im Kapitel 2 Abschnitt 2.6 beschrieben. Die Suche nach den Malwares kann parallel ausgeführt werden, was im Prototyp V2 noch nicht möglich war.

Domain Model

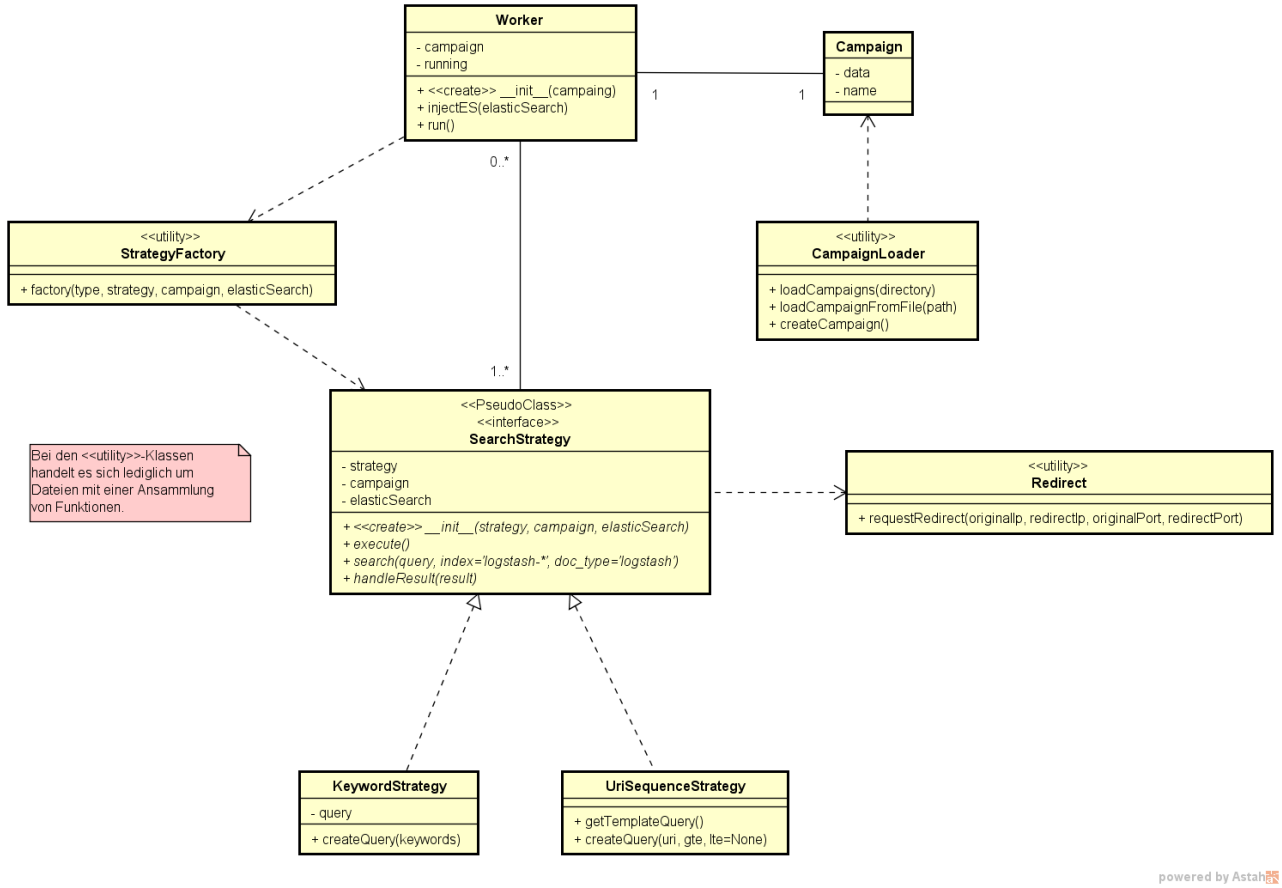


Abbildung 7.7.: Fish Tank Suite: Triggerfish Class Diagram

Worker Der Worker implementiert die Thread-Klasse und kann deshalb mit anderen Workern parallel ausgeführt werden. Jeder Worker besitzt genau eine Campaign, die mehrere Strategies beinhalten kann.

Campaign Die Campaign kapselt die Informationen, die das Campaign-Konfigurationsfile liefert.

CampaignLoader Der CampaignLoader ist wie alle anderen Utilities keine Klasse, sondern eine Datei, die ausschliesslich Funktionen beinhaltet. Wichtig ist nur die Funktion "loadCampaigns()", welche alle JSON-Dateien lädt, die die Konfiguration einer Campaign beinhalten.

SearchStrategy Die Searchstrategy ist eine PseudoKlasse, was in anderen Sprachen, wie zum Beispiel Java, als abstrakte Klasse bezeichnet wird. Die SearchStrategy bietet einen Konstruktor, einige Hilfsfunktionen und die Funktion "execute()", welche die Suche nach der Malware startet.

StrategyFactory Die StrategyFactory instantiiert die konkreten Strategies.

Redirect Die Funktion "requestRedirect()" sendet die Informationen für einen Redirect an den Mandarinfish-Router.

7.4.10. Mandarinfish Router

Der Mandarinfish Router ist von der Funktionsweise her fast gleich wie beim Prototyp V2. Es ist nun auch möglich den Port umzuleiten, das hat den Vorteil, dass der Fake C&C nicht auf Port 80 und 443 laufen muss. Zudem ist es nun möglich mehrere Fake C&C auf dem selben Host zu betreiben. Im Prototyp V2 war es noch möglich doppelte und ungültige Iptables zu setzten. Das wurde, wie im Activity Diagram ersichtlich, verbessert.

Representational State Transfer (REST) Application Programming Interface (API)

Titel	Setze Redirect
URL	/v1/redirect
Method	POST
Data Params	{ originalIp: [string], redirectIp: [string], originalPort: [string], redirectPort: [string] }
Success Repsonse	**Code:** 200 **Content:** -
Error Response	**Code:** 400 **Content:** - **Code:** 400 **Content:** {error: 'Malformed Syntax'}

Tabelle 7.6.: Fish Tank Suite: Setze Redirect API

Activity Diagram

Die Parameter der REST API müssen validiert werden, da Shell-Befehle ausgeführt werden, ansonsten wären Command-Injections möglich. Es dürfen auch keine doppelten oder ungültigen Iptables gesetzt werden.

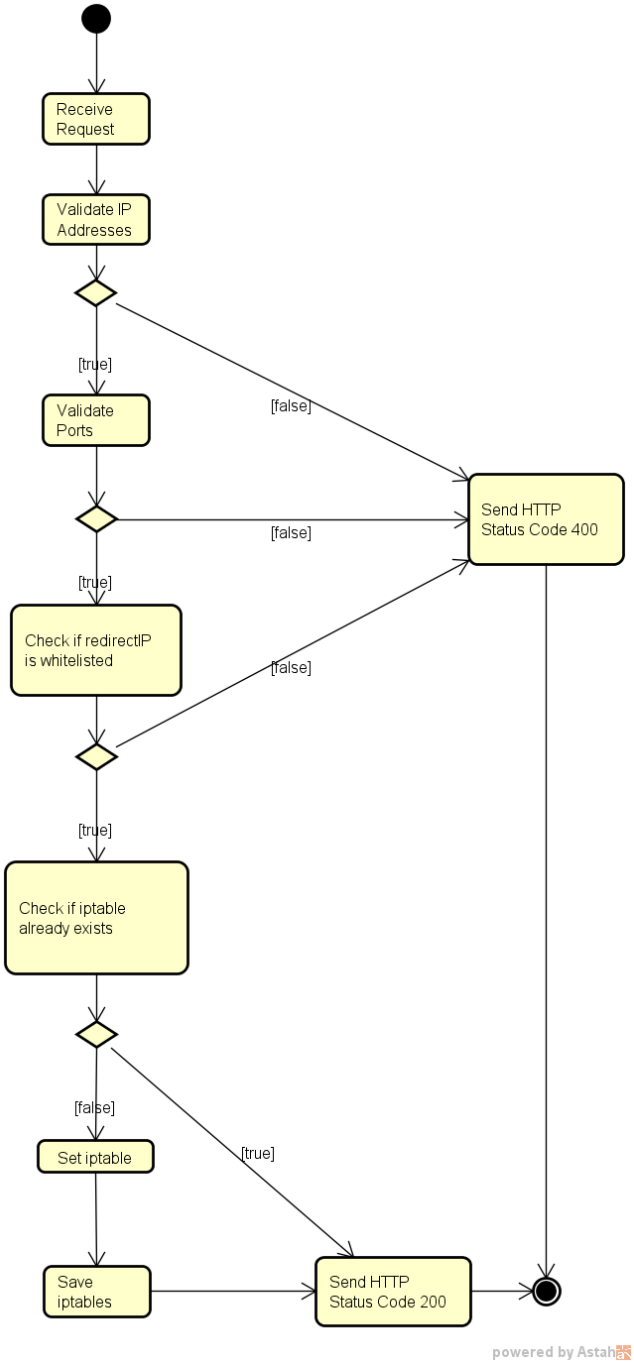


Abbildung 7.8.: Fish Tank Suite: Mandarin Activity Diagram

Iptable persistence

Damit die Iptables nach einem Neustart des Servers noch vorhanden sind, werden die Iptables im Schritt "Save Iptables" in eine Datei geschrieben. Die Software "Iptable-persistence" stellt nach einem Neustart die Iptables wieder her.

7.4.11. Piranhafish Manager

Beim Piranhafish Manager handelt es sich um eine Gitlab[1] Instanz, welche für die Handhabung der Campaigns des Triggerfish und Decrypt Filters zuständig ist. Dabei wird über Git Hooks sichergestellt, dass alle Systeme immer auf dem neusten Stand gegenüber dem Piranhafish sind. Dazu existiert ein Campaign Git Repository, welches die Repos vom Logstash Decrypt Filter und des Triggerfishs als seine Git Submodules konfiguriert hat. Auf den jeweiligen Zielsystemen befinden sich ebenfalls Mirrors des Campaign Git Repository, welche eigene Hooks besitzen, um die auf dem jeweiligen Host aufgeschaltete Software zu aktualisieren.

Piranhafish Hook Der Prianhafish Hook pusht auf alle seine Origins, sobald der push auf das Campaign Repo beendet wurde (post-recieve).

Triggerfish Hook Der Triggerfish Hook wird ausgeführt, sobald der Push vom Piranhafish beendet wurde, dabei wird auch die neuste Version des Triggerfishs gepullt.

Logstash Decrypt Filter Hook Der Logstash Decrypt Filter Hook wird nach dem Push vom Piranhafish ausgeführt und pullt dabei die neuste Version des Decrypt Filters.

[1] https://gitlab.com

Ablauf

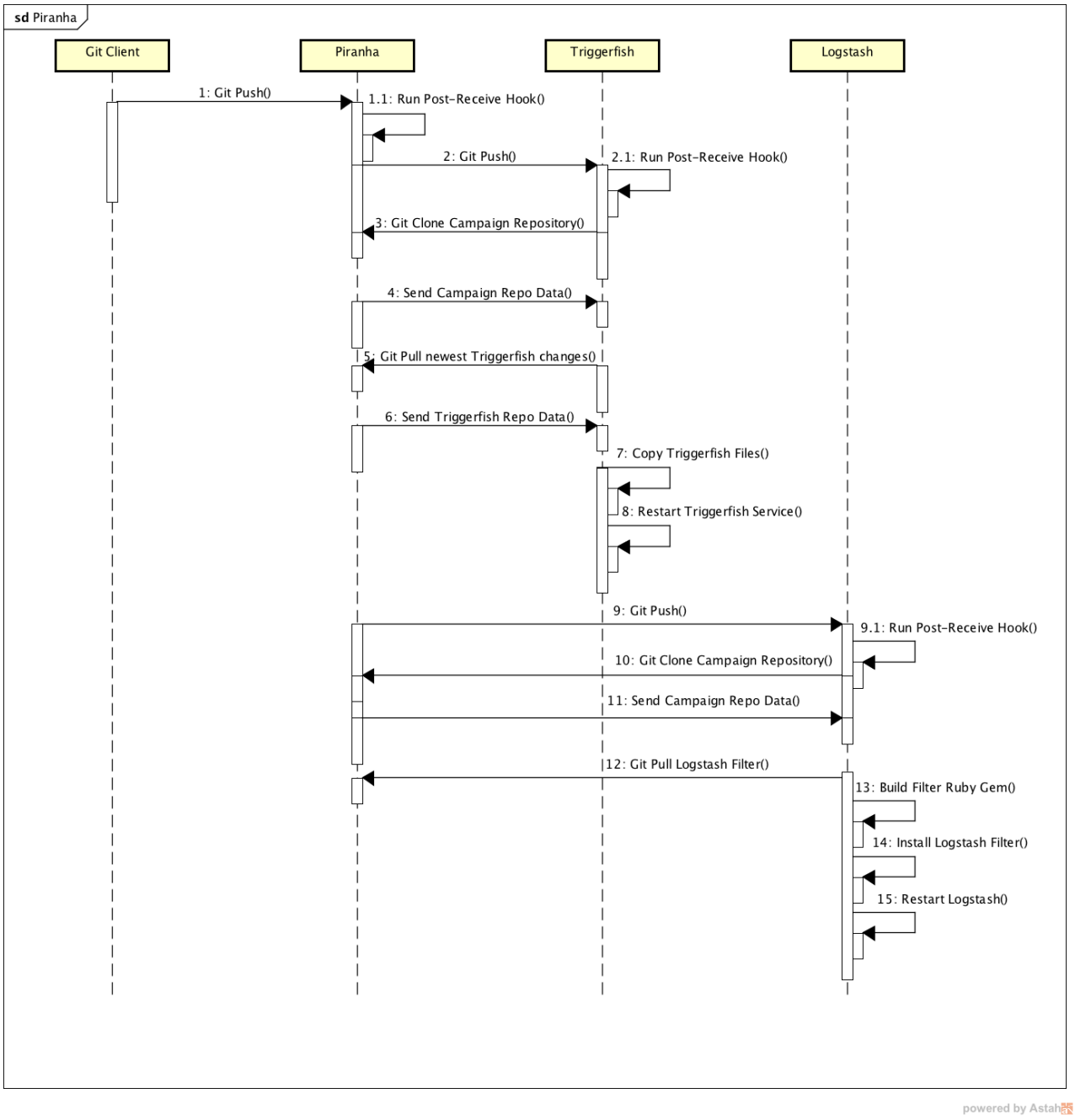


Abbildung 7.9.: Fish Tank Suite: Piranha Git Hooks

7.4.12. Fake Command & Control

Der Fake C&C wird erst aktiv, sobald der erste Request der Malware redirected wurde, dabei hört der Fake C&C nur einen Port ab und zwar für HTTP.

Titel	Request Forwarden
URL	*
Method	POST
Data Params	Keine, ganzer Request wird kopiert.
Success Response	Sende Response von Original C&C an Client zurück
Error Response	-

Tabelle 7.8.: Fish Tank Suite: Request forwarding

Titel	Response modifizieren
URL	/pattern/
Method	POST
Data Params	Keine, ganzer Request wird kopiert.
Success Response	Sende modifizierten Response zurück an Client.
Error Response	-

Tabelle 7.10.: Fish Tank Suite: Request forwarding

7.5. Testing

In diesem Abschnitt wird die Fish Tank Suite im Detail getestet. Es wird darauf verzichtet, bereits getestete Funktionen aus der Proof of Concept Validierung nochmals zu testen.

7.5.1. Testfälle

Die Testfälle werden mit dem Pattern FTS (Fish Tank Suite) gekennzeichnet.

FTS-T01 Keyword Strategy

ID/Bezeichnung	P1-T01 - Keyword Strategy
Beschreibung	Die Keyword Strategy versucht in verschlüsseltem Payload einen Match für ein gewisses Keyword zu finden.
Testvoraussetzung	Fish Tank Suite vollständig aufgesetzt, alle Systeme aktiv , keine Iptable auf dem Mandarinfish gesetzt, Malware konfiguriert und Squid Proxy als Systemproxy hinterlegt.
Testschritte	1. XOR oder AES Malware starten 2. Auf Kibana prüfen, ob Requests geloggt werden 3. Auf dem Mandarinfish prüfen, ob Iptable gesetzt 4. Fake C&C Access log für Keyword Strategy prüfen, ob Requests umgeleitet werden
Erwartetes Ergebnis	Requests können durch den Logstash Decrypt Filter entschlüsselt und durch den Triggerfish erkannt werden. Der Triggerfish setzt dann auf dem Mandarinfish die nötige Iptable und der Fake C&C bekommt die umgeleiteten Requests.

Tabelle 7.12.: Fish Tank Suite: Testfall 1

FTS-T02 Uri Sequence Strategy

ID/Bezeichnung	FTS-T02 - Uri Sequence Strategy
Beschreibung	Die Uri Sequence Strategy sucht nach einem Intervall und ein gewisses Pattern in den Ziel-Uris der Malware Requests.
Testvoraussetzung	Fish Tank Suite vollständig aufgesetzt, alle Systeme aktiv , keine Iptable auf dem Mandarinfish gesetzt und Fake Malware konfiguriert und funktionsfähig.
Testschritte	1. Fake Malware starten und mehrere Intervalle abwarten (30s/Intervall) 2. Auf Mandarinfish prüfen, ob Iptable gesetzt wurde 3. Anhand der Konsolenausgaben vom Client prüfen, ob Request über Fake C&C umgeleitet wird.
Erwartetes Ergebnis	Der Triggerfish setzt auf dem Mandarinfish die nötige Iptable und der Fake C&C bekommt die umgeleiteten Requests.

Tabelle 7.14.: Fish Tank Suite: Testfall 2

FTS-T03 Campaign Update

ID/Bezeichnung	FTS-T03 - Campaign Update
Beschreibung	Vorhandene Campaign wird angepasst und an alle nötigen Systeme verteilt.
Testvoraussetzung	Fish Tank Suite vollständig aufgesetzt, alle Systeme aktiv und Push Rechte für das Campaigns Git Repository auf dem Piranhafish
Testschritte	1. Vorhandenes Campaign JSON anpassen 2. Änderungen an Piranha pushen 3. Auf Triggerfish prüfen, ob Änderungen übernommen wurden 4. Auf Elasticsearch Host prüfen, ob Campaign Änderungen übernommen wurden
Erwartetes Ergebnis	Neuste Version des Logstash Filters ist auf dem Elasticsearch Host installiert.

Tabelle 7.16.: Fish Tank Suite: Testfall 3

7.5.2. Testprotokoll

FTS-T01 Keyword Strategy

Zum Testen wurde folgende Konfiguration verwendet, aus Vertraulichkeitsgründen wurden jedoch die Keywords, IP's, keys und der prefix anonymisiert.

```
{
  "name": "C1",
  "redirectIp": "FakeCC IP",
  "portRedirects": [
    {
      "originalPort": "80",
      "redirectPort": "3000"
    }
  ],
  "SearchStrategies": [
    {
      "name": "payload",
      "prefix" : "prefix",
      "type": "KeywordStrategy",
      "keywords": [
        "keyword",
        "keyword"
      ]
    }
  ],
  "encryption" : {
   "xor" : ["key"],
   "aes" : ["key"]
  }
}
```

Auflistung 27: Fish Tank Suite: Keyword Search Strategy Config Beispiel

Die Malware wurde gestartet und sendet verschlüsselte Requests über den System Proxy an den Original C&C. Auf Kibana werden die Request mit Tags (Campaign Name und Cipher) markiert und können so als entschlüsselte Requests erkannt werden.

method	tags
-	
POST	C1, XOR
-	
POST	C1, XOR

Abbildung 7.10.: Fish Tank Suite: Kibana entschlüsselte Requests

Auf dem Mandarinfish muss nun also nach relativ kurzer Zeit eine Iptable gesetzt worden sein, welche die Original C&C IP und Destination Port auf die Fake C&C IP und deren Destination Port umleitet.

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DNAT       tcp  --  anywhere             [OriginalCC IP]  tcp dpt:http to:[FakeCC IP]:3000

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
```

Auflistung 28: Fish Tank Suite: Mandarin Iptables

Ebenfalls sollten nun auf dem Fake C&C die umgeleiteten Requests in seinem Log auftauchen. Die Malware sendet weiter Requests ohne zu merken, dass die Requests über den Fake C&C umgeleitet werden. Im Access Log können die gesendeten Responses des Original C&C (die nicht modifizierten) und die Paths der Requests, die vom Client gesendet wurde, eingesehen werden.

```
[2016-12-11 08:03:27.457] [INFO] fakecc - original CC Response to Client
[2016-12-11 08:03:37.678] [INFO] fakecc - path
[2016-12-11 08:03:37.769] [INFO] fakecc - original CC Response to Client
[2016-12-11 08:03:47.704] [INFO] fakecc - path
[2016-12-11 08:03:47.790] [INFO] fakecc - original CC Response to Client
```

Auflistung 29: Fish Tank Suite: Fake C&C Access log

FTS-T02 Uri Sequence Strategy

Nachfolgende Konfiguration wurde zur Erkennung einer selbst entwickelten Malware verwendet. Diese Malware unterstützt HTTPS und wird auf einen eigenen Fake C&C umgeleitet.

```
{
  "name": "C2",
  "redirectIp": "fakeCC-IP",
  "portRedirects": [
    {
      "originalPort": "443",
      "redirectPort": "4000"
    }
  ],
  "SearchStrategies": [
    {
      "name": "register",
      "type": "UriSequenceStrategy",
      "sequence": {
        "entry": {"uri": "register"},
        "next": {"uri": "count", "timeSpan": "30s"}
      }
    }
  ]
}
```

Auflistung 30: Fish Tank Suite: URI Sequence Strategy Config Beispiel

Die Fake Malware wurde gestartet und sendet nach einem ersten Register alle 30 Sekunden einen weiteren Request. Der Intervall und die Ziel-Uris *register* und *count* sollen nun durch den Triggerfish erkannt werden.
Sobald sie erkannt werden, wird eine Iptable gesetzt, die den HTTPS Verkehr auf den Port 4000 des Fake C&C Hosts umleitet.

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DNAT       tcp  --  anywhere             [OriginalCC IP]  tcp dpt:https to:[FakeCC
 ↪  IP]:4000

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
```

Auflistung 31: Fish Tank Suite: Mandarin Iptables

Sobald die Iptable gesetzt ist werden alle Repsonses, die nicht an die URI *register* gehen, durch den Text "This is not from the Original Server" ersetzt. Dieser Text wird durch den Fake C&C gesetzt.

```
send: Mjc2NzI1ZDItYzZlMy00ZjQ1LWEzY2ItOGM1NzBhNmZhNzk5
path: /v1/register
received: ODc0ZjdmZDYtNjBjOC00ODY0LWIyMWQtZmY5YjdiMTc0MmVk
send: MmQ1NzkxYzUtZmU2ZS00ZGFkLTlmOTUtZTkwMzI4MWQ5NzZh
path: /v1/count
received: YWMxZDY1ZWYtMWJlNC00N2M3LTk0YWEtN2I5Yjc3NzgwNjgx
send: NTdhM2UwZmMtZWRkZC00MzZkLTlhNzMtMzA0Y2Y2OTI1MGQz
path: /v1/count
received: This is not from the Original Server
send: NDE2YWM0ODItYzBmYS00OTg1LThkZDMtMzE4YjkwZmI5ZmRi
```

Auflistung 32: Fish Tank Suite: Client Log

FTS-T03 Campaign Update

Das Ändern einer vorhandenen Campaign kann wichtig werden, wenn neue Erkenntnisse zu einer Malware bestehen (z.B. neuer Key oder neues Keyword, nach dem geprüft werden soll). Zusätzlich werden die neusten Änderungen vom Logstash Decrypt Filter und vom Triggerfish übernommen.
Falls der Decrypt Filter sich geändert hat und eine neue Release Version verfügbar ist (falls die installierte Version auf Elasticsearch gleich der Version vom Logstash Decrypt Filter Repository ist, wird der Filter nicht installiert), muss die Versionsnummer unter *s.version* in der *Gemspec* Datei angepasst werden. Die Änderungen am Logstash Filter Decrypt müssen nun auf den Piranha Host publiziert werden, damit das Campaigns Repository über einen Git Hook die neusten Änderungen holen kann.

```
Gem::Specification.new do |s|
  s.name = 'logstash-filter-decrypt'
  s.version = '0.0.2'
```

Auflistung 33: Fish Tank Suite: Logstash Filter Decrypt Version

Sobald ein neuer Push auf das Campaigns Repository stattfindet, wird der Hook ausgeführt und die neue Version des Filters installiert.

```
git commit --allow-empty -m "Trigger Hook"
```

Auflistung 34: Fish Tank Suite: Leerer Commit für Campaigns Repository

Beim nächsten push des Campaigns Repository werden alle Hooks ausgeführt und die neuere Version des Decrypt Filter auf Elasticsearch installiert.

```
Counting objects: 1, done.
Writing objects: 100% (1/1), 190 bytes | 0 bytes/s, done.
Total 1 (delta 0), reused 0 (delta 0)
remote: remote: From /home/ccproxy/Git/campaigns
remote: remote:  * branch            master     -> FETCH_HEAD
remote: remote:    74e77d4..e0831a9  master     -> origin/master
remote: remote: Updating 74e77d4..e0831a9
remote: remote: Fast-forward
remote: remote: From piranha:ccproxy/triggerfish-agent
remote: remote:  * branch            master     -> FETCH_HEAD
remote: remote: Already up-to-date.
remote: To ccproxy@trigger:Git/campaigns.git
remote:    74e77d4..e0831a9  master -> master
remote:    532bd58..74e77d4  elasticsearch/master -> elasticsearch/master
remote:    532bd58..74e77d4  triggerfish/master -> triggerfish/master
remote: remote: From /home/ccproxy/Git/campaigns
remote: remote:  * branch            master     -> FETCH_HEAD
remote: remote:    74e77d4..e0831a9  master     -> origin/master
remote: remote: Updating 74e77d4..e0831a9
remote: remote: Fast-forward
remote: remote: From prianha:ccproxy/logstash-filter-decrypt
remote: remote:  * branch            master     -> FETCH_HEAD
remote: remote: Already up-to-date.
remote: remote:   Successfully built RubyGem
remote: remote:   Name: logstash-filter-decrypt
remote: remote:   Version: 0.0.2
remote: remote:   File: logstash-filter-decrypt-0.0.2.gem
remote: remote: Validating /home/ccproxy/campaigns/
logstash-filter-decrypt/logstash-filter-decrypt-0.0.2.gem
remote: remote: Installing logstash-filter-decrypt
remote: remote: Installation successful
remote: To elasticsearch:Git/campaigns.git
remote:    74e77d4..e0831a9  master -> master
remote:    532bd58..74e77d4  elasticsearch/master -> elasticsearch/master
remote:    74e77d4..e0831a9  triggerfish/master -> triggerfish/master
To git@piranha.hacking-lab.com:ccproxy/campaigns.git
   74e77d4..e0831a9  master -> master
```

Auflistung 35: Fish Tank Suite: Git Push Log

Auf dem Elasticsearch Host ist nun die neuste Version des Logstash Filters installiert. Beim Triggerfish ist die Versionierung kein Problem, da immer alle Dateien überschrieben werden.

```
sudo /usr/share/logstash/bin/logstash-plugin list --verbose logstash-filter-decrypt
 ↪  logstash-filter-decrypt (0.0.2)
```

Auflistung 36: Fish Tank Suite: Logstash Filter Version

7.5.3. Performance

Um einen Einblick zu bekommen, wie performant die Fish Tank Suite Lösung ist, wurden HTTP Performance Tests durchgeführt, welche 150 Benutzer simulieren. Diese Benutzer greifen dabei auf eine Auswahl der Top 25 Wikipedia Artikel zu. Diese Requests sollen dabei über den Proxy geleitet werden. Zusätzlich zu den 150 Benutzern wird eine Malware gestartet (Keyword Strategy). Für die nachfolgenden Performance Tests wurde locust[16][2] verwendet
Bei der Simulation sind dabei folgende Indikatoren wichtig:

Zeit bis Message aus der Queue Falls nun die Queue mit Request überhäuft wird und dazwischen einzelne Requests durch Logstash entschlüsselt werden sollen, kann die Abarbeitung der Queue länger dauern als geplant.

Zeit bis zur Erkennung der Malware Sobald die Message in der Elasticsearch gespeichert ist, braucht der Triggerfish einen Moment, bis er die zutreffenden Messages findet und eine Iptable auf dem Mandarinfish setzt.

Zeit von Erkennung bis Abwehr Dies beinhaltet die beiden oben genannten Indikatoren und soll einen möglichst realitätsnahen Wert darstellen, wie lange es dauert, eine Malware zu entdecken und Abwehrmassnahmen einzuleiten.

Logging

Der Pufferfish loggt alles direkt in die RabbitMQ, muss allerdings von allen Requests den Body lesen, was ein blockender Aufruf ist. Somit wird der Pufferfish auch zum Flaschenhals der Software. Wenn nicht geloggt wird, kann auch nicht entschlüsselt werden.
HTTP Performance Vergleich zwischen ohne Proxy und mit Proxy:

Type	Name	# requests	# fails	Median	Average	Min	Max	Content Size	# reqs/sec
GET	/wiki/Che_Guevara	210	0	240	246	133	426	538570	2.5
GET	/wiki/Donald_Trump	226	0	380	387	204	616	1215777	1.8
GET	/wiki/Elizabeth_II	238	0	190	196	134	320	484860	2.7
GET	/wiki/Fantastic_Beasts_and_Where_to_Find_Them_(film)	231	0	120	130	64	395	255262	2.9
GET	/wiki/Fidel_Castro	234	0	250	257	151	425	649136	2.7
GET	/wiki/Final_Fantasy_XV	212	0	120	121	60	398	207046	1.8
GET	/wiki/Shigeru_Miyamoto	225	0	140	143	86	385	271748	2.4
GET	/wiki/United_States_presidential_election,_2016	208	0	310	320	146	512	1131552	2.9
GET	/wiki/Westworld_(TV_series)	232	0	130	139	91	424	268090	1.8
	Total	2016	0	190	214	60	616	553929	21.5

Abbildung 7.11.: Fish Tank Suite: HTTP Performance ohne Proxy

[2] http://locust.io/

Type	Name	# requests	# fails	Median	Average	Min	Max	Content Size	# reqs/sec
GET	/wiki/Che_Guevara	206	0	31000	41000	6464	309402	538570	0.1
GET	/wiki/Donald_Trump	185	0	61000	78066	16960	349124	1215777	0.8
GET	/wiki/Elizabeth_II	228	0	20000	31294	3670	516172	484860	0.4
GET	/wiki/Fantastic_Beasts_and_Where_to_Find_Them_(film)	248	0	10000	17739	1037	257482	255262	0.5
GET	/wiki/Fidel_Castro	213	0	35000	46049	4890	290443	649136	0.4
GET	/wiki/Final_Fantasy_XV	222	0	8100	13842	866	125699	207046	0.3
GET	/wiki/Shigeru_Miyamoto	244	0	14000	21756	1769	270186	271748	0.3
GET	/wiki/United_States_presidential_election,_2016	214	0	45000	58233	12204	430021	1131115	0.1
GET	/wiki/Westworld_(TV_series)	248	0	12000	17728	1443	150534	268090	0.2
	Total	2008	0	20000	34597	866	516172	532270	3.1

Abbildung 7.12.: Fish Tank Suite: HTTP Performance mit Proxy

Es gibt eine Performance Verschlechterung, wenn der ganze Traffic über den Proxy geht. Besserung könnte hier womöglich noch ein Ausklammern der "Known Goods" bringen, damit nicht alles geloggt wird, sondern nur die Requests, die interessant sein könnten.

Entschlüsseln

Entschlüsselt wird auf dem Elasticsearch Host, wo der Logstash Decrypt Filter die Messages aus der RabbitMQ entgegennimmt. Hier kann es zu Zeitverzögerungen kommen, falls eine Message aus der Queue länger benötigt und dadurch die nachfolgenden Messages stoppt. Logstash kann allerdings gut skaliert werden, wodurch die Performance verbessert werden kann.
Performance Logstash Für diesen Test wurde zusätzlich zu den 150 Benutzern eine Malware gestartet, um die Performance des Entschlüsselns mit viel nebenläufigem Verkehr beobachten zu können.
Durch den nebenläufigen Verkehr kann es zu Lastspitzen kommen, aber wie man beim Logging bereits gesehen hat, können im Durchschnitt ca. 3 Requests pro Sekunde durch den Proxy abgearbeitet werden, wodurch keine zu grosse Lastspitzen entstehen.

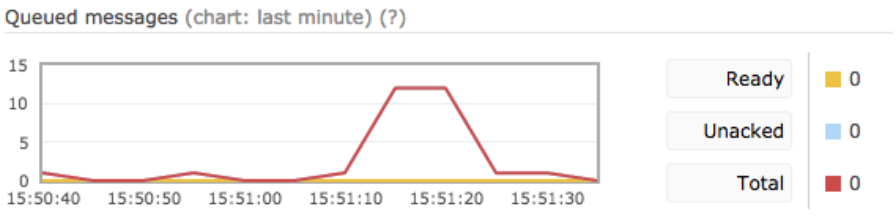


Abbildung 7.13.: Fish Tank Suite: RabbitMQ Lastspitzen

Das Entschlüsseln selbst wird daher nicht allzu sehr verzögert (Event Timestamp ist der Zeitpunkt, wo die Message an die Queue gesendet wurde). Die Entschlüsselung benötigt bloss einige Millisekunden, aber der Event kam 1 Sekunde verspätet an die Reihe.

```
[2016-12-12T15:52:03,429][INFO ][logstash.filters.decrypt ] Decrypt Body
[2016-12-12T15:52:03,443][INFO ][logstash.filters.decrypt ] Event after filter
 ↪  {:event=>2016-12-12T14:52:02.942Z mandarin REQMOD}
```

Auflistung 37: Fish Tank Suite: Logstash Log Decrypt Beispiel

Ein Performance Engpass existiert bei der Entschlüsselung nicht, und die Möglichkeit, einen RabbitMQ Cluster und mehrere Logstash Consumers zu installieren, existiert immer.

Abwehr

Bei der Abwehr muss in möglichst kurzem Zeitraum eine Iptable gesetzt werden, um die Malware auf den Fake C&C Server umzuleiten.
Bei diesem Test wurde ebenfalls die Malware mit 150 Benutzern zusammen verwendet. Kurz nachdem die Malware gestartet wurde, wird auf dem Mandarinfish die Iptable gesetzt

```
Validating IP Addresses
IP: 212.254.246.109 is v4
IP: 192.168.200.125 is v4
Validating Ports
All Port OK
Checking if redirect IP is whitelisted
IP: 192.168.200.125is in subnet: 192.168.200.0/24
is Host Adress
Check if Iptable already exists
Iptable doesnt exist
```

Auflistung 38: Fish Tank Suite: Mandarinfish Iptable setzen

Die umgeleiteten Requests werden auch wieder über den Fake C&C Server geleitet.
Die Abwehr und Umleitung der Requests funktioniert auch mit 150 Benutzern, die ebenfalls den Proxy verwenden.

7.5.4. Zusammenfassung

Die Fish Tank Suite hält den Datenverkehr von 150 Benutzern und mehr aus, die Geschwindigkeit lässt jedoch merklich nach. Um die letzten Flaschenhälse zu entfernen, müsste der Pufferfish performanter werden, da aber der ganze Inhalt der Requests oder Responses gelesen werden muss, gibt es nicht viele Optimierungsmöglichkeiten.

8. Schlussfolgerung

In diesem Kapitel wird folgende Frage beantwortet: *"Was sind die Vor- und Nachteile der erarbeiteten Lösung?"*.

8.1. Resultat

Die Funktionen der Fish Tank Suite können durch die einzelnen Systemteile am einfachsten erklärt werden.

8.1.1. Piranhafish (Configuration Management)

Der Piranhafish Manager ermöglicht es, dem Security Engineer der Fish Tank Suite neue Campaigns hinzuzufügen. Über den Piranhafish wird auch neuer Programmcode für die Informationsanreicherung in der Datenbank oder neue Suchstrategien verteilt.

8.1.2. Pufferfish (Logging)

Der Pufferfish Logger sorgt für eine reibungslose Datenübertragung vom Proxy zur Datenbank. Er wird durch eine RabbitMQ Queue unterstützt, um bei Neustarts oder Lastspitzen das System stabil zu halten.

8.1.3. Triggerfish (Search Strategies)

Der Triggerfish Agent sucht auf einer Elastcisearch Datenbank nach Malware Aktivitäten. Er unterstützt diverse Suchstrategien. Es kann zum Beispiel nach Schlüsselwörtern im Body der Pakete gesucht werden. Es ist auch möglich nach zeitlichen Abfolgen zu suchen.

8.1.4. Mandarinfish (Redirect)

Der Mandarinfish Router kann einzelne Zieladressen auf andere Adressen umleiten. Zudem ist es auch möglich, einzelne Ports umzuleiten. Um Missbrauch zu vermeiden, kann durch eine Subnet Whitelist verhindert werden, dass Umleitungen auf Adressen im Internet gemacht werden.

8.1.5. Fake Command & Control (C&C) Server

Der Fake C&C Server nimmt die Anfragen der Malware entgegen (Falls umgeleitet) und entscheidet, ob die Anfrage weiter an den Original C&C Server gesendet wird. Die Antwort des Original C&C Servers kann so verändert werden, dass die Malware weiterschläft.

8.2. Bewertung

Die Fish Tank Suite erfüllt alle Anforderungen, die während dem Projekt erhoben wurden. Es ist möglich, eine Malware stillzulegen, ohne dass der Angreifer das sofort bemerkt. Wie schnell der Angreifer dahinterkommt und seine Strategie wechselt, müsste in einem realen Szenario getestet werden. Wenn man aber bedenkt, wie gross die zeitlichen Abstände sind, in denen die Malware Nachrichten an den Original C&C Server sendet, dann kann man davon ausgehen, dass das Eingreifen in den Angriff einige Monate unbemerkt bleibt. In dieser Zeit muss etwas gegen den Angriff unternommen werden, denn die Fish Tank Suite ist nicht dazu vorgesehen, den Angriff komplett zu unterbinden.

Während dem Projekt war oft die Rede von Praxistauglichkeit. Dieses Ziel wurde anfangs berücksichtigt und hat die Architektur stark beeinflusst. Im späteren Verlauf rückte diese Anforderung in den Hintergrund, da sonst andere Anforderungen weniger zur Geltung gekommen wären. Es müssen noch einige Aspekte, wie zum Beispiel Security, beachtet werden, um eine praxistaugliche Software zu erhalten. Trotzdem läuft die Software relativ stabil und erfüllt ihre Aufgabe zuverlässig.

8.3. Ausblick

Wenn die übrigen Punkte (Praxistauglichkeit) noch erfüllt werden und ein Test in einem echten Szenario stattfindet, dann ist die Fish Tank Suite eine gute Lösung für die Problemstellung. Die Software könnte als Open Source Software weiterentwickelt werden. Es könnte noch ein öffentliches Repository für Suchstrategien eingerichtet werden, so dass jeder seine Suchstrategien teilen kann. Damit die Software vollständig durch Kampagnen konfiguriert werden kann, muss die Konfigurationsdatei mit weiteren Informationen versehen werden, damit auch der Fake C&C Server automatisch erstellt wird. Wenn diese Punkte erfüllt sind, dann ist die Fish Tank Suite eine Lösung, die man in Erwägung ziehen sollte.

Literatur

[1] Atlassian. *JIRA Documentation*. [Online; Zugegriffen 22-September-2016]. 2016. URL: https://confluence.atlassian.com/jira/jira-documentation-1556.html.

[2] Elastic. *Centralize, Transform & Stash Your Data*. [Online; Zugegriffen 28-October-2016]. 2016. URL: https://www.elastic.co/products/logstash.

[3] Elastic. *Packetbeat: Transaction Protocols Configuration*. [Online; Zugegriffen 30-October-2016]. 2016. URL: https://www.elastic.co/guide/en/beats/packetbeat/current/configuration-protocols.html.

[4] Elastic. *rabbitmq*. [Online; Zugegriffen 28-October-2016]. 2016. URL: https://www.elastic.co/guide/en/logstash/current/plugins-inputs-rabbitmq.html.

[5] Elastic. *Your Window into the Elastic Stack*. [Online; Zugegriffen 28-October-2016]. 2016. URL: https://www.elastic.co/products/kibana.

[6] FiddlerCore. »FiddlerCore Documentation«. In: (2016).

[7] FireEye. *Anatomy of Advanced Persistent Threats*. [Online; Zugegriffen 24-October-2016]. 2016. URL: https://www.fireeye.com/current-threats/anatomy-of-a-cyber-attack.html.

[8] GitLab. *GitLab CI Documentation*. [Online; Zugegriffen 24-September-2016]. 2016. URL: https://docs.gitlab.com/ee/ci/README.html.

[9] Alan Hardy. *Elasticsearch - Key Features by Alan Hardy*. [Online; Zugegriffen 28-October-2016]. Elastic. 2016. URL: https://www.elastic.co/videos/elasticsearch-key-features-by-alan-hardy.

[10] A. Cerpa J. Elson. *Internet Content Adaptation Protocol (ICAP)*. [Online; Zugegriffen 25-October-2016]. 2003. URL: https://tools.ietf.org/html/rfc3507.

[11] Jeff Sutherland Ken Schwaber. »Der Scrum Guide - Der gültige Leitfaden für Scrum«. In: (2013).

[12] Thomas Knellwolf. *Cyber-Attacke auf die Ruag: Bundesrat setzt Taskforce ein*. [Online; Zugegriffen 18-November-2016]. 2016. URL: http://www.tagesanzeiger.ch/schweiz/standard/CyberAttacke-auf-die-Ruag-Bundesrat-setzt-Taskforce-ein/story/29638139.

[13] Craig Larman. *UML 2 und Patterns angewendet Objektorientierter Softwareentwicklung*. mitp, 2005. ISBN: 978-3-8266-1453-8.

[14] Eric Lawrence. *Debugging with Fiddler*. [Online; Zugegriffen 30-September-2016]. 2016. URL: https://fiddlerbook.com/.

[15] linux_joy. *tcpdump*. [Online; Zugegriffen 28-October-2016]. 2016. URL: https://wiki.ubuntuusers.de/tcpdump/.

[16] Locust. *Locust Documentation*. [Online; Zugegriffen 5-Dezember-2016]. 2016. URL: http://docs.locust.io/en/latest/.

[17] Microsoft. *Microsoft Developer Network*. [Online; Zugegriffen 01-Oktober-2016]. 2016. URL: https://msdn.microsoft.com.

[18] S. Lawrence R. Khare. *Upgrading to TLS Within HTTP/1.1*. [Online; Zugegriffen 03-October-2016]. 2000. URL: https://tools.ietf.org/rfc/rfc2817.

[19] Eric Rescorla. *ssldump - Linux man page*. [Online; Zugegriffen 28-October-2016]. 2016. URL: https://linux.die.net/man/1/ssldump.

[20] Daniel Roethlisberger. *SSLsplit - transparent SSL/TLS interception*. [Online; Zugegriffen 02-October-2016]. 2016. URL: https://www.roe.ch/SSLsplit.

[21] Slack. *Incoming Webhooks*. [Online; Zugegriffen 24-September-2016]. 2016. URL: https://api.slack.com/incoming-webhooks.

[22] squid-cache. *Squid configuration directive logformat*. [Online; Zugegriffen 30-October-2016]. 2016. URL: http://www.squid-cache.org/Doc/config/logformat/.

[23] Testprotokoll. *Ergebnis: Testprotokoll*. [Online; Zugegriffen 30-November-2016]. 2016. URL: http://www.hermes.admin.ch/szenarien/szenario_50_Alles/scenario/de/ergebnis_testprotokoll.html.

[24] Ubuntu. *IptablesHowTo*. [Online; Zugegriffen 15-October-2016]. 2016. URL: https://help.ubuntu.com/community/IptablesHowTo.

[25] Wikidot.com. *Fiddler Wiki*. [Online; Zugegriffen 30-September-2016]. 2016. URL: http://fiddler.wikidot.com/.

[26] Wikipedia. *Advanced Persistent Threat*. [Online; Zugegriffen 24-October-2016]. 2016. URL: https://de.wikipedia.org/wiki/Advanced_Persistent_Threat.

[27] Wikipedia. *Internet Content Adaptation Protocol*. [Online; Zugegriffen 25-October-2016]. 2016. URL: https://de.wikipedia.org/wiki/Internet_Content_Adaptation_Protocol.

[28] Wikipedia. *iptables*. [Online; Zugegriffen 15-october-2016]. 2016. URL: https://de.wikipedia.org/wiki/Iptables.

[29] Wikipedia. *Proxy (Rechnernetz)*. [Online; Zugegriffen 02-October-2016]. 2016. URL: https://de.wikipedia.org/wiki/Proxy_(Rechnernetz).

Glossar und Abkürzungsverzeichnis

0-Day Bei einem 0-Day handelt es sich um einen Bug der seit Release einer Software oder nach einem Update vorhanden ist..

ACL Access Control List.

AES128 Beim Advanced Encryption Standard 128 handelt es sich um eine symmetrische Verschlüsselung mit einer Schlüssellänge von 128 Bit..

API Application Programming Interface.

APT Advanced Persistent Threat.

C&C Command & Control.

CN Unter dem Common Name wird ein Zertifikat ausgestellt (Hostname)..

EDA Eidgenössische Departement für auswärtige Angelegenheiten.

FTS Full Text Search.

HSR Hochschule für Technik Rapperswil.

ICAP Internet Content Adaptation Protocol.

REST Representational State Transfer.

SSL Secure Socket Layer.