Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Constantin Wahl

Erweiterung einer Honeypot-Software zur Erkennung von IoT-spezifischen Angriffen und zur Verwendung einer ELK-Umgebung

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Constantin Wahl

Erweiterung einer Honeypot-Software zur Erkennung von IoT-spezifischen Angriffen und zur Verwendung einer ELK-Umgebung

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Klaus-Peter Kossakowski
Zweitgutachter: Prof. Dr. Martin Becke

Eingereicht am: 23. April 2019

Constantin Wahl

Thema der Arbeit
Erweiterung einer Honeypot-Software zur Erkennung von IoT-spezifischen Angriffen und zur Verwendung einer ELK-Umgebung

Stichworte
IoT, Internet of Things, Honeypot, Telnet

Kurzzusammenfassung
In einer Welt in der immer mehr Alltagsgegenstände mit dem Internet verbunden sind, wächst die Anzahl der Angriffe mit Hilfe alter und unsicherer Protokolle im Einsatz mit der Masse an Alltagsgegenständen. Daher müssen Honeypots zur Erkennung und Analyse solcher Angriffe erweitert werden. Außerdem werden die gesammelten Daten des Honeypots in den ELK-Stack portiert und eine Analysemöglichkeit durch diesen Aufgezeigt.

Constantin Wahl

Title of the thesis
Extention of a Honeypot-Software for recognization of IoT-specific attacks and use in an ELK-environment

Keywords
IoT, Internet of Things, Honeypot, Telnet

Abstract
In a world where every day more and more things get access to the internet, old and therefor vulnerable protocols like Telnet are used for configuration. This is a thread because the protocolls can be and are more and more used for attacks against and with those devices. This leads to the addition of those old protocols to modern Honedypots. Also in this thesis the Honeypot-data is integrated into the Elastic-Stack for analyical purposes.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

1.1 Motivation

In der modernen Welt ist es immer wichtiger, mit dem Internet verbunden zu sein. Jeder möchte immer erreichbar und stets auf dem neusten Stand sein. Je mehr wir mit der Technologie voranschreiten, desto mehr Geräte sind mit dem Internet verbunden.

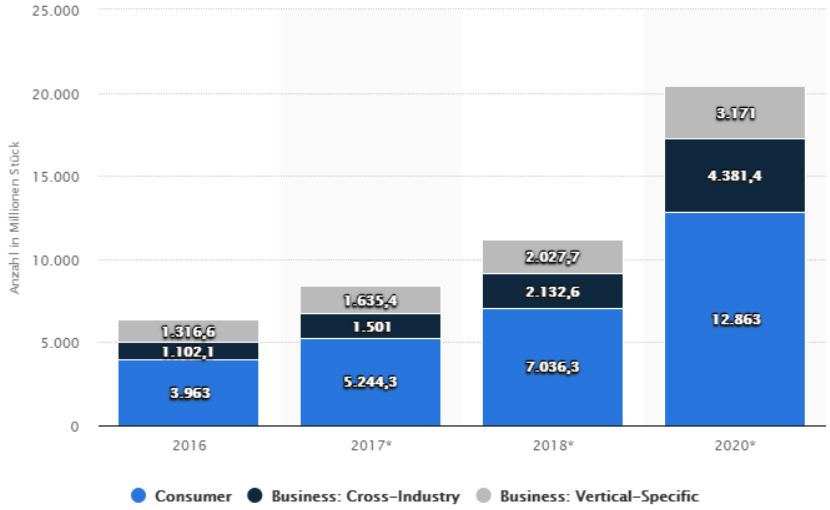

Abbildung 1.1: Entwicklung der Anzahl der geschätzten IoT-Geräte bis 2020 nach Sta16 (2016)

Abbildung 1.1 zeigt die Entwicklung der Menge an Alltagsgegenständen, die schon mit dem Internet verbunden sind. So waren es 2016 rund 5 Milliarden Geräte. Diese Anzahl soll sich Umfragen zufolge bis 2020 auf rund 20 Milliarden vervierfachen. Die Daten der Abbildung sind für jedes Jahr nach 2016 mit einem Stern gekennzeichnet,

da es sich nur um Prognosen handelt, welche für der Erstellung der Statistik gemacht wurden. Ein Grund dafür ist die Entwicklung des sogenannten Internet of Things. Dieser Begriff bezeichnet die Technologie, auch Alltagsgegenstände wie Kühlschränke, Fernseher etc. an das Internet anzuschließen. Leider wachsen mit Hinzunahme dieser Gegenstände auch die entsprechenden Möglichkeiten derjenigen, die die Technologie für kriminelle Machenschaften nutzen möchten.
Die neuen Clients im Internet of Things könnten hierzu missbraucht werden, weil es schwierig ist, diese komplett auf dem neuesten Stand der Technik zu behalten. Daher werden häufig noch alte Techniken, die zu anfällig für Angriffe von außen sind, um sie großflächig zu nutzen, zur Kommunikation verwendet. Ein Beispiel dafür war der größte bekannte Angriff auf die Verfügbarkeit eines Systems mit Hilfe eines sogenannten Bot-Netzwerkes, bei dem mit Hilfe vieler Clients eine Anfragelast von mehreren Terabits geschaffen und damit die Verfügbarkeit des Systems eingeschränkt wurde.
Deshalb soll es in dieser Arbeit darum gehen, wie diese Angriffe aussehen und wie man sie erkennen und aus ihnen Informationen über die Angreifer sammeln kann, um daraus ggf. später Abwehrstrategien zu entwickeln.

1.2 Zielsetzung

Das Ziel dieser Bachelorarbeit ist die Integration des SPOT-Honeypots in eine ELK-Umgebung. Hierzu gehört gegebenenfalls die Portierung der SPOT-Honeypot-Software und der Anschluss an den Elastic-Stack. Dies sollte so gestaltet sein, dass im Falle einer Änderung nur die Konfiguration eines Teils des Stacks angepasst werden muss.

Zusätzlich soll der SPOT-Honeypot erweitert werden, um gängige Angriffe in heute üblichen IoT-Einsatzumgebungen zu erkennen. Die zu erkennenden Angriffe schließen sogenannte laterale Bewegungen in einer Umgebung ein, die z.B. in einem Smart-Home durch das gemeinsam genutzte WLAN definiert wird, durch die infizierte Systeme neue Angriffsziele identifizieren.

1.3 Zielgruppe

Die in dieser Thesis behandelten Themen richten sich an Leser, welche ein gewisses Maß an Wissen über Netzwerke und die verschiedenen Methoden der möglichen Angriffe besitzt. Es werden Grundlagen nur auf Grund dieser Voraussetzung gegeben, da dies ansonsten den Rahmen der Arbeit übersteigen würde. Für Leser mit wenig Wissen in dem Bereich von Netzwerkschichten und deren Zusammenspiel wird empfohlen dieses zuvor zu erweitern. Allgemeines zu Protokollen und den behandelten im speziellen, sind Teil der Arbeit.

1.4 Aufbau der Arbeit

Kapitel 1, Einleitung

Kapitel 1 beschreibt Thematik und Aufbau der Arbeit. Außerdem wird eine Motivation geliefert und das Ziel der Thesis beleuchtet.

Kapitel 2, Grundlagen

Kapitel 2 liefert die notwendigen Grundlagen zum Thema. So wird eine Einführung zum Thema Internet of Things gegeben, kurz beleuchtet was Protokolle in diesem Zusammengang sind und Grundlagen zum Thema Honeypots geliefert. Außerdem werden Bedrohungen durch das Internet of Things betrachtet und eine Einführung in das Logfile Speicher- und Analysetool ELK-Stack gegeben.

Kapitel 3, Konzept der Erweiterung

Das Kapitel 3 erläutert das Konzept der Erweiterung des Honeypots und die dafür notwendigen Schritte. So wird das Vorgehen bei der Einbindung des ELK-Stacks beschrieben und einen Einführung in die umsetzbaren Teile des TELNET Protokolls gegeben.

Kapitel 4, Implementierung

Kapitel 4 beschreibt die Implementierung, der Erweiterung um die Einbindung des ELK-Stacks und des TELNET-Protokolls in das Honeypot-System. Außerdem werden Begründungen für die benutzte Technologie geliefert und die benutzten Frameworks beleuchtet.

Kapitel 5, Zusammenfassung und Ausblick

schließt das Thema ab, betrachtet eine Überprüfung der Funktionalität und zieht ein Fazit. Zusätzlich wird ein Ausblick zum Thema präsentiert, indem Anschlussmöglichkeiten für weitere Arbeiten aufgezeigt werden.

2 Grundlagen

Nachdem im letzten Kapitel die Motivation aufgezeigt wurde, werden im folgenden Kapitel die Grundlagen für die folgende Arbeit gegeben. Dabei soll zuerst auf die Grundlagen des Internet of Things und dann allgemein auf Honeypots und zum Schluss auf Grundlagen des ELK-Stack eingegangen werden.

2.1 Grundlagen Internet of Things

Der Begriff Internet of Things (IoT) wurde als erstes von Kevin Ashton im Jahre 1999 auf einer Konferenz benutzt und stellte dort eine Bezeichnung dafür dar, dass die Maschinen lernen müssten, die physische Welt eigenständig erfassen zu können. Dies müsse der Fall sein, da die Menschen, auf deren Informationen die Maschinen basieren, diese nicht in ausreichender Menge und Präzision sammeln können. Daher liegt es nahe, den Maschinen diese Möglichkeit zu geben und mit diesen Informationen die physische Welt zu optimieren (vgl. Ashton, 2009).
Dennoch ist dies nur das erste Vorkommen und es gibt bis heute immer noch keine einheitliche Definition. Das Institute of Electrical and Electronics Engineers, New York (IEEE), hat ein Dokument veröffentlicht, in dem ein Versuch gestartet wurde, eine anerkannte Definition zu liefern. Dort heißt es:
Die Entwicklung des IoT geht auf die Erfindung der Radio-Frequecy Identifacition (RFID) im zweiten Weltkrieg zurück. Dort wurde diese Technik für Radare genutzt. Später fand die Technologie Anwendung, um sensorisch den Hormonlevel von Kühen zu kontrollieren und noch heute nutzt man sie zur Zugangskontrolle für Türen u.Ä. (vgl. IEEE, 2015, S. 6ff).
Derselbe Artikel spricht über das Internet of Things als eine Architektur in drei Schichten. Demnach ist ein IoT-Gerät eine Hardware, die in erster Schicht, u.U. sensorisch,

Daten sammelt oder aus dem Internet erhält. Im nächsten Schritt werden diese Daten kommuniziert, die im letzten Schritt zur eigentlichen Anwendung gebracht werden. In Abbildung 2.1 (S.4) wird dieses Verfahren nochmals verdeutlicht.
Diese Arbeit beschäftigt sich hauptsächlich mit dem Austausch von Daten und der Konfiguration der IoT-Geräte. Dafür werden im Folgenden die Grundlagen geschaffen.

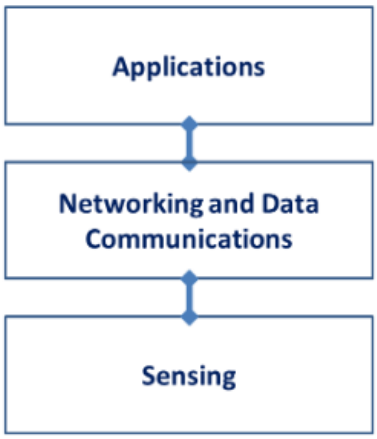

Abbildung 2.1: Aufbau der IoT Architektur nach Bild in IEEE (2015)

2.2 Protokolle

Bei einem Angriff durch IoT-Geräte kommen verschiedene Protokolle zur Verwendung. IoT-Geräte müssen administriert werden, um sie zu konfigurieren und zu steuern und den unterschiedlichen Umgebungen anzupassen. Hierfür kommen hauptsächlich Protokolle in den Einsatz, die eine Fernsteuerung ermöglichen, u.a. Secure Shell und TELNET. Dabei ist TELNET die ältere und unsicherere Möglichkeit, da es keine Verschlüsselung bietet. Weitere Charakteristika der Protokolle werden im Folgenden dargelegt. Laut einer Studie von IBM wurden 78 Prozent der Angriffe durch IoT-Geräte im Jahr 2015 schon mit TELNET durchgeführt. ein wesentlich kleinerer aber nicht numerisch belegter Anteil sollte demnach nur auf IoT-spezifische Protokolle zurückgreifen.(vgl. Craig, 2016) Dies ist der Grund, warum sich der Autor in dieser Arbeit mit dem TELNET-Protokoll beschäftigt.

2.2.1 TELNET

Wie im letzten Kapitel bereits erwähnt, soll es in dieser Arbeit hauptsächlich um die Kommunikation von IoT-Geräten gehen und darum, wie diese für Angriffe missbraucht werden können. Das Protokoll, das dafür häufig benutzt wird, ist das TELNET-Protokoll. Bei TELNET handelt es sich um ein Protokoll zur Übertragung von Daten mit Hilfe einer TCP Verbindung. Außerdem ist es möglich das virtuelle Terminal, das durch das Protokoll erzeugt wird, als Fernsteuerung zu nutzen (vgl. RFC854, 1983).
Ein weiterer Nutzen sind im Voraus vereinbarte Optionen. Außerdem wird am Anfang einer Session ein Mapping der eigenen Einstellungen des Terminals als NVT auf den Partner vorgenommen. Dadurch ist es weder für Server noch User notwendig, die Charakteristika des jeweils anderen zu speichern (vgl. RFC854, 1983).

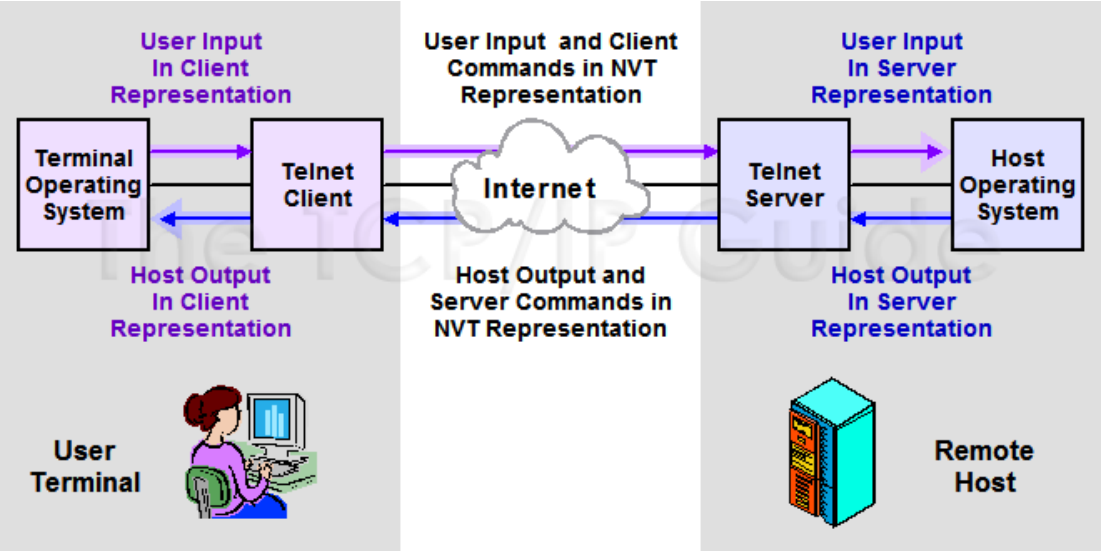

Abbildung 2.2: TELNET-NVT Architektur nach Kozierok (2005)

Abbildung 2.2 zeigt die typische Architektur der TELNET Kommunikation. Man sieht in dieser Architektur, dass die Daten nur weitergegeben und die entsprechenden Befehle nur als Daten gehandhabt werden, während es für die Steuerung des TELNET die gesonderten Befehle gibt, die am Anfang der Session ausgehandelt werden. Dies zeigt, dass sich mit dem TELNET Protokoll der Server Host fernsteuern lässt, da die

Daten, die von dem User Host gesendet wurden, interpretiert und die Ergebnisse als TELNET Datensatz zurück zum User Host gesendet werden.
Heutzutage wird dieses Protokoll, obwohl es relativ alt ist, noch immer viel genutzt. Dies hat erhebliche Nachteile für die Sicherheit der Systeme. Dies liegt daran, dass die Daten bei der Übertragung im Klartext geschickt werden. Es gibt also, im Gegensatz zu der sichereren Secure Shell Protokoll (SSH) Übertragung, keine Sicherheitsvorkehrungen. Da dies so ist, wird das Protokoll gerne und mit immer größerer Beliebtheit als Angriffsmethode genutzt. Daher wird die Erkennung und Analyse solcher Angriffe, mit Hilfe von Honeypots, immer bedeutsamer. Eine etwas modernere Variante zur Kommunikation von IoT-Geräten ist das Secure Shell Protokoll welches im Folgenden noch erklärt wird.

2.2.2 Secure Shell

Das Secure Shell Protokoll wird im RFC4253 als Transportprotokoll beschrieben, welches normalerweise über einen TCP/IP Stream eine remote Verbindung ähnlich der, die durch TELNET aufgebaut wird. Ein Unterschied besteht darin, dass es eine Verschlüsselung der Daten gibt, welche bei TELNET nicht gegeben ist. Daher ist das Protokoll weniger anfällig für bestimmte Angriffe. Außerdem bietet die Secure Shell auch immer eine Authentifizierung zwischen Clients und Servers (vgl. Lonvick und Ylonen, 2006). Das Protokoll bietet die Möglichkeit der Authentifikation über unterschiedliche Methoden. Zum einen ist eine Username/Passwort Authentifikation möglich. Zum anderen kann ein Public Key Verfahren genutzt werden. Letzteres wird deutlich empfohlen, da es eine höhere Sicherheit bietet (vgl. Lonvick und Ylonen, 2006). Im IoT wird hauptsächlich auf Public Key Verfahren gesetzt, auch weil eine sich immer wiederholenden Anmeldeprozedur über die Username/Passwort Authentifikation vermieden wird, die zu aufwändig wäre.
Die verwendeten Algorithmen für die Schlüssel müssen so sicher gewählt sein, dass sie den stärksten bekannten kryptografischen Attacken für Dekaden standhalten (vgl. Lonvick und Ylonen, 2006).

2.3 Grundlagen Honeypots

Unter Honeypots versteht man speziell konfigurierte Systeme, welche Angreifer anziehen und Schwachstellen darlegen, um das Verhalten der Angreifer beobachten zu können. Die Daten der Angreifer und ihres Verhaltens im angegriffenen System werden gesammelt und analysiert. Hieraus können Rückschlüsse über die Gefahr für den User oder seine Organisation gezogen werden. Der Betreiber des Honeypots erhält einen Eindruck davon, welche Systeme betroffen sind und wie die betroffenen Systeme aussehen und sich ggf. verändern. Außerdem kann gelernt werden, wie man diese Gefahren minimieren oder komplett beseitigen kann. Man unterscheidet dabei verschiedene Arten von Honeypots (vgl. Mairh u. a., 2011).

2.3.1 Low Interaction Honeypots

Bei low interactive Honeypots handelt es sich um einen einfach strukturierten Honeypot, auf dem typischerweise kein explizites Betriebssystem läuft, sondern nur einige ausgewählte Services, die ein echtes System simulieren. Da für den Angreifer nichts zu verändern ist, kann man mit dieser Art von Honeypot nicht besonders viele Informationen sammeln. Allerdings ist es möglich, neue Malware zu identifizieren und den Netzwerkverkehr mitzuschreiben. Etwas anders sieht dies schon bei den folgenden, etwas interaktiven Mid Interaction Honeypots aus, welche im Folgenden näher betrachtet werden.(vgl. Mairh u. a., 2011)

2.3.2 Mid Interaction Honeypots

Bei Mid Interaction Honeypots handelt es sich um ein ähnliches Konzept wie zuvor erwähnt. Auch diese Systeme agieren noch ohne eigenes Betriebssystem. Allerdings ist die Anzahl der möglichen Erkenntnisse stark erhöht, da mehr Interaktion im System möglich ist und diese mehrere verschiedene Angriffe zulassen und loggen können. (vgl. Mairh u. a., 2011). Allerdings sind Mid Interaction Honeypots deutlich komplexer in ihrer Konfiguration, weil verschiedene Sicherheitslücken kontrolliert und konfiguriert werden müssen.

2.3.3 High Interaction Honeypots

High Interactive Honeypots sind die fortgeschrittenste Variante der Honeypots. Anders als bei den anderen Ausführungen der Technologie besitzen diese ein eigenes Betriebssystem. Dies führt dazu, dass der Angreifer alle Möglichkeiten hat, die das Betriebssystem bietet. Dadurch ist diese Version der Honeypots besonders gefährlich für das umgebene System, da es sich um eine echte uneingeschränkte Maschine handelt. Dabei ist diese Art von System auch die aufschlussreichste für den Betreiber, da die Angreifer sich so verhalten können wie sie möchten, mit dem Unterschied, dass sie beobachtet werden können (vgl. Mairh u. a., 2011).

2.4 Bedrohungen durch IoT-Angriffe

IoT-Geräte können für Angriffe missbräuchlich genutzt werden, u.U. können sogar kompromittierte IoT-Geräte andere IoT-Geräte oder andere Systeme angreifen. Daher wird im Folgenden ein Überblick darüber gegeben, welche Bedrohung von solchen Angriffen ausgeht, beziehungsweise welche Angriffe damit zumeist umgesetzt werden.

2.4.1 Laterale Bewegungen

Laterale Bewegungen sind jene, bei denen sich der Angreifer von dem ursprünglich kompromittierten System im Netzwerk zu einer anderen Maschine bewegt, um an ein System zu gelangen, welches für ihn wertvolle Informationen beinhaltet. Dabei gibt es zwei verschiedene Möglichkeiten der Ausbreitung. Zum einen die Möglichkeit stringent von einem zum anderen System in die Tiefe zu gehen und somit zu verbreiten. Die zweite Möglichkeit ist die Ausbreitung von einem kompromittierten System auf möglichst viele verschiedene von einem Punkt aus. Laterale Bewegungen können erkannt werden, indem Login-Versuche aufgezeichnet werden, welche aus dem Netzwerk stammen, in dem sich das System befindet, auf das der Logon-Versuch ausgeführt wird (vgl. Heard Nicholas, 2016).

2.4.2 Amplification Denial of Service

Bei Amplification Denial of Service handelt es sich um einen Angriff, der zwei Probleme in Netzwerkprotokollen ausnutzt. Der erste ist, dass der IP-Header nicht authentifiziert ist, und das zweite besteht darin, dass bei UDP kein Handshake erfolgt. Ersteres sorgt dafür, dass Anfragen für Datenübertragungen gefälscht werden können. Letzteres ist für den DoS Angriff verantwortlich, da durch diese Sicherheitslücke die Angriffe direkt beantwortet werden (vgl. Krämer u. a., 2015).

Die Angreifer können diese Probleme auf zwei Weisen nutzen. Zum einen ist der Angreifer durch den Service, der als Schwachstelle genutzt wird, versteckt und somit schwieriger herauszufinden. Zum anderen kann durch kleine Anfragen eine große Antwort provoziert und zum Opfer geschickt werden. Dadurch kann teilweise, je nach Protokoll, eine Antwortgröße von bis zum 100-fachen der Anfragegröße erreicht werden (vgl. Krämer u. a., 2015). Durch die Vielzahl der ggf. betroffenen IoT-Geräte kann hierbei ein erhebliches Datenvolumen entstehen, das zu Störungen führt, weil diese hohen Datenmengen und/ oder reguläre Anfragen nicht mehr verarbeitet werden können.

2.5 Grundlagen ELK-Stack

Der ELK-Stack ist eine Software zum Ablegen und Sortieren sowie der Analyse von Dokumenten verschiedener Formate in normalisierter Form (vgl. search19, 2019). Im Folgenden werden die einzelnen Teile des Projektes erklärt.
In Abbildung 2.3 ist der ELK-Stack und die Zusammenarbeit der verschiedenen Teile illustriert, die im Folgenden weiter erklärt werden. In dieser Grafik ist allerdings neben Logstash auch Beats zu sehen, welches eine leichtere Alternative zu Logstash darstellt. In dieser Arbeit wird dies allerdings nicht weiter betrachtet, da es nicht Teil der betrachteten ELK-Stack-Integration ist.

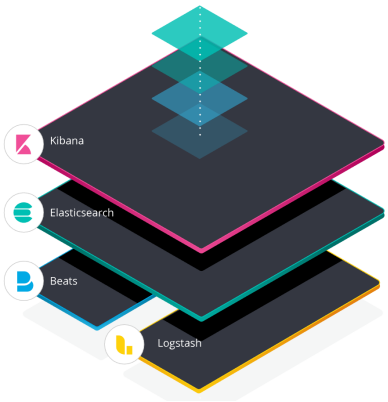

Abbildung 2.3: Zusammenspiel der einzelnen Teile des Elastic-Stacks nach Grafik auf Elastic19 (2019)

2.5.1 Elasticsearch

Elasticsearch ist die Hauptkomponente des ELK-Stacks und fungiert als Datenbank für die zu speichernden Dokumente Elastic19 (2019). Dabei werden die entsprechenden Dokumente per REST-API mit Indexierung, zur besseren Verwaltung der Dokumente, bei großer Menge versehen und im JSON Format abgelegt.
Durch die Indexierung kommt es dazu, dass jede Query in einer sehr kurzen Zeit bearbeitet werden kann. Daher ist es möglich, sehr große Datenmengen zu verarbeiten, ohne eine Einbuße an Geschwindigkeit zu haben. Queries zum Abrufen der Dokumente können mit verschiedenen Programmiersprachen genutzt werden. In Abbildung 2.4 ist ein allgemeiner HTTP-GET an Elasticsearch zu sehen, mit dem ein Dokument, welches zuvor dort gespeichert wurde, abgerufen wird.

```
curl -H "Content-Type: application/json" -XGET
'http://localhost:9200/social-*/_search' -d '{
  "query": {
    "match": {
      "message": "myProduct"
    }
  },
  "aggregations": {
    "top_10_states": {
      "terms": {
        "field": "state",
        "size": 10
      }
    }
  }
}'
```

Abbildung 2.4: Beispiel einer Curl HTTP-Get Abfrage für eine generische Suche in Elasticsearch siehe search19 (2019)

2.5.2 Logstash

Logstash wird als Pipeline im ELK-Stack genutzt, um Daten verschiedener Art zu verarbeiten und diese danach in Elasticsearch oder einem anderen Datenspeicher zu speichern search19 (2019). Dabei können die Daten verschiedene Ursprünge haben und auch komplett unterschiedlich in Größe und Aussehen sein. Außerdem werden die Daten von Logstash einem Filterprozess unterzogen, in dem wichtige Daten in ähnliche Formate gebracht werden. So können verschiedene vordefinierte Felder wie zum Beispiel Namen, Datum oder IP-Adressen erkannt werden während sie in den unterschiedlichsten Formaten vorliegen. Einmal zugeordnet, werden sie standardisiert und in ein normalisiertes Format transformiert. Ist der Filterprozess abgeschlossen werden die Daten im neuen Format an die Speicherressource weitergeleitet.

2.5.3 Kibana

Kibana ist die visuelle Komponente des ELK-Stack. Mit Kibana wird es möglich, die in Elasticsearch gespeicherten Dokumente zu analysieren und zu visualisieren, z.B. woher und zu welcher Zeit manche Phänomene auftreten (vgl. search19, 2019). Außerdem gibt es verschiedene Visualisierungsmöglichkeiten für sämtlichen Daten, so dass unterschiedliche Auswertungen möglich sind. Kibana bietet Histogramme, Pie Charts, Liniendiagramme und viele andere, inklusive der Möglichkeit per Grammatik eigene Visualisierungen zu erschaffen. Es ist möglich, geografische Daten direkt in einer Karte anzeigen zu lassen (siehe Abb. 2.5).

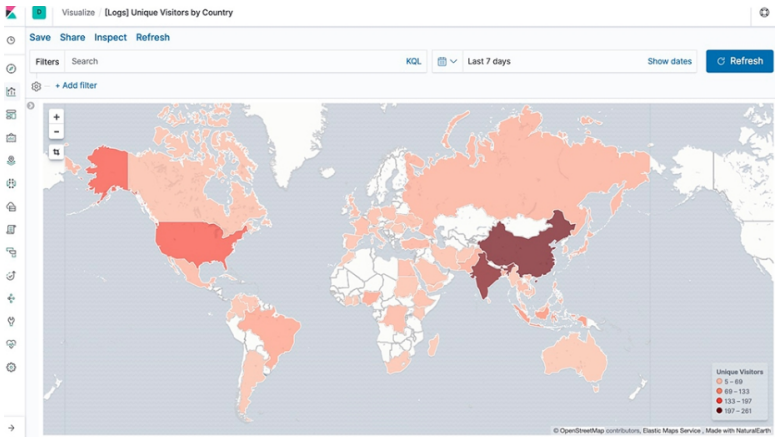

Abbildung 2.5: Beispiel einer geografischen Visualisierung nach search19 (2019)

3 Konzept der Erweiterung

Wie schon zuvor erläutert geht es in dieser Thesis um die Erweiterung einer Honeypot Software zur Erfassung von Angriffen mit IoT-Geräten. Dazu muss konzeptionell im Folgenden belegt werden, was einen solchen Angriff ausmacht und wie diese zu erkennen sind. Um die Veränderungen zu erklären, werden im folgenden Abschnitt zunächst die vorhandenen Funktionen der zu erweiternden Honeypot-Software und anschließend die im Rahmen dieser Arbeit hinzugefügten Funktionen erläutert.

3.1 Bisherige Funktionen der Software

Der Honeypot stellt verschiedene Services zur Verfügung, welche nach erfolgreicher Authentifizierung ein fiktives Dateisystem bieten, da es kein echtes Betriebssystem gibt. Es handelt sich um einen Software-Honeypot und dementsprechend ist es nicht im Sinne der Anwendung, das echte Dateisystem des Servers, auf dem der Honeypot läuft, dem Angreifer zu zeigen.
Die bereitgestellten Protokolle sind SMTP, IMAP, POP3, FTP, HTTP-Proxy und SSH.

Es kann das fiktive Dateisystem per XML-Datei konfiguriert werden. Wichtig dabei ist, dass das fiktive Dateisystem nur einen Baum der Tiefe 3 bereitstellt.
Außerdem können die Logs bereits aufbereitet und exportiert werden. Diese Funktion wird allerdings im Laufe dieser Arbeit dahingehend verändert, dass durch den ELK-Stack die komplette Verwaltung und Aggregation der Daten automatisiert durchgeführt wird.
Außerdem kann der Honeypot verschiedene IP-Adressen zur gleichen Zeit bereitstellen, was dazu führt, dass eine größere Menge an Daten gesammelt werden kann.

Protokoll	File System	Login
SMTP	nein	ja
IMAP	nein	ja
POP3	nein	ja
FTP	ja	ja
HTTP-Proxy	nein	ja
SSH	nein	ja

Abbildung 3.1: Protokoll Funktionen in Honeypot

Abbildung 3.1 zeigt welche Protokolle das gegebene Filesystem nutzten und welche Protokolle einen Login verlangen und diesen protokollieren.
In Abbildung 3.2 werden mögliche Angriffe, die von dem Honeypot im Ursprünglichen Zustand erkannt werden können.

Protokoll	Port Scan	Passwort raten
SMTP	ja	nein
IMAP	ja	ja
POP3	ja	ja
FTP	ja	ja
HTTP-Proxy	ja	ja
SSH	ja	nein

Abbildung 3.2: Durch den Honeypot erkennbare Angriffe

3.2 IoT-Angriffe

IoT-Geräte können für verschiedene Angriffe genutzt werden, wobei die IoT-Geräte vom Angreifer übernommen und für andere Angriffe in Form von Bot-Netzen weiterverwendet werden. Das Problem an Bot-Netzen besteht für den Angreifer darin, dass ein Gerät, das Teil des Netzes ist, u.U. nur einmalig genutzt werden kann, weil die genutzte IP-Adresse danach bekannt ist und die Geräte somit von der Kompromittierung durch den Angreifer befreit werden könnten. In der Realität ist dies häufig allerdings nicht durchführbar, weil die Masse und die Verteilung der IP-Adressen logistisch nur

schwer zu bewältigen ist.
Mit solchen Netzen ist es dann möglich verschiedene Angriffe auszuführen. So werden zum Beispiel Angriffe möglich, bei denen die Verfügbarkeit der Systeme in Mitleidenschaft gezogen wird. Diese sogenannten Denial of Service Angriffe sind solche, bei denen so viel Last auf das Netzwerk des Opfers gegeben wird, dass es nicht mehr möglich ist, reguläre Anfragen zu bearbeiten.
Im Folgenden soll es darum gehen, wie Bots erzeugt und letztlich gesteuert werden. Die Steuerung von Bots geschieht meist durch bestimmte Protokolle, die zur Fernsteuerung und zum Versenden von Daten konzipiert sind. TELNET wird zur Konfiguration von Geräten eingesetzt und stellt damit ein Angriffsfenster dar. Hierzu gehört neben TELNET auch Secure Shell (SSH), wobei es sich bei SSH um eine modernere Software handelt, da es unter anderem seine Daten verschlüsselt versendet. Einzelheiten sind im Kapitel 2, Grundlagen nachzulesen. Außerdem gibt es noch weitere speziell für IoT-Geräte konzipierte Protokolle, die allerdings nicht Gegenstand dieser Arbeit sind, wie z.B. Zigbee
Diese Arbeit beschäftigt sich jedoch hauptsächlich mit dem TELNET-Protokoll und dessen Implementierung für den zu erweiternden Honeypot, da dieser die Funktionen des Secure Shell Protokolls bereits beinhaltet.
Trotzdem stellt sich noch immer die Frage, warum TELNET noch immer genutzt wird. Dies liegt hauptsächlich darin begründet, dass weniger Leistung benötigt wird, da durch das Protokoll keine Verschlüsselungen verlangt werden. Dadurch ist das TELNET-Protokoll praktischer für Geräte, die günstig bleiben sollen und daher nicht viel Leistung bieten können.

3.3 Zusätzliche Funktionen nach der Erweiterung

Wie im vorigen Abschnitt beschrieben, soll das TELNET Protokoll die Kommunikation zwischen zwei Endgeräten ermöglichen. IoT-Geräte nutzen dieses zur Konfiguration von außen. Das Protokoll kann aber ebenso von dem Gerät aus zur Kontrolle anderer Geräte genutzt werden. Laut einer Studie japanischer Wissenschaftler hat sich seit 2014 die Menge der Angriffe auf Geräte mit dem TELNET-Protokoll stark erhöht. So gab es 2008 kaum Angriffe dieser Art, wobei es danach einen starken Anstieg der Angriffe

pro Darknet-IP-Adresse gab.(vgl. Pa u. a., 2016) Die entsprechende Entwicklung ist in Abbildung 3.3 abzulesen.

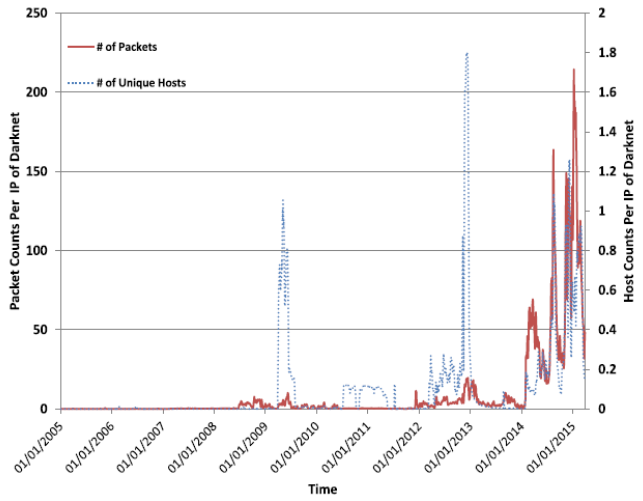

Abbildung 3.3: Anzahl der Angriffe auf TELNET pro Darknet-IP-Adresse (vgl. Pa u. a., 2016)

Dies ist eindeutig auf die Entwicklung der IoT-Geräte zurückzuführen, welche das erste Mal 2009 aufkamen. Daher ist es in letzter Konsequenz wichtig, TELNET in die Honeypots aufzunehmen und damit das zuvor schon zur Analyse genutzte SSH-Protokoll zu ergänzen.
Außerdem werden bei der Erweiterung die Daten des Honeypots durch den ELK-Stack eingelesen und aufbereitet. Sollte sich in Zukunft an der Form der Daten oder der Version des Honeypots etwas ändern, muss nur Logstash als filterndes System angepasst werden, um die Daten weiterhin so ablegen zu können wie gedacht. Dasselbe gilt für das TELNET-Protokoll, das so entwickelt wird, dass die Daten, obwohl sie keiner Filterung bedürfen, weil sie im Programm bereits wie benötigt formatiert werden, über Logstash in Elasticsearch eingefügt werden.

3.4 Methodik der Angriffserkennung

Die trivialste und in einigen Fällen auch die schnellste Methode, um unberechtigt Zugang zu einem TELNET System zu erreichen, ist das Erraten der Login/Passwortkombination. Dabei wird meist von bekannten oder häufigen Zugangskennungen (Logins) wie Admin, Guest, Field, Service, u.a. ausgegangen und versucht, das dazu passende Passwort zu erraten. Dies zeigte eine Studie aus dem Jahr 2016, wonach Angreifer bei Verwendung von TELNET für den Angriff häufig die Methode des Password Guessings nutzten, um an gültige Daten zu gelangen. Dabei wurden die Angriffe teils mit System, manchmal allerdings auch ohne Reihenfolge ausgeführt.
In diesem Fall wurden die Kombinationen zufällig zusammen ausgeführt. Im Folgenden beschäftigt sich der Autor mit der Methode des Passwort-Ratens und der Erfassung und Auswertung der abgesetzten Befehle nach dem erfolgreichen Login. Zunächst wird nun das TELNET-Protokoll für den Honeypot als zusätzlicher Dienst implementiert. Dieses wird so eingerichtet, dass zum Zugriff auf den Honeypot ein Login mit Username und Passwort verlangt wird. Allerdings wird das System so manipuliert, dass nach einer konfigurierbaren Anzahl von erfolglosen Login-Versuchen der Login tatsächlich dennoch erfolgt. Hierbei ist nicht relevant, welcher Username und welches Passwort verwendet werden. Bei diesem Vorgehen kann erfahren werden, ob Angreifer, die mit dem Raten von Login-Daten versuchen sich Zugang zum Gerät zu verschaffen, diese erfolgreichen Zugangsdaten für spätere Angriffe erneut zu nutzen.
Um Rückschlüsse ziehen zu können, wie die Angreifer agieren, werden die Login-Versuche protokolliert und die Login-Daten, welche genutzt wurden, inkl. Zeitstempel gespeichert. Dabei ist außerdem wichtig, dass in den Logs erfasst wird, welche Username- und Passwort-Kombinationen benutzt wurden und welche erfolgreich waren.
Falls der Angreifer sich zu einem späteren Zeitpunkt erneut anmelden möchte, mit diesen Informationen ist es dann möglich festzustellen, ob er versucht sich mit derselben Kombination Zugang zu verschaffen. Versuchen Angreifer erneut mit einer bereits als gültig erkannten Username und Passwort Kombinationen sich Zugang zu verschaffen, so wird dies zugelassen.

Ist der Angreifer dann erfolgreich eingeloggt, wird jeder Befehl, der gegeben wird, mit einer Standartantwort abgewiesen und festgehalten, welche Befehle abgesetzt wurden.

3.5 ELK-Stack

Im Honeypot werden die gewonnen Daten des Angreifers in einer Textdatei (Log-Datei) abgelegt. Dies geschieht in drei unterschiedlichen Logging-Services, dem Alarm-Logger, dem Debug-Logger und dem Malware-Logger. Diese Logging-Daten werden durch Logstash aus der vorherigen Log-Datei gelesen, wobei Logstash sich merkt, welche Einträge sich schon in darunter liegenden Datenbank Elasticsearch befinden. Daraufhin werden die Daten noch verarbeitet und hierbei können unterschiedliche Daten ausgelesen und normalisiert werden. Logstash ist vorteilhaft in Hinblick auf die Log-Daten, die von der Software bei Nutzung der anderen Protokolle erzeugt werden, da diese zum Zweck der Vereinheitlichung der Logs nicht verändert werden müssen. So wird im vorliegenden Fall das Datum der Dateien vereinheitlicht und in einem Datumsfeld abgelegt. Außerdem werden hier die verschiedenen Typen von Attacken klassifiziert und die IP-Adressen der Angreifer als geografische Daten in einem Feld dafür gespeichert. So ist es später mit Hilfe der Visualisierungs-Software Kibana möglich, z.B. zu visualisieren, z.B. aus welche Ländern besonders oft angegriffen wurde. Ist die Verarbeitung abgeschlossen, werden die veränderten Daten als neues JSON per HTTP-Post an Elasticsearch weitergeleitet. Diese Daten werden dann in Elasticsearch unter dem Index Logstash gespeichert und können jederzeit per HTTP Request oder über Kibana abgerufen werden. Elasticsearch könnte aufgrund der Modularität des ELK-Stack auch durch eine andere Anwendung ersetzt werden.
Der ELK-Stack wird auch für die Speicherung und den Abruf der verschiedenen Login-Daten, pro IP-Adresse, genutzt.

3.6 Anforderungen

Die Anforderungen an das System ändern sich insofern, als dass eine JAVA Runtime benötigt wird, um den ELK-Stack nutzen zu können. Dabei muss zum momentanen Zeitpunkt mindestens Java 8 genutzt werden. Außerdem gibt es Einschränkungen

in der Nutzung von Python. Python darf nur in der maximalen Version 2.7 genutzt werden, da die genutzte Python-Bibliothek "telnetsrv"keine funktionierende Version für die Python-Interpreter ab Version 3.0 bietet. Eine ausgiebige Recherche des Autors hat ergeben, dass momentan keine alternative Bibliothek zur Verfügung steht.

4 Implementierung

4.1 Bibliotheken und Frameworks

Zur Implementierung wurde Ubuntu als Betriebssystem verwendet, da dieses weit verbreitet ist und eine Linux-Umgebung für Tests notwendig ist, da das zu erweiternde Honeypot System für Linux-Server geschrieben und optimiert wurde.

4.1.1 Python

Python wird als Programmiersprache genutzt, da es sich anbietet dieselbe Scriptsprache zu nutzen, wie im zu erweiternden System, da dadurch die Nutzung einer Schnittstelle nicht notwendig ist. Außerdem ist es vorteilhaft Python zu benutzen, da es in jedem Linux-System schon vorhanden ist und leicht zu verstehen und zu warten ist. Problematisch stellt sich nur die Verwendung der Python-Version dar, da manche der genutzten Frameworks nicht mit einer Version von Python 3.0 oder höher kompatibel sind. Daher muss das System mit Python 2.7 genutzt werden.

4.1.2 JSON

JSON ist ein universelles Format zur Kodierung von zu verarbeitenden Daten. JSON wird genutzt, da die Elasticsearch-Datenbank Dokumente zum Ablegen und Aufrufen im JSON Format annimmt und abgibt. Daher wurde die Honeypot Software so angepasst, dass die log-Daten direkt im JSON-Format angelegt und per API an Elasticsearch weitergeleitet werden, anstatt sie in einem Dateiformat auf dem lokalen Dateisystem zu speichern und sie danach mühsam einzulesen und in ein Auswertungsprogramm zu übertragen.

4.1.3 YAML

YAML wird zur Beschreibung von Konfigurationsdateien benutzt und ist somit eine Datenserialisierungssprache, welche auf Unicode basiert und für unterschiedliche Plattformen genutzt werden kann. In dieser Arbeit wurde YAML zur Konfiguration des Elastic-Stacks genutzt, um vorhandene Daten zu serialisieren, zu vereinheitlichen und in Elasticsearch abzulegen.
Ein Beispiel ist zu finden in der Datei "telnet.cfg", welche in Abbildung 4.1 zu sehen ist.

```
input {
        http {
                port => "4200"
                codec => json
        }
}
filter{
        geoip {
                source => "clientip"
        }
}
output {
        if [headers][request_method] == "PUT" {
                elasticsearch{
                        hosts => ["localhost:9200"]
                        document_type =>"telnet"
                        action => update
                        document_id => "%{document_id}"

                }
        }
        else{
                elasticsearch {
                        hosts => ["localhost:9200"]
                        document_type =>"telnet"
                }
        }
        stdout { codec => rubydebug }
}
```

Abbildung 4.1: Logstash Konfiguration Input-File für TELNET-Logs

4.1.4 telnetsrv

Die telnetsrv-Bibliothek bietet die Möglichkeit, unkompliziert einen TELNET-Server aufzusetzen, da nur eine Subklasse mit den gewünschten Befehlen geschrieben werden muss. Die Bibliothek übernimmt die Einhaltung des TELNET-Protokolls nach RFC854 (1983).

4.2 Elastic Stack

4.2.1 Elasticsearch

Um den Elasticsearch nutzen zu können muss nur das System von der Website "https://elastic.co/elasticsearch"heruntergeladen werden. Dort stehen verschiedene Dateiformate zum Download zur Verfügung. Nach dem Download kann Elasticsearch einfach gestartet werden. Um eine gewisse Sicherheit in den Elastic Stack zu bekommen kann ein Plug-In namens "x-pack"genutzt werden, welches eben diese Sicherheitsfunktionen mitbringt. Nach der Installation des Plug-Ins muss ein Passwort für die User "Elasticünd "Kibanaërzeugt werden. Dafür gibt es in dem Plug-In ein Programm, welches Passwörter generiert. Danach muss die Kibana.yml, wie in Abbildung 1, geändert werden, damit Kibana die Elasticsearch Daten abfragen kann. Diese Funktion ist allerdings kostenpflichtig und mit der Basic Lizenz des Programms nicht nutzbar. Deshalb kann im Rahmen dieser Arbeit nicht darauf zurück gegriffen werden. Wenn Elasticsearch läuft, kann mit einem Post-Befehl standardmäßig auf IP-Adresse 127.0.0.1 Port 9200 die Dokumente in Form von JSON-Dokumenten abgelegt werden.
Dieser Endpunkt wird genutzt um die von Logstash veränderten Datensätze auf Elasticsearch abzulegen, sollte aber um die Modularität des Stacks zu gewährleisten nicht direkt zum ablegen der Dokumente genutzt werden. Diese Logs, welche in der Konfiguration der Software durch die Services, die nicht Teil der Thesis sind, erzeugt werden, werden in dieser Arbeit in Elasticsearch ebenfalls eingefügt, indem Logstash genutzt wird, welches in der Lage ist, durch Filter, die vorhandenen Logs einzulesen und in ein JSON zu verwandeln, welches dann in Elasticsearch abgelegt werden kann.
Hierzu muss lediglich der Logfile geschrieben werden und sobald Logstash gestartet wird, werden sämtliche Logs aus dem File, welcher in der Logstash.conf Datei beschrieben ist, in Elasticsearch übernommen.In dieser Konfigurationsdatei werden Input aus verschiedenen Quellen in Logstash angegeben, sowie den Output nach Elasticsearch konfiguriert. Dies geschieht in einer Datei mit dem Namen "Pipelines.yml", in welcher für jeden Service, der Daten zu Logstash in einer sogenannten "Pipeline"transportiert, eine eigene Konfigurationsdatei angegeben wird. Jede dieser Konfigurationsdateien gibt Inputquelle, falls gewünscht Filter und Output-Resource für die Logstash-Pipeline an, die genutzt werden sollen. Dabei können diese auch variieren. In der vorliegenden

Arbeit ist dies, wie in Abbildung 4.2 (Pipeline der zuvor genutzten Logs) und Figure 4.4 (die TELNET-Pipeline) konfiguriert. Außerdem müsste, wenn Authentifikation gewünscht ist, das Passwort und der User ebenfalls in diesen Dateien angegeben werden müssen.

```
input {
        file {
                path => "/var/log/spot/spot.log"
        }
}
filter{
}

output{
        stdout { codec => rubydebug }
        elasticsearch { hosts => ["localhost:9200"] }
}
```

Abbildung 4.2: Logstash Konfiguration Input-File für bestehende Logs

```
input {
        http {
                port => "4200"
                codec => json
        }
}
filter{
        geoip {
                source => "clientip"
        }
}
output {
        if [headers][request_method] == "PUT" {
                elasticsearch{
                        hosts => ["localhost:9200"]
                        document_type =>"telnet"
                        action => update
                        document_id => "%{document_id}"

                }
        }
        else{
                elasticsearch {
                        hosts => ["localhost:9200"]
                        document_type =>"telnet"
                }
        }
        stdout { codec => rubydebug }
}
```

Abbildung 4.3: Logstash Konfiguration Input-File für TELNET-Logs

4.2.2 Kibana

Nachdem im vorigen Abschnitt Elasticsearch betrachtet wurde, wird nun ein Blick auf Kibana geworfen, welches ebenfalls ein Teil des Elastic-Stacks ist und dessen grafische Oberfläche darstellt, wie bereits im Kapitel 2, Grundlagen erläutert. In dieser Arbeit soll Kibana nun genutzt werden, um auf die Log-Daten in Elasticsearch zugreifen zu können, welche dort durch Logstash abgelegt wurden. Dabei könnte in Zukunft festgestellt werden, ob sich Angriffe wiederholen oder zum Beispiel aus bestimmten Regionen Häufungen zu erkennen sind. Kibana bedarf keiner komplizierten Konfiguration, außer der Auswahl des Indices, welcher betrachtet werden soll. Dies wird in der grafischen Oberfläche beim ersten Login allerdings anschaulich erklärt.
Wurden durch Logstash geografische Daten ermittelt, können diese durch einen Geo-Hash von Kibana in einer Weltkarte sichtbar gemacht werden. Dies wird in Kapitel 5 Zusammenfassung und Ausblick im Rahmen der technischen Überprüfung für den vorliegenden Fall dargestellt.

4.3 Integration des Servers in den Honeypot

Die Integration in den Honeypot ist über einen eigenen Connection Handler gelöst, der den TELNET-Handler aufruft, in dem sämtliche Funktionen des Protokolls ausgeführt werden. Im folgenden Abschnitt werden diese erläutert und die Funktionsweise der Implementierung erklärt.

4.4 Implementierung des TELNET-Protokolls

Nachdem zuvor die Integration in den ELK-Stack beschrieben wurde, wird nun im Folgenden die Implementierung der Erweiterung der unterstützten Protokolle, der Honeypot-Software betrachtet.
In der Datei des TELNET-Handlers muss eine zweite Klasse implementiert werden, die nur einen Konstruktor beinhaltet, der die Klasse Protocol vererbt, da das Protokoll ansonsten nicht gefunden wird. Allerdings braucht die Klasse ansonsten keinerlei Funktionen. Diese werden von der Klasse "_TELNETHandler"übernommen.

Um das Protokoll nach dem zugehörigen (vgl. RFC854, 1983) zu implementieren wird in diesem Fall die Open Source Bibliothek "telnetsrv"genutzt. Diese bietet die Möglichkeit das Protokoll RFC-gerecht zu nutzen, ohne den Aufwand der Implementierung des gesamten Protokolls. Hierzu wird lediglich ein Handler geschrieben, welcher eine Subklasse des Handlers ist, der durch die Bibliothek bereitgestellt wird. In diesem Handler können verschiedene Befehle konfiguriert werden, die das System anbieten soll. Im vorliegenden Fall werden keine solchen Befehle implementiert, da der Honeypot nur den Versuch eines Befehls loggen soll, diese aber nicht ausführt.

Es wird eine Authentifikation eingebaut. Verbindet sich ein Angreifer mit der Adresse, so wird er aufgefordert sich anzumelden. Versucht er dies, wird die Methode "Auth-Callbackäusgeführt, welche in Elasticsearch prüft, ob die gegebene IP-Adresse sich bereits in der Datenbank befindet und entsprechend der folgenden Szenarien reagiert:
Erster Login-Versuch der IP-Adresse, die vorher noch nie einen Login-Versuch unternommen hat:
Beim ersten Login-Versuch einer IP-Adresse wird ein Eintrag in Elasticsearch angelegt, welcher aus dem Typ des Angriffs, einem Zähler für missglückte Login-Versuche und einer Liste der genutzten Anmeldedaten besteht. Pro Eintrag in der Liste werden Passwort und Username sowie mithilfe eines Zählers die Anzahl der Login-Versuche für die spezifische Kombination und eine Liste von Zeitstempeln für die verschiedenen Login-Versuche festgehalten. Als letztes wird eine Variable mitgeführt, die über "True-öder "Falseängibt, ob die Kombination als valide Login-Daten gezählt wird. Im ersten Versuch wird dieser immer auf "False"gesetzt, da die Software nicht jeden Angreifer beim ersten Versuch, sich anzumelden, das Login zulassen soll. Ist das Anlegen des Datensatzes erfolgreich, so wird die Verbindung beendet und dem Angreifer zuvor mitgeteilt, dass die angegebenen Daten nicht zulässig gewesen sind.
Datensatz vorhanden und Username/Passwort-Kombination nicht zulässig:
Ist der Datensatz für die IP-Adresse bereits vorhanden, wird die Liste der bereits genutzten Kombinationen aus Username und Passwort auf die angegebenen Daten durchsucht und geprüft, ob diese bereits genutzt wurden.
Ist dies nicht der Fall wird in der Liste der Anmeldedaten ein neuer Eintrag angelegt. Dabei wird geprüft, ob die Anzahl der nicht erfolgreichen Login-Versuche für diese

IP-Adresse bereits so hoch ist, dass der Honeypot Zugang gewähren soll. Je nach Ergebnis dieser Prüfung wird der Eintrag mit der Variable für den Login mit "Trueöder "False"vervollständigt.
Ist der Versuch als "True"verzeichnet, so wird der Counter wieder auf 0 gesetzt und der Angreifer authentifiziert. Anderenfalls wird der Login-Versuch abgewiesen und der Counter erhöht.
Versucht sich jemand von einer IP-Adresse zu verbinden und nutzt bereits zugelassene Anmeldedaten, so wird er authentifiziert. Allerdings wird in der Liste der genutzten Daten der Datensatz dahingehend verändert, dass die Anzahl der Nutzung der Username/Passwort-Kombination erfolgreich genutzt wurde.

4.5 Problemstellungen während der Implementierung

Zuerst versuchte der Autor dieser Arbeit den TELNET-Server mit Hilfe der Bibliothek 'telnetsrv' mit Python Version 3.5 zu implementieren. Leider gab es das Problem, dass die Bibliothek entgegen der Zusicherung der Herausgeber nicht mit Python 3 kompatibel war, da eine Klasse nicht im Paket vorhanden war. Nachdem der Versucht damit mehrere Male fehlschlug wurde die neue Klasse mit Python 2.7 implementiert, worauf hin die Bibliothek genutzt werden konnte und die Modalitäten des TELNET-Protokolls nach (vgl. RFC854, 1983) implementiert werden konnte. Ein weiteres Problem bestand darin, dass der Index in Elasticsearch ein Mapping für einen Index nicht ändert. Es muss immer ein neuer Index angelegt und die Dokumente von einem Index in den anderen übernommen werden, sollte man das Mapping eines Indexes ändern wollen.

5 Zusammenfassung und Ausblick

Im Folgenden werden die Ergebnisse der Arbeit überprüft und die technische Entwicklung dargestellt. Außerdem werden die Ergebnisse eines Testlaufs der Software vorgestellt sowie ein Fazit gezogen sowie ein Ausblick für die künftige Fortführung präsentiert.

5.1 Technische Überprüfung

Zur technischen Überprüfung der in der Arbeit präsentierten Schritte wurde der Honeypot 24 Stunden an ein mit dem Internet verbundenes Netzwerk angeschlossen und hat dadurch Daten sammeln können. Leider ergaben sich dadurch keine neuen Erkenntnisse, da es in dieser Zeit keine Zugriffe auf die IP-Adresse gab. Daher wurden die gewünschten Funktionen manuell per TELNET-Zugriffsversuch getestet. Dabei wurde getestet, wie das Programm auf leere Strings als Passwort oder Username reagiert und ob das Programm die Authentifizierung nach einer gewissen Anzahl an Versuchen von einer IP-Adresse zulässt. Beides stelle sich als möglich heraus. Außerdem wurde getestet, ob verschiedene Kombinationen richtig gespeichert werden und damit das Programm alle Login-Daten auffassen kann. In Abbildung 5.1 sind beispielhafte Einträge in Elasticsearch zu sehen, wie sie in Kibana stehen bevor eine Visualisierung durchgeführt wurde.

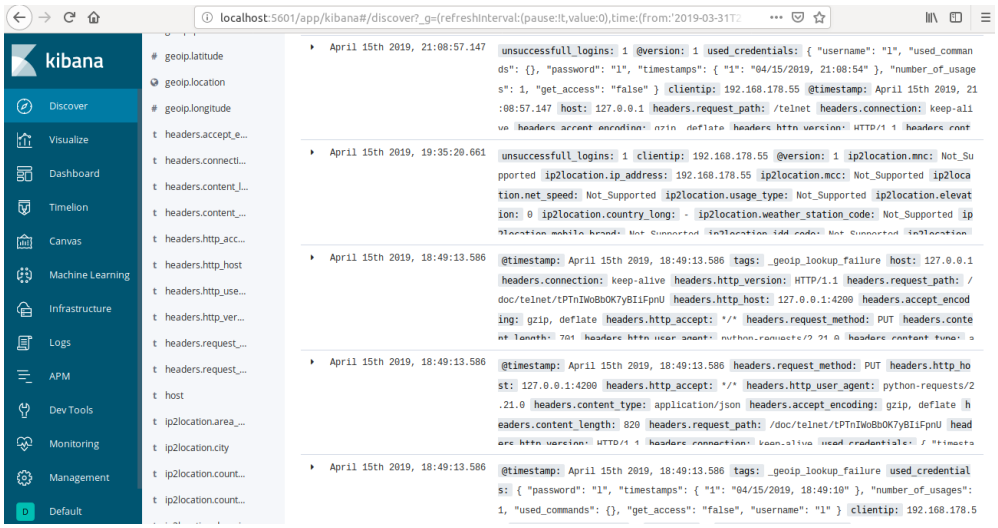

Abbildung 5.1: Kibanadaten ohne Visualisierung

Um die Überprüfung abzuschießen wurden die gesammelten Daten in Kibana geladen, um eine Auswertung mit Hilfe der dort zur Verfügung stehenden Tools vorzunehmen. Hierfür muss Kibana in der GUI konfiguriert werden. Beim ersten Login muss der Index, welcher analysiert werden soll, angegeben werden. Danach können Auswertungen angestoßen werden. Zur Überprüfung der zuvor gesammelten Daten durch den Honeypot wird eine Auswertung der Geoip-Daten sowie einer Grafik der Zugriffe auf das TELNET-Protokoll des Honeypots pro Zeiteinheit genutzt. In Abbildung 5.2 ist die Auswertung der Daten zu sehen, die die Anzahl der Zugriffe pro Zeiteinheit zeigt.

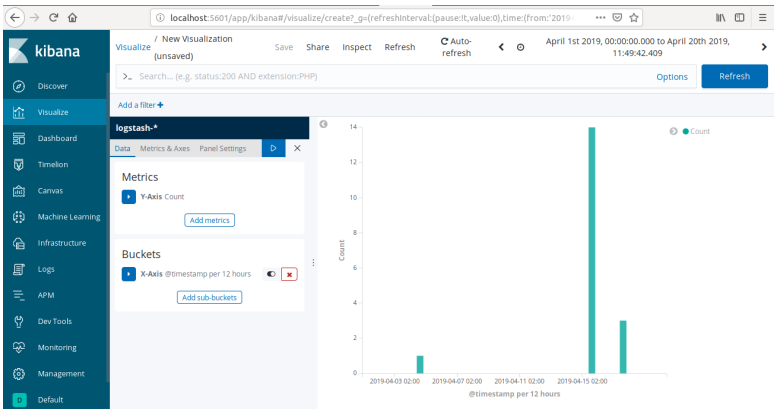

Abbildung 5.2: Zugriffe pro Zeiteinheit in Kibana

Als Abschluss der Überprüfung wird in Abbildung 5.3 einer Karte angezeigt, in der farblich hervorgehoben zu sehen ist, aus welchem Land die IP-Adressen der Angreifer kommen. In diesem Fall sind diese alle aus Deutschland, da es sich nur um die verwendeten Testdaten handelt.

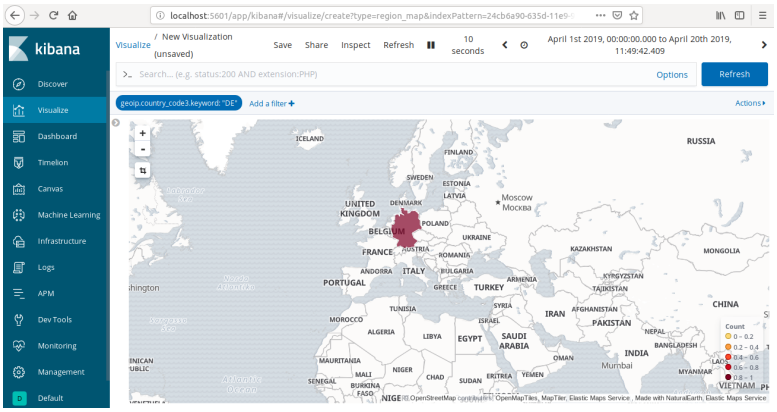

Abbildung 5.3: Farbige Karte der Auswertung der IP-Adresse in Kibana

Unter dem Reiter Dashboard können auch verschiedene Diagramme gleichzeitig angezeigt werden.
Die Grafiken der Auswertung zeigen, dass auch die Auswertung in Kibana wie gewünscht möglich ist. Somit kann im letzten Abschnitt nun ein Fazit und ein Ausblick für folgende Arbeiten gegeben werden.

5.2 Fazit

Zum Schluss kann nun das Ziel überprüft und ein Fazit gezogen werden. Das Ziel der Arbeit war es IoT-Geräte und deren Angriffe erkennbar zu machen, indem das TELNET-Protokoll umgesetzt sowie erkannt und protokolliert wird, was der Angreifer versucht zu tun. Dieses Ziel ist, wie im vorigen Teil der Arbeit zu sehen ist, geglückt. Die sogenannten lateralen Bewegungen werden nicht, wie zunächst gedacht, explizit erkannt. Allerdings ist das Erkennen nachträglich möglich, da sämtliche von Angreifer ausgeführten Befehle dokumentiert werden.
Zusätzlich ist eine Integration des ELK-Stacks geglückt und eine Veränderung der Anforderungen über die Erweiterung der Logstash-Konfiguration möglich. Im letzten Teil der Arbeit wird nun ein Ausblick für die Zukunft gegeben und mögliche weitere Erweiterungen vorgeschlagen.

5.3 Ausblick

In der Zukunft werden die von der Menschheit genutzten technischen Geräte noch weit mehr vernetzt sein. Wenn Maschinen immer mehr Bereiche unseres Lebens übernehmen, steigt die Anzahl der Gefahren für diese und durch diese. Dies wird weiterhin ein immer weiter wachsendes Problem werden, welches dazu führt, dass die Sicherheit im IT-Sektor immer mehr an Bedeutung gewinnen wird und auch muss. Speziell die Gefahr durch Angriffe auf IoT-Geräte und dann von IoT-Geräten muss beobachtet und studiert werden, damit hiergegen Maßnahmen getroffen werden können. Sollte sich der Stellenwert solcher Abwehrmaßnahmen nicht in naher Zukunft in der Gesellschaft erhöhen, könnten Angriffe wie "Wannacry"bald zur Tagesordnung gehören und dann gefährlicher und größer denn je gegen die Wirtschaft, das Militär, die Infrastruktur oder die Bevölkerung gerichtet sein. Deshalb wird es umso wichtiger werden, die Gefahren zu analysieren und mit Hilfe von künstlicher Intelligenz Wege zu finden, mit denen die Gefahren zumindest beherrschbar gehalten werden, denn komplett sicher kann man im Internet in der heutigen Zeit vor solchen Angriffen nicht mehr sein. Als mögliche zukünftige Erweiterungen zur Erhöhung der Fähigkeiten des Honeypots könnten in Zukunft die restlichen Protokolle dahingehend verändert

werden, dass diese nicht mehr ihre Logs in einem Textformat in einem File auf dem Server ablegen, welche dann von Logstash ausgelesen und zu Elasticsearch weiterleitet werden. Stattdessen könnten diese Protokollimplementierungen ebenfalls direkt ihre Logs mit Logstash in Elasticsearch ablegen. Im Übrigen könnte es dazu kommen, dass das Programm erneut etwas angepasst werden muss, da laut kürzlich erfolgter Ankündigung der Autoren von Elasticsearch sich das Ablagesystem mit Version 7.0 dahingehend ändert, dass Dokumenttypen entfernt werden.

Literaturverzeichnis

[Sta16 2016] : *Statista.com.* 2016. – URL https://de.statista.com/statistik/daten/studie/537093/umfrage/anzahl-der-vernetzten-geraete-im-internet-der-dinge-iot-weltwe

[Elastic19 2019] : *elastic.co.* April 2019. – URL https://www.elastic.co/elk-stack

[search19 2019] : *Elastic.co.* April 2019. – URL https://www.elastic.co/products/elasticsearch

[Ashton 2009] Ashton, Kevin: That 'Internet of Things' Thing. In: *RFID Journal* (2009), Juni

[Craig 2016] Craig, Scott: Telnet: An Attacker's Gateway to the IoT. In: *securityintelligence* (2016). – URL https://securityintelligence.com/telnet-an-attackers-gateway-to-the-iot/

[Heard Nicholas 2016] Heard Nicholas, A.: *Dynamic Networks And Cyber-security.* World Scientific Publishing Europe Ltd, 2016. – URL https://www.ebook.de/de/product/25725487/heard_nicholas_a_dynamic_networks_and_cyber_security.html. – ISBN 1786340747

[IEEE 2015] IEEE: Towards a definition of the Internet of Things (IoT), 2015

[Kozierok 2005] Kozierok, Charles: *The TCP/IP Guide: A Comprehensive, Illustrated Internet Protocols Reference.* San Francisco, CA, USA : No Starch Press, 2005. – ISBN 159327047X, 9781593270476

[Krämer u. a. 2015] Krämer, Lukas ; Krupp, Johannes ; Makita, Daisuke ; Nishizoe, Tomomi ; Koide, Takashi ; Yoshioka, Katsunari ; Rossow, Christian: AmpPot: Monitoring and Defending Against Amplification DDoS Attacks. In: *Research in Attacks, Intrusions, and Defenses*. Springer International Publishing, 2015, S. 615–636

[Lonvick und Ylonen 2006] Lonvick, Chris M. ; Ylonen, Tatu: *The Secure Shell (SSH) Protocol Architecture*. RFC 4251. Januar 2006. – URL https://rfc-editor.org/rfc/rfc4251.txt

[Mairh u. a. 2011] Mairh, Abhishek ; Barik, Debabrat ; Verma, Kanchan ; Jena, Debasish: Honeypot in network security. In: *Proceedings of the 2011 International Conference on Communication, Computing & Security - ICCCS '11*, ACM Press, 2011

[Pa u. a. 2016] Pa, Yin Minn P. ; Suzuki, Shogo ; Yoshioka, Katsunari ; Matsumoto, Tsutomu ; Kasama, Takahiro ; Rossow, Christian: IoTPOT: A Novel Honeypot for Revealing Current IoT Threats. In: *Journal of Information Processing* 24 (2016), Nr. 3, S. 522–533

[RFC854 1983] RFC854: *Telnet Protocol Specification*. RFC 854. Mai 1983. – URL https://rfc-editor.org/rfc/rfc854.txt

Erklärung zur selbständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Hamburg, 23. April 2019

Constantin Wahl