INSTITUTE FOR NETWORKED SOLUTIONS

OST Ostschweizer Fachhochschule

MISP
Malware Information Sharing Platform

Studienarbeit

Studiengang Informatik
OST - Ostschweizer Fachhochschule
Campus Rapperswil-Jona

Herbstsemester 2021

Autoren:	Janis Wolf
	Marius Zindel

Ausgabe:	E-Prints
Version:	23. Dezember 2021
Betreuer:	Ivan Bütler

Abstract

Problemstellung

Das Cyber Defense Modul an der OST vermittelt Studierenden diverse Tools und Methoden, um sich besser gegen Cyber Kriminelle zu schützen. Dazu gehört auch MISP (Malware Information Sharing Platform), eine Software für die Klassifizierung und den Austausch von Indicators of Compromize (IOCs) und Malware Samples.

Ziel der Arbeit

Ziel dieser Arbeit ist die Erstellung von 8 verschiedenen Laborumgebungen auf der HackingLab Plattform, um den Studierenden einen Einblick in MISP zu geben. Ein Lab beinhaltet jeweils eine Einführung ins Thema, eine Problemstellung mit Step by Step Anleitungen, sowie Verständnisfragen und eine Lernkontrolle.

Ergebnis

Es wurden 8 unterschiedliche MISP-Labs erstellt. Sie behandeln diverse Features von MISP und decken einen grossen Teil der MISPFunktionalität ab. Auf GitHub wurde ein Repository eingerichtet, welches die MISP Installation auf Basis von Docker automatisiert. Die Docker Labs sind modular und können einfach via Skript (Python) angepasst werden. Die Übungen sind auf die Hacking-Lab LiveCD https://livecd.hacking-lab.com/ abgestimmt.

Inhaltsverzeichnis

KAPITEL 1

Aufgabenstellung

Titel: MISP - Malware Information Sharing Platform
Fachgebiet: Sicherheit
Betreuende: Ivan Bütler
Beschreibung:

Einführung

Es gibt verschiedenste verteilte Datenbanken. Sie kennen sicher DNS - ein System um weltweit DNS Namen aufzulösen. So etwas ähnliches gibt es aber auch für die Abwehr von Cyber Attacken. Man nennt das verteilte System MISP - Malware Information Sharing Platform. Das GovCERT der Schweiz und viele Schweizer Banken oder MELANI setzen auf die Funktion von MISP.

Mit MISP können Unternehmen und Organisationen das Verhalten von Malware in einer Datenbank festhalten und Anderen als Feed für deren Cyber Abwehrsysteme bereitstellen. Stellen Sie sich so etwas wie einen Feed von Malware, Taktik und Technik vor.

Hintergrund

Die OST beginnt im HS2021 mit dem neuen Modul "Cyber Defense" und startet ab FS2022 das neue "Incident Response Modul". Um den Studierenden einen praktischen Zugang zum Thema Cyber Defense und Incident Response zu geben, soll das Hacking-Lab um eine MISP Lernumgebung erweitert werden.

Aufgabenstellung

Im Rahmen dieser Arbeit sollen MISP basierende Übungen und die MISP Platform selbst im Hacking-Lab entwickelt werden, welches durch Studierende der obigen Module für Übungszwecke verwendet werden kann. Die MISP Platform wird docker-compose auf der Hacking-Lab LiveCD hochgefahren und steht dort lokal für die Übungen bereit. Das Framework für die Übungen soll einfach erweiterbar sein. Die Übungen sind in der Englischen Sprache zu erstellen, wie auch Musterlösungen der Übungen bereitstehen.

Lernziele MISP Labs

Die MISP Labs sollen folgende Lernziele verfolgen

- Kennenlernen MISP Funktionalität
- Erstellung von Events, Objects und Attribute
- Erstellung von Correlation Graph und Visualisierung der Ereignisse
- Anreicherung MISP durch externe Feeds
- Verständnis für das Sharing Konzept von MISP fördern

Docker-Labs

Auf der Hacking-Lab LiveCD steht ein Docker Daemon bereit, welcher für die MISP Labs genutzt werden sollen. Die Labs werden über ein Public Repo bereitgestellt, welches die Studierenden vor der Übung pullen und nutzen.

Abbildung 1.1: Aufgabenstellung Seite 1

MISP - Erklärung auf Englisch

MISP is a threat intelligence platform for gathering, sharing, storing and correlating Indicators of Compromise (IOC) of targeted attacks, threat intelligence, financial fraud information, vulnerability information or even counter-terrorism information. It will store IOCs in a structured manner, and allows the correlation, automated exports for IDS, or SIEM, in STIX or OpenIOC and synchronize to other MISP instances. It makes it easier to share and receive malware information from trusted partners and trust-groups

Rapperswil, 20. September 2021

Betreuer; Ivan Bütler Student Janis Wolf Student Marius Zindel

Abbildung 1.2: Aufgabenstellung Seite 2

KAPITEL 2

Management Summary

2.1 Ausgangslage

2.1.1 Weshalb Threat Intelligence Sharing essenziell ist

Täglich finden zahlreiche Angriffe auf Computersysteme statt. Gefährlich wird es, sobald Angreifer sich zusammenschliessen und Informationen miteinander teilen. Es wird immer mehr beobachtet, dass diese ihr Wissen über Sicherheitslücken, Informationen oder schädlichen Quellcode untereinander austauschen. Um dieser Gefahr entgegenzuwirken wird es zunehmend wichtiger, dass sich Unternehmen ebenfalls austauschen und Informationen über erkannte Angriffe miteinander teilen. Dabei sollten auch konkurrierende Firmen zusammenarbeiten und gemeinsam gegen kriminelle Akteure vorgehen.

2.1.2 Situation

Fachexpertinnen und -experten für Computersicherheit sind vermehrt gefragt. Dies erkannte auch die Ostschweizer Fachhochschule und baut vermehrt am Campus Rapperswil-Jona Vertiefungsmodule im Computersicherheitsbereich auf. Der Bachelorstudiengang Informatik erhielt zusätzlich einen eigenen Studienschwerpunkt Cybersecurity, welcher mit dem schweizweit einmaligen Hacking-Lab die Studierenden in verschiedenen Themen mit praktischen Übungen beim Lernen unterstützt [1]. Für das Modul Cyber Defense, welches zum ersten Mal im Herbstsemester 2021 stattfand, sollte eine Übungsumgebung erarbeitet werden. Die Studierende sollten an das Thema Threat Intelligence Sharing mit der Malware Information Sharing Platform (MISP) herangeführt werden.

2.1.3 Lösungsansatz

Auf der für die Studierenden bereits bekannten Hacking-Lab Plattform soll eine weitere Übungsumgebung bereitgestellt werden. Die Idee ist es, diese in Form eines Labs pro Themenbereich zu erstellen. Die Struktur sollte möglichst so implementiert werden, dass die Labs in Folgearbeiten weiterentwickelt werden können.

2.2 Vorgehen

2.2.1 Ansatz

Für das Lösen der Übungen muss eine Infrastruktur zur Verfügung gestellt werden. Einerseits wird das Hacking-Lab verwendet, um die Studierende während den Aufgaben zu führen und ihnen Aufträge zuzuteilen. Andererseits muss die Malware Information Sharing Platform selbst durch die Übungsteilnehmer benutzt werden können.

2.2.2 Umfeld

Die Übungen sollen im Modul Cyber Defense direkt in der ersten Durchführung im Herbstsemester 2021 zum Unterrichtsstoff beitragen.

2.2.3 Technologien

Als Grundlage dient die Hacking-Lab LiveCD, welche für die Studierenden kostenlos zur Verfügung steht [2]. Malware Information Sharing Platform wird als Docker auf der Hacking-Lab LiveCD auf dem Computer der Studierende ausgeführt. Dabei können einerseits Kosten eingespart werden, da keine Ressourcen externer Cloud Anbietern verwendet werden müssen und andererseits können verschiedene Übungsszenarien einfach und zeitnah gestartet werden. Für die Initialisierung wird ein Python Skript verwendet, welches beim ersten Start von MISP über eine API die vorgegebenen Labordaten zur MISP Instanz hinzufügt.

2.3 Ergebnisse

2.3.1 Erreichte Ziele

Ursprüngliches Ziel dieser Arbeit war es, acht Übungsaufgaben bis zur 12. Semesterwoche zu entwerfen, welche die Studierenden direkt im selben Semester beim Erlernen der Malware Information Sharing Platform unterstützen sollte. Dies wurde dank eines straffen Zeitplans erfolgreich umgesetzt.
Zusätzlich wurde durch den Betreuer Ivan Bütler das Angebot gemacht, eine 90 minütige Vorlesung zu halten und das erarbeitete Wissen durch die Autoren direkt an die Studierenden weiterzugeben. Diese wurde nach organisatorischen Anpassungen im Zeitplan ebenfalls erfolgreich durchgeführt.

2.3.2 Nicht erreichte Ziele

Alle Ziele wurden erreicht. Während der Arbeit wurden insgesamt zehn Konzepte für Übungen erarbeitet, von welchen schlussendlich aus zeitlichen Gründen, sowie zur Sicherung der Qualität, die gewünschten acht Aufgaben implementiert wurden.

2.4 Ausblick

Eine Weiterentwicklung der Labs ist sehr gut möglich. Die noch offene Integration mit einem Security Information and Event Management System (zum Beispiel Wazuh) oder eines Network Intrusion Detection System (zum Beispiel Snort) könnte den Studierenden helfen, ihr bereits gelerntes Wissen mit MISP in Verbindung zu bringen. Ebenfalls können bereits umgesetzte Labs um weitere Details erweitert werden.

KAPITEL 3

Technischer Bericht

3.1 Einleitung und Übersicht

3.1.1 Ausgangslage

Computersicherheit ist ein sehr ernstzunehmendes Thema, welches jeden betrifft. Die Zahl der Angriffe auf natürliche Personen sowie vor allem auch Unternehmen wächst von Jahr zu Jahr stetig [3]. Angreifer tauschen dabei untereinander Wissen über Sicherheitslücken, Informationen oder gar schädlichen Quellcode untereinander aus. Umso schwieriger wird es auf der zu verteidigenden Seite. Ohne Insiderwissen zu aktuell laufenden Cyber-Angriffskampagnen auf eine spezifische Branche kann oft erst zu spät reagiert werden. Umso wichtiger ist es daher, dass Unternehmen zusammenarbeiten und gemeinsam gegen kriminelle Akteure vorgehen und diese abwehren. Ein dezentrales System, welches das Erfassen und den Austausch von Informationen zu Angriffen auf Computersysteme vereinfachen soll ist die Malware Information Sharing Platform, kurz MISP.

3.1.2 Problemstellung

Mit steigender Nachfrage für Computersicherheit steigt auch die Nachfrage nach Fachexpertinnen und -experten in diesem Bereich. Die Ostschweizer Fachhochschule und baut daher vermehrt am Campus Rapperswil-Jona Vertiefungsmodule im Computersicherheitsbereich auf. Eines davon ist das Modul Cyber Defense, welches die Studierenden auf der Verteidigungsseite ausbilden soll. Bisher fehlt jedoch eine Übungsumgebung auf welcher die im Unterricht behandelten Themen zu MISP praktisch vertieft werden können.

3.1.3 Vorarbeiten

Direkte Vorarbeiten, sei es eine Studien- oder Bachelorarbeit gibt es keine. Die Übungsumgebung zu MISP wird komplett neu erstellt und mögliche Aufgaben sowie Infrastruktur aufgebaut. Die Hacking-Lab LiveCD, welche für die Studierenden zur Verfügung steht, sowie die Hacking-Lab Plattform selbst kann jedoch für die Übungsszenarien eingesetzt werden.

3.1.4 Ziele

Im Rahmen dieser Arbeit sollen folgende Ziele erreicht werden:

- Erstellen einer Übungsumgebung zum Thema Threat Intelligence Sharing mit MISP
- Die Übungsumgebung soll ungefähr acht Labs enthalten
- Die Studierenden sollten folgende Lernziele in den Labs verfolgen:
  - Kennenlernen der MISP Funktionalität
  - Erstellung von Events, Attributes und Objects
  - Grafische Darstellungen entwerfen
  - Das Feed-Konzept in MISP kennenlernen
  - Synchronisation zwischen mehreren MISP Server umsetzen
- Eine 90 minütige Vorlesung zu MISP halten

3.1.5 Einschränkungen

Die praktischen Übungen werden als Docker Container auf den Computern der Studierenden selbst gestartet und durchgeführt. Docker ist ein sehr mächtiges Tool, welches Plattformübergreifend eingesetzt werden kann. Die Labs können daher in der Hacking-Lab LiveCD oder direkt auf dem Rechner des Studierenden ausgeführt werden. Aufgrund der für uns unbekannten Konfigurationen der privaten Notebooks halten wir uns daher vor, nur die Hacking-Lab LiveCD offiziell für die Übungen zu unterstützen.

3.1.6 Umsetzung

Als Plattform für die Aufgabenstellungen wird das Hacking-Lab verwendet. Dieses wird bereits im Modul Cyber Defense eingesetzt und ist den Studierenden bereits bekannt. Im Hacking-Lab können Aufgabenstellungen einfach in Markdown geschrieben werden.

Eine MISP Instanz gibt es bisher noch nicht an der Ostschweizer Fachhochschule. Daher wurde mehrere Konzepte entwickelt, wie MISP in verschiedenen Szenarien zur Verfügung stehen soll. Eine Übersicht sowie eine begründete Designentscheidung ist im Abschnitt *Systemübersicht* vorzufinden. Schlussendlich entschieden wir uns für eine Umsetzung mit mehreren Docker Container, welche in der Hacking-Lab LiveCD gestartet werden.

Um administrative Aufgaben für die Übungsteilnehmer möglichst zu minimieren wurde ein Konfigurationsskript in Python erstellt, welches beim ersten Start der Lab-Umgebung Daten und Einstellungen in MISP initialisiert. Das Konfigurationsskript nutzt die offizielle API von MISP und wurde so entwickelt, dass es möglichst einfach um weitere Labs erweitert werden kann.

3.2 MISP Research

3.2.1 Übersicht

In diesem Kapitel wird die notwendige Recherche für die Arbeit dokumentiert. Unbekannte Begriffe, sofern nicht erklärt, können dem Glossar entnommen werden. Die gesammelten Informationen konnten ebenfalls später für die Vorlesung verwendet werden.

3.2.2 Einführung

Zuerst zur Frage - Was ist MISP? Es folgt die offizielle Definition welche der CIRCL Website [4] entnommen wurde.

> MISP (Open Source Threat Intelligence and Sharing Platform) software facilitates the exchange and sharing of threat intelligence, Indicators of Compromise (IoCs) about targeted malware and attacks, financial fraud or any intelligence within your community of trusted members. MISP sharing is a distributed model containing technical and non-technical information which can be shared within closed, semi-private or open communities. Exchanging such information should result in faster detection of targeted attacks and improve the detection ratio, whilst also reducing the number of false positives.

Die oben erwähnte Definition beschreibt MISP sehr gut. Hauptsächlich wird MISP also dafür verwendet, Malware und Threats zu dokumentieren. Damit diese Dokumentation auch für Andere zur Verfügung gestellt werden kann lassen sich diese Informationen durch Feeds über Organisationen hinaus teilen. Durch die Community kann Malware so rasch und mit grosser Reichweite effektiv und teilweise sogar automatisiert bekämpft werden.
MISP ist Open Source Projekt und auf Github [5] verfügbar. Unterhalten wird MISP von CIRCL.

Was ist CIRCL?

Wie erwähnt ist MISP ein Open Source Projekt. Die Entwicklung wird unter anderem aber stark von CIRCL unterstützt und begleitet. CIRCL bietet mehrere öffentliche MISP Instanzen an, auf welchen produktive Daten zur freien Verfügung stehen.
CIRCL selbst besteht aus einem Team von Entwicklern in Luxemburg. Die Organisation bietet auch diverse Trainings an. Diese sind meisten auf IT Sicherheit ausgelegt. Natürlich werden auch MISP Trainings angeboten.

Wer verwendet MISP?

In einem grossen Unternehmen lauern mehr Gefahren und grössere Chancen auf Malware zu treffen. Diese Unternehmen haben dann auch eher die Kapazität die MISP Instanz zu betreuen. Sobald eine Malware auftritt kann diese dann in MISP erfasst und dokumentiert werden. Dies macht es für das IT Abteilung einfacher den Überblick zu behalten.

Einschränkungen von MISP

MISP basiert auf dem Vertrauen von Anderen. Das heisst, dass Daten welche von anderen MISP Instanzen in das eigene System synchronisiert werden, können auch Fehler enthalten. Diese Fehler führen zu potentiell gravierenden Auswirkungen.
Die Richtigkeit der Daten muss daher selbst geprüft werden und kann nicht automatiert getestet werden. MISP bietet an, dass User die Events als False Positiv melden können.
Bei der Auswahl, welchen Quellen man vertraut sollte man sich daher zuerst gut informieren.

Features von MISP

Eine Zusammenstellung der Features in MISP findet sich hier [6].

3.2.3 Events

Ein Event ist in MISP eines der grundlegenden Elemente. In einem Event werden Daten über einen Vorfall erfasst. Die enthalten Daten sind in Attribute oder Objekte verpackt.
Zwischen Events können dann durch die Attribute und Objekte Korrelationen zwischen Events hergestellt werden. Ein Event kann auch verschiedene Tags besitzen. Diese machen das Suchen und Filtern einfacher.
Um ein Event zu erstellen werden vorerst folgende Daten benötigt.

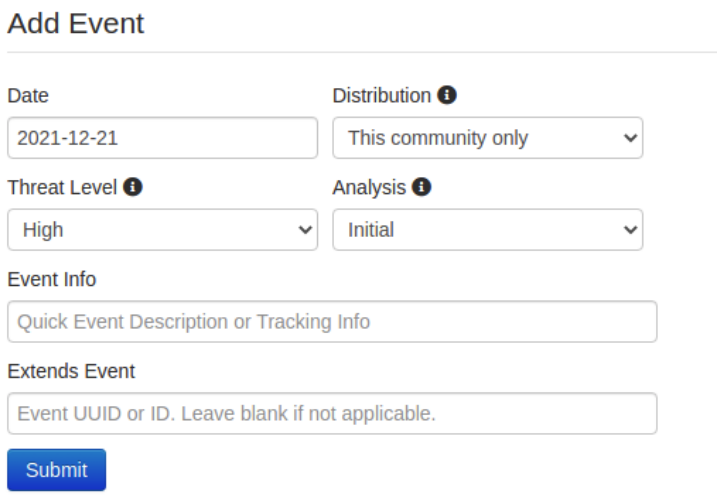

Abbildung 3.1: MISP Event erstellen

- **Date**: Wann das Ereigniss eingetroffen ist
- **Distribution**: Wie weit das Event an andere geteilt wird
- **Threat Level**: Welche Stufe an Wichtigkeit das Event hat
- **Analysis**: Aktueller Stand der Analyse
- **Event Info**: Titel des Events
- **Extends Event**: Hängt zusätzliche Daten an existierendes Event an

Wenn man ein Event nun betrachtet sieht man auf den ersten Blick die wichtigsten Daten.

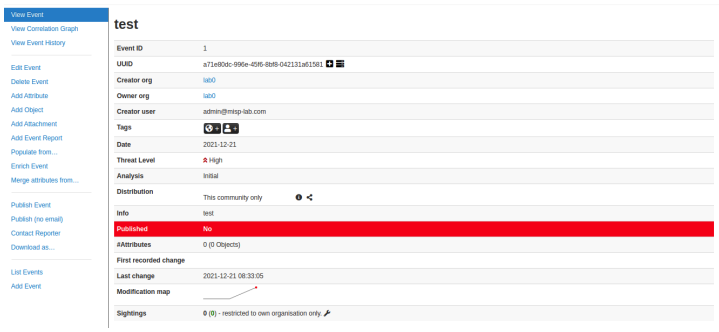

Abbildung 3.2: MISP Event betrachten

3.2.4 Attributes

Um ein Event strukturiert mit Daten zu ergänzen braucht es Attribute. Ein MISP Attribut hat immer eine Kategorie und einen Typ. Das Attribut vom Typ **ip-src** in der Kategorie **Network activity** kann beispielsweise nur genau einen Wert speichern, eine IP Adresse. Die Kategorien und Typen ermöglichen das Filtern und Suchen.

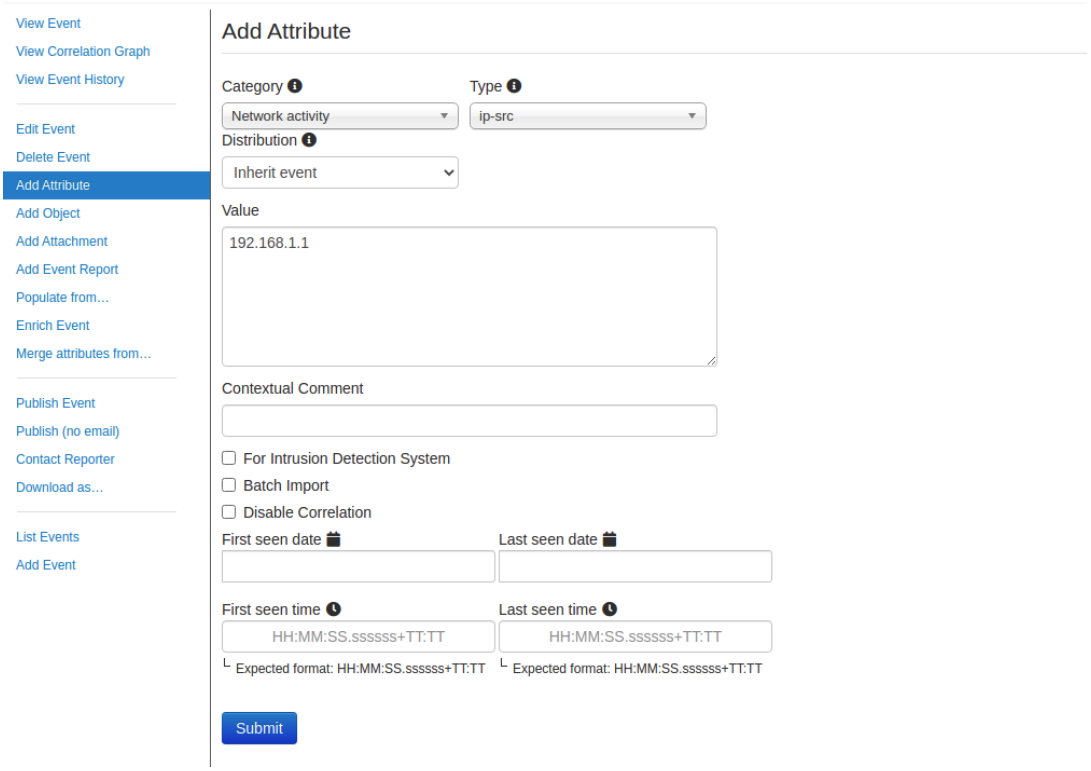

Abbildung 3.3: MISP Attribut erstellen

Auf einem Attribut können weitere Flags gesetzt werden. Diese Flags geben zum Beispiel das Distribution Level vor oder ob das Attribut relevant ist für ein IDS.

3.2.5 Objects

Objekte in MISP sind nützlich für die Erfassung von strukturierte Daten in einem Event. Es können mehrere Attribute in einem Objekt verpackt werden.
MISP bietet diverse Objektvorlagen an. Diese sind in Kategorien aufgeteilt (*climate, financial, health, internal, network* ...). Unter *network* gibt es zum Beispiel folgende Objekte

- asn
- domain-ip
- ip-port
- url
- whois
- ...

Wie man aus den Beispielen sehen kann gibt es für diverse Attribute bereits ein Objekt, welches man verwenden kann. Mit Objekten ist es dann einfacher für Analysten den Event Graph und sonstige Korrelationen aufzuzeichnen.
Ein gutes Beispiel ist das Personen Objekt. Vor- und Nachname könnten auch einzeln als Attribute eingetragen werden. Das macht allerdings wenig Sinn, da Vor- und Nachnamen so keinen Bezug aufeinander haben. Ein Objekt hingegen gruppiert diese Attribute, und es ist klar, welche Attribute zusammengehören.

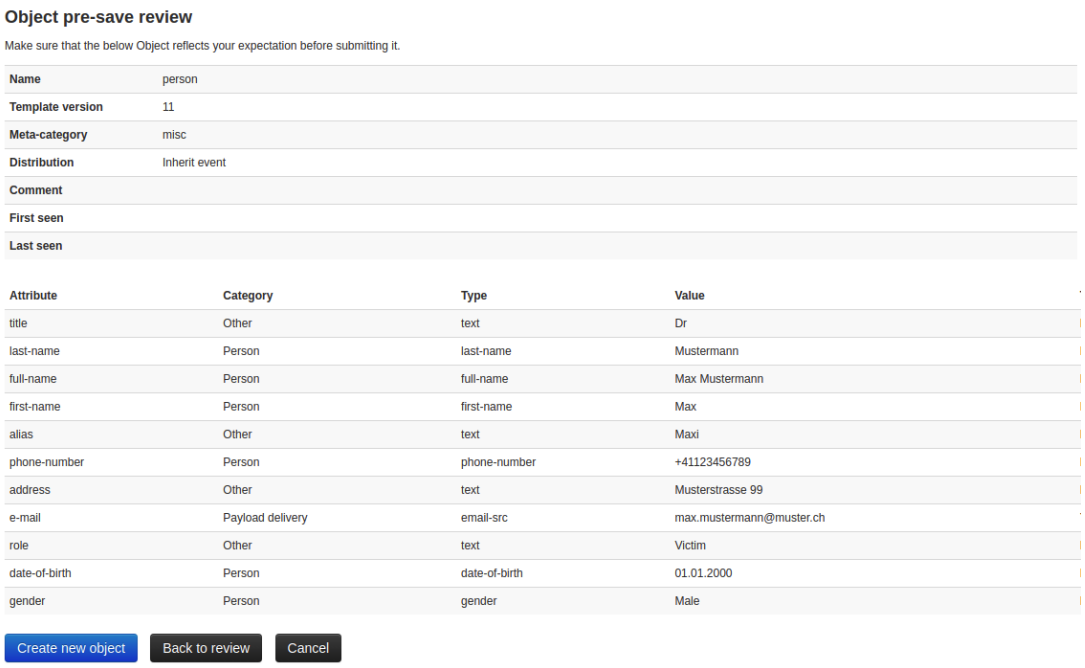

Abbildung 3.4: MISP Objekt erstellen

Weitere Informationen sind hier [7] zu finden.

3.2.6 MISP Sharing model

Ein Hauptfeature von MISP ist es Daten mit anderen Organisation zu teilen. Das Teilen von Informationen kann in MISP auf zwei verschiedene Varianten umgesetzt werden:

1. Das Verwenden von Feeds
2. Synchronisation zwischen MISP Instanzen

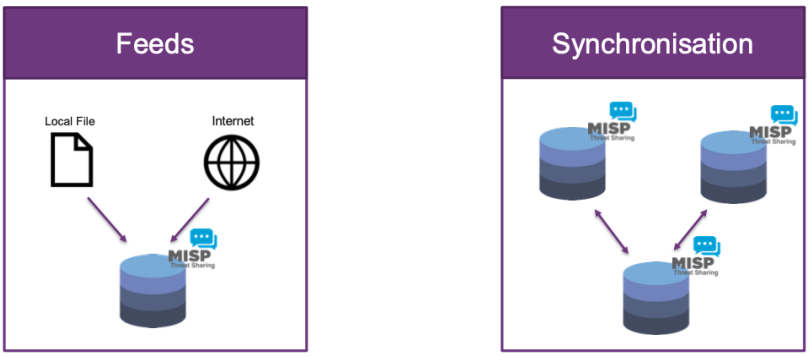

Abbildung 3.5: Sharing Ansätze in MISP

Feeds

Feeds werden hauptsächlich verwendet um Daten, welche aus dem Internet oder einen lokalen Filesystem zu MISP hinzuzufügen. Dabei können die Daten als Text, CSV oder im MISP-Format abgelegt werden. Feeds können einerseits als neue Events gespeichert werden. Andererseits bietet sich an, diese auf einem Redis Server zu cachen. Bei Korrelationen werden so Hinweise im Abschnitt "Feed Hits" angezeigt. Von Grund auf werden dutzende an Feeds von MISP zur Verfügung gestellt. Eigene Feeds können ebenfalls hinzugefügt werden.

Synchronisation

Von einer Synchronisierung spricht man, sobald zwei MISP Server miteinander verbunden wurden und so in eine oder gar beide Richtungen Daten ausgetauscht werden können. Für das Synchronisieren werden fünf Distribution Levels verwendet, welche bestimmen, wie weit das Event oder Attribut in einer Community von MISP Servern geteilt werden.

- Your organisation only
- This community only
- Connected communities
- All communities
- Sharing group

Instanzen können über das Push und/oder Pull Verfahren untereinander Events austauschen. Dabei kann zusätzlich nach Tags und Organisationen gefiltert werden.
Weitere Informationen sind hier[8] zu finden.

3.2.7 Taxonomies

Um in MISP den Überblick nicht zu verlieren können Events mit Tags ausgestattet werden. Diese Tags werden durch Taxonomien zusammengefasst. In einer neuen MISP Instanz muss die Taxonomie Liste zuerst aktualisiert werden. Es können dann einzelne Taxonomien eingeschaltet werden. Man kann sich eine Taxonomie wie eine Gruppe von Tags vorstellen. Eine Taxonomie heisst beispielsweise **tlp**. Dieser enthält Tags der Gruppe **Traffic Light Protocol** welches im Security Umfeld bekannt ist.
Ein Tag kann auf einem Event gesetzt werden, oder auf Attributen selbst. Objekte können nicht mit Tags versehen werden, jedoch die einzelnen Attribute im Objekt schon.
TLP Tags werden oft auf Events eingesetzt, damit beim Sharing an andere Organisationen keine Attribute mit TLP RED verteilt werden.

Weitere Informationen sind hier [9] zu finden.

3.2.8 Galaxies

Eine Galaxie in MISP ist eine Methodik um grosse Objekte, auch genannt Cluster, zu kategorisieren. Diese Objekte können an Events oder Attribute angebracht werden.
Ein Cluster besteht aus einem Key Value Store und kann eins oder mehrere Elemente enthalten.

Galaxies spielen in MISP eine sehr grosse Rolle. Da diese aber auch sehr komplex sind wird in den Übungen nur bedingt eingegangen.
Weitere Informationen sind hier [10] zu finden.

3.2.9 Sightings & Proposals

Auf einer MISP Instanz sammeln sich hunderte, tausende oder noch mehr Events. Die Daten dieser Events werden von Malware Researcher, SOC Mitarbeitern etc. erfasst. Dabei kann es schon mal vorkommen, dass sich ein Fehler einschleicht. Es gibt darum **Sightings** und **Proposals** welche dabei helfen sollen, die Daten so aktuell, relevant und korrekt wie möglich zu halten.
Sightings bieten die Möglichkeit auf Attributebene die Daten als *Sighting* (Daten sind korrekt und valid) oder *False Positive* (Daten sind nicht korrekt) zu identifizieren. Diese Rückmeldungen können visuell einfach verständlich pro Attribut dargestellt.

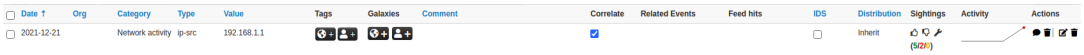

Abbildung 3.6: Sightings

Sightings sind bereits ein guter Indikator, ob die Daten korrekt sind. Proposals werden eingesetzt um die Daten zu verbessern. Zum Beispiel wenn es sich offensichtlich um einen Tippfehler oder Datenfehler handelt. Ein anderer Benutzer kann nun entweder vorschlagen das Attribut zu löschen oder anzupassen.
Im folgenden Beispiel wird vorgeschlagen, dass die IP Adresse eigentlich *192.168.1.2* lautet, und nicht *192.168.1.1*. Der Besitzer dieses Events erhält somit eine Benachrichtigung und kann so den Vorschlag annehmen oder zurückweisen.

Abbildung 3.7: Attribute Proposal

Die beiden beschriebenen Features helfen die allgemeine Qualität der Daten zu verbessern. Durch Proposals können sogar neue Attribute und Objekte als Vorschlag hinzugefügt werden.
Weitere Informationen sind hier [11] zu finden.

3.2.10 Warning Lists

MISP bietet die Möglichkeit **Warninglists** zu aktivieren. Diese Listen helfen dabei, potentielle False Positives in den Daten zu identifizieren.
Standardmässig sind keine der Listen aktiviert. MISP enthält aber momentan (Stand: 23.12.2021) **71 Warninglists**. Um ein paar Beispiele zu nennen:

- List of known Wikimedia address ranges
- List of known URL Shorteners domains
- List of known SMTP sending IP ranges
- Fingerprint of trusted CA certificates

Die Listen enthalten meistens eine Sammlung von IP Adressen, Hostnamen oder Hashes. Die Listen werden auch regelmässig aktualisiert. Es ist auch möglich eigene Warninglists zu erstellen.
Im Beispiel sieht man ein SHA256 Hash, welcher in der Warninglist *Fingerprint of trusted CA certificates* gefunden wurde. Solche Indizien können in der Analyse sehr hilfreich sein.

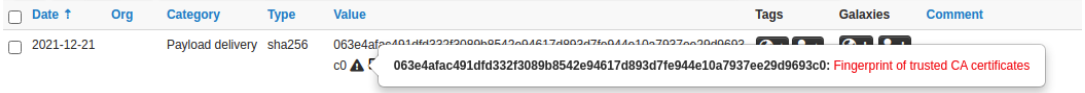

Abbildung 3.8: Warninglist Example

Weitere Informationen sind hier [12] zu finden.

3.2.11 Notice Lists

Die Notice List ist ein eher neues Feature. Circle will damit einen Ort haben um Richtlinien (wie zb GDPR) in Bezug auf Attribute, Objekte und Kategorien abzulegen. Diese Notice Lists soll den Benutzer dann warnen, falls ein Event gegen eine solche Richtlinie verstossen sollte. Zum jetzigen Zeitpunkt (Stand 23.12.2021) findet man dort aber nur die GDPR Regulation.
Die Funktion hat aber viel Potential und wird sicher in der Zukunft noch vermehrt verwendet und erweitert. Weitere Informationen sind hier [13] zu finden.

3.2.12 MISP Modules

MISP Module können die Funktionalität einer MISP Instanz erweitern. Diese Module können in der MISP Instanz Konfiguration noch weiter angepasst werden. Es gibt grundlegend 3 Typen von Modulen.
Die Module sind auch Open Source und sind in Python geschrieben. Dies erlaubt es auch eigene Module für MISP zu schrieben und zu nutzen.

Expansion Module

Diese Module erweitern die Funktionalität indem sie die bereits in Events abgelegten Daten erweitern. Es kann zum Beispiel geprüft werden ob eine Adresse in einen Bitcoin Scam verwickelt ist. Ein anderes Modul zeigt an, aus welchem Land eine IP Adresse stammt.

Export Module

Die Export Module bieten für den Export mehr Optionen an. Ein Event kann zum Beispiel als PDF exportiert werden. Dies konnte MISP standardmässig nicht.

Import Module

Die Import Module können zum Beispiel den Import von Daten aus einem CSV machen. Oder es wird ermöglicht über OCR Daten in MISP zu importieren.

3.2.13 API

MISP bietet eine REST API für die Anbindung von externen Quellen an. Über die Schnittstelle können diverse Daten ausgelesen und auch modifiziert werden.

Die Dokumentation zur Schnittstelle findet sich hier [14]. So können REST Queries mit GET, POST, PUT etc. auf die MISP Instanz abgesetzt werden.
Es besteht auch die Möglichkeit vom Webinterface von MISP Queries zu bauen und auszuführen. Unter **API** → **REST Client** kann man solche Queries erstellen und testen.

Schlussendlich wird die API hauptsächlich für den automatisierten Datenaustausch verwendet. So kann sich beispielsweise eine Firewall oder IDS die nötigen Informationen bei der MISP Instanz abholen. Die Queries können dann so angepasst werden, dass nur relevante Daten zurückgegeben werden.

Die MISP API bietet mit einer guten Dokumentation und sehr grosser Funktionalität eine wichtige Komponente einer MISP Instanz. Externe Komponenten können durch die Schnittstelle von MISP profitieren.

MISP bietet auch eine Python Library, genannt PyMISP. Diese ermöglichst es via Python die MISP API aufzurufen. Informationen zu PyMISP sind Weitere Informationen sind hier [15] zu finden.

3.3 Anforderungsspezifikation

3.3.1 Übersicht

In diesem Unterkapitel wird in aufzählender Form die genaue Anforderungsspezifikation an das MISP Lab definiert. Diese nicht funktionalen Anforderungen werden am Ende der Studienarbeit mittels Nutzertests überprüft und reflektiert.

3.3.2 Anforderungen

Sprache

Die Aufgabenstellung an die Übungsteilnehmer soll im Hacking Lab auf Englisch verfasst werden. Dies ermöglicht auch nicht deutschsprachigen Lernenden das Lab zu absolvieren.

Darstellung

Die Aufgabenstellung soll Grafiken enthalten. Diese sollen als unterstützende Medien dienen und die Sachverhalte in grafischer Form dem Studenten vermitteln. Gute Beispiele dafür sind Screenshots, Diagramme und und sonstige Illustrationen.

Deployment

Das Deployment soll für den Studenten ohne Vorkenntnisse möglich sein. Es besteht die Möglichkeit eine Infrastruktur bei einem Cloud Provider zu nutzen. Wird eine Umgebung lokal benötigt ist die offiziell unterstützte Plattform die Hacking Lab VM.

Performance

Die Lab Umgebung soll für den Studenten in kurzer Zeit bereitstehen. Je nach Deployment (bezogen auf grössere Umgebungen zum Beispiel in der Cloud) dauert die Lab Instanzierung aber auch bis zu einer Stunde. Dies bedeutet aber, dass der Student frühzeitig informiert werden muss.

Zeitrahmen

Das MISP Lab muss in mehrere kleine Aufgaben unterteilt werden, sodass die Studenten fortlaufend kleine Lernerfolge erzielt. Eine Aufgabe sollte daher in maximal 45 Minuten absolviert werden können.

Lernkontrolle

Nach dem erfolgreichen Absolvieren einer Aufgabenstellung müssen die Lernziele messbar geprüft werden können. Dies kann mittels PDF Bericht oder Flag vom Übungsteilnehmer eingereicht und vom Übungsbetreuer kontrolliert werden.

Zielgruppe

Die Zielgruppe der Labs sind Studierende oder Fachkräfte im Bereich Informatik. Ein Grundwissen in diesem Bereich wird daher vorausgesetzt. Die Labs müssen jedoch so aufgebaut werden, dass diese auch ohne vertieftes Wissen im Cyber Security- oder Infrastruktur-Bereich gelöst werden können.

3.4 Systemübersicht

3.4.1 Übersicht

Für das MISP Lab werden verschiedene Arten von Deployments verwendet, um die Übungen zu absolvieren. In diesem Unterkapitel werden die einzelnen Setups aufgelistet und dokumentiert. Die unterschiedlichen Setups wurden so gewählt, dass einerseits für den Studierenden möglichst viele verschiedene Aufgabentypen bereit gestellt werden können, andererseits keine zu hohen Kosten durch den Cloud Provider generiert werden.
Die unterschiedlichen Setups werden mit Vor- und Nachteilen evaluiert und beschrieben.

3.4.2 Setup 1

Ziel dieses Setup ist es, dem Studenten eine voll funktionale MISP Instanz lokal zur Verfügung zu stellen. Dazu muss Docker und docker-compose auf dem System installiert sein. Das offiziell unterstützte System dieses Setups is die Hacking Lab Live CD (Virtuelle Maschine). Für diese Umgebung verwenden wir das von MISP betreute Github Repository [16].

Vorteile

- schnelle, lokale Installation
- vielfältige Anpassungsmöglichkeiten im Image
- keine anfallenden Kosten für die Hochschule

Nachteile

- benötigt docker und docker-compose
- benötigt lokale Ressourcen (Arbeitsspeicher, CPU, ...)
- keine öffentliche IP Adresse

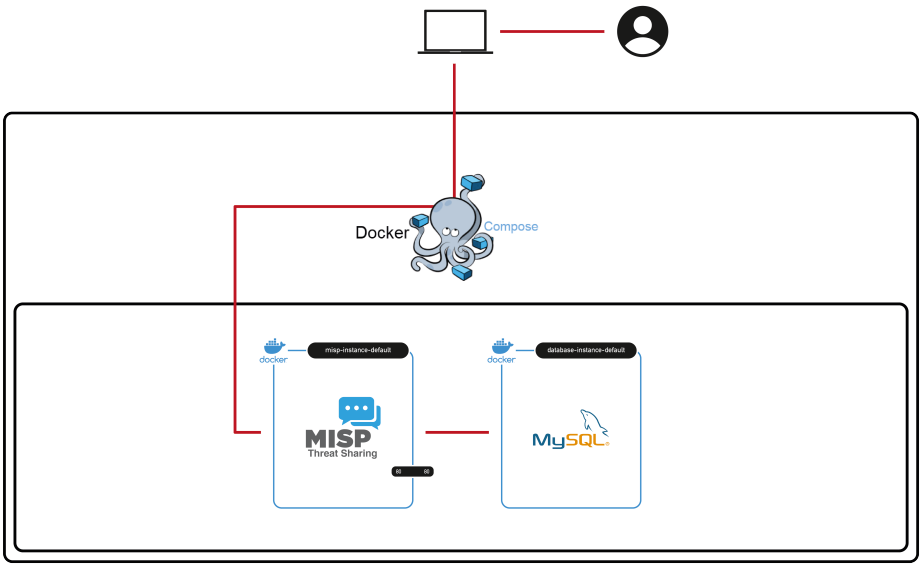

Abbildung 3.9: Systemübersicht Setup 1

3.4.3 Setup 2

Um das Deployment noch einfacher zu gestalten wird in diesem Setup der Docker Container mit MISP in der Hacking Lab Infrastruktur laufen gelassen. Der Container kann vom Studenten direkt im Hacking Lab gestartet werden. Im Hintergrund wird dann ein docker-compose hochgefahren und dem Studenten dann zur Verfügung gestellt. Dieser Container enthält auch eine komplett lauffähige MISP Instanz.

Vorteile

- einfaches Deployment für Studenten (One Click)
- vielfältige Anpassungsmöglichkeiten im Image

Nachteile

- Ressourcen der Hochschule werden beansprucht
- Unbekannte Funktionsweise des Hacking-Labs

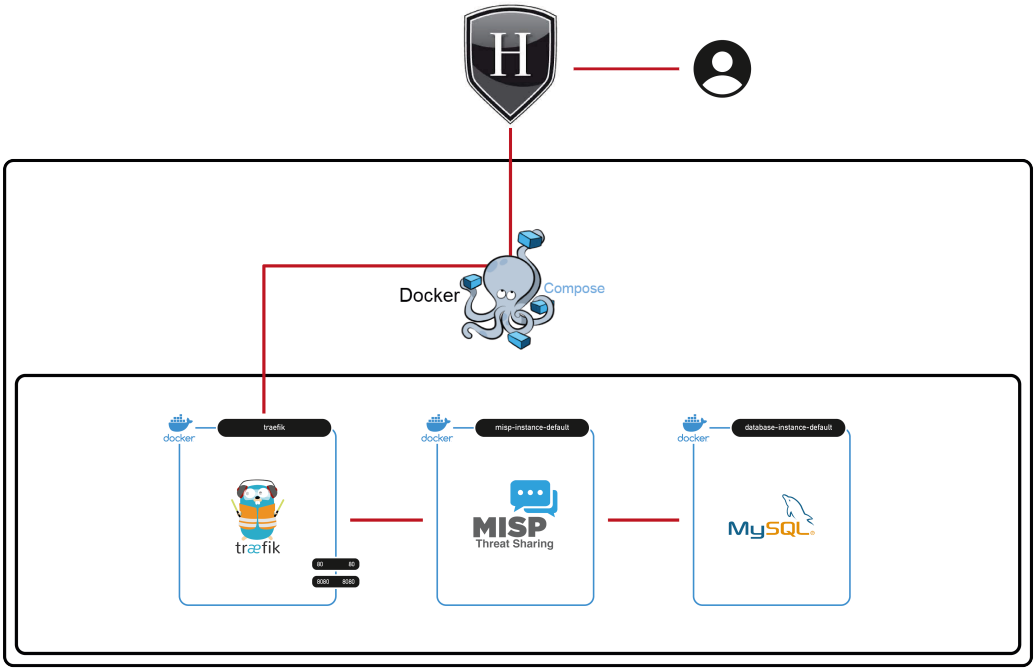

Abbildung 3.10: Systemübersicht Setup 2

3.4.4 Setup 3

Wenn mehr Rechenleistung und eine grössere Umgebung mit mehreren virtuellen Maschinen benötigt ist, kommt das dritte Setup zum Zug. Der Student kann über das Hacking Lab ein Terraform Skript anstossen, welches im Hintergrund eine Umgebung in Azure instanziert. Diese Umgebung kann vom Author des Labs beliebig angepasst werden. Je nach Labs gibt es unterschiedliche virtuelle Maschinen.
Der Student kann dann über einen Jump Host auf die MISP Instanz (oder andere Services in der Umgebung) zugreifen.

Vorteile

- einfaches Deployment für Studenten (One Click)
- öffentliche IP Adressen verfügbar
- grosse Umgebungen möglich

Nachteile

- laufend anfallende Kosten
- Deployment dauert lange bis nutzbar

Nutzbarkeit dieses Setups

Die Komplexität dieser Variante steigt mit der Anzahl involvierter Komponenten stark. Da durch das Setup 1 alle Labs abgedeckt werden, wird diese Variante in unserer Arbeit nicht verwendet.

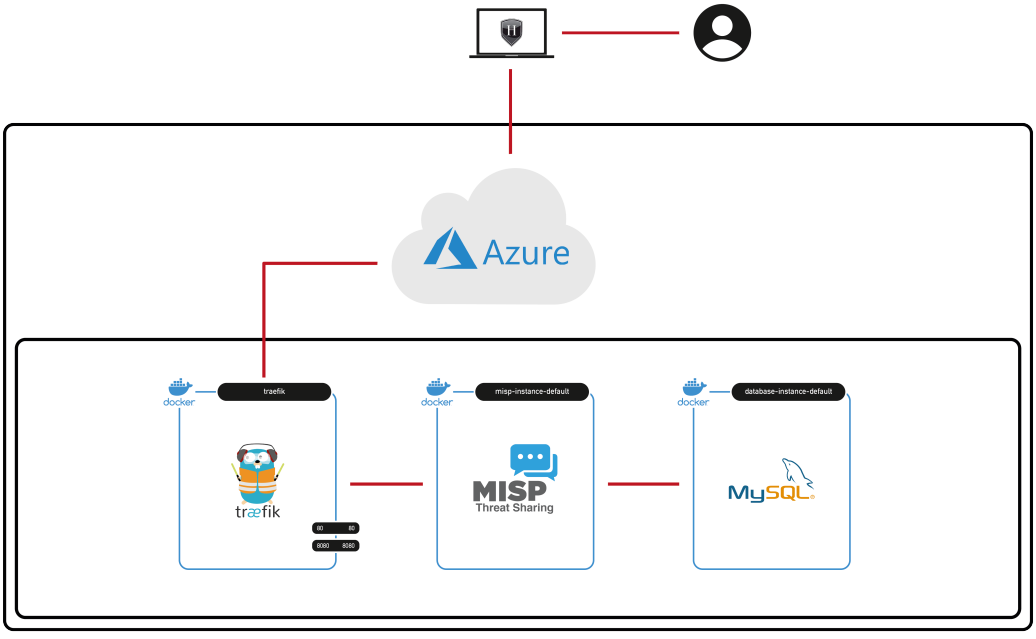

Abbildung 3.11: Systemübersicht Setup 3

3.4.5 Systemübersicht effektiv (Setup 1 mit Anpassungen)

In der späteren Entwicklung der Labs hat man sich dazu entschieden, nur das Setup 1 zu verwenden. Grund dafür war, dass dieses Setup eine schnelle und simple Konfiguration für den Studenten bedeutet. Zusätzlich ist die Docker Installation so sehr anpassungsfähig und läuft zuverlässig. Setup 2 und 3 wurden zur Vollständigkeit in der Dokumentation nicht entfernt.
Das Setup 1 wurde noch ein wenig ausgebaut und angepasst und stellt mithilfe von einem Reverse Proxy (Traefik) vier verschiedene MISP Instanzen zur Verfügung. Die vier MISP Instanzen sind unterschiedlich konfiguriert.

Vorteile

- schnelle, lokale Installation
- vielfältige Anpassungsmöglichkeiten im Image
- keine anfallenden Kosten für die Hochschule
- viele Möglichkeiten für Übungen, da 4 MISP Instanzen zur Verfügung stehen

Nachteile

- benötigt docker und docker-compose
- benötigt lokale Ressourcen (Arbeitsspeicher, CPU, ...)
- keine öffentliche IP Adresse

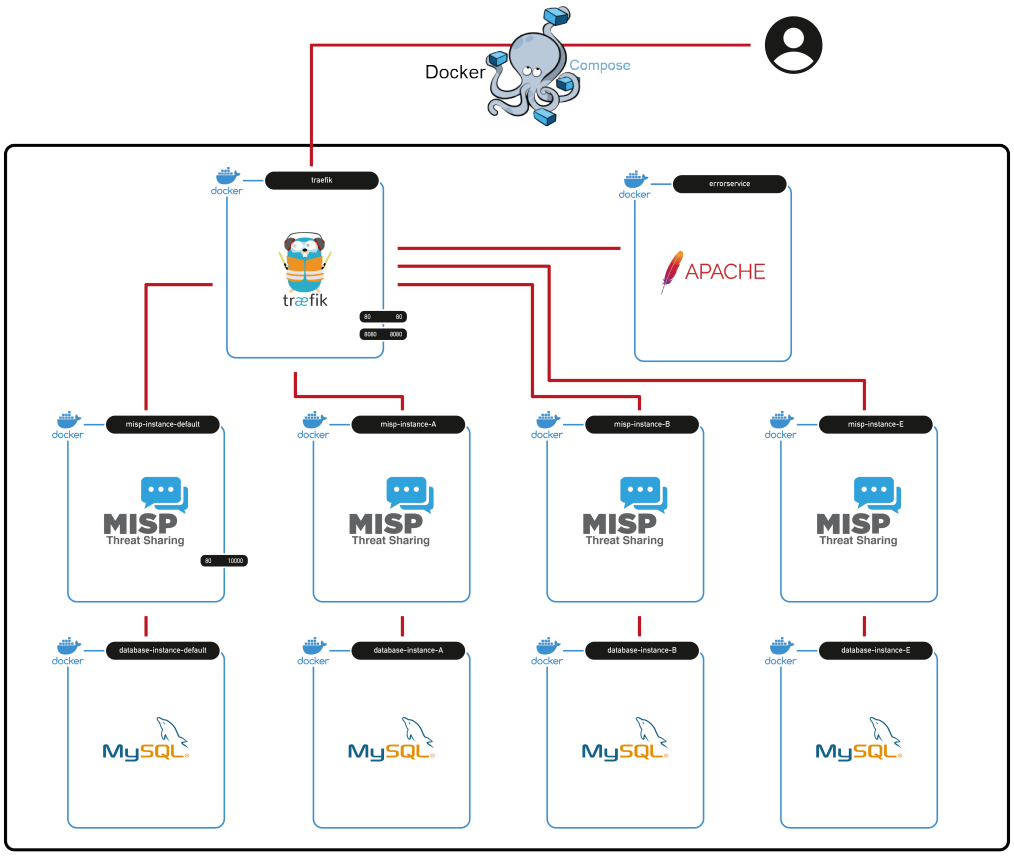

Abbildung 3.12: Systemübersicht Setup 1 mit Anpassungen

3.5 Implementation

3.5.1 Übersicht

In diesem Abschnitt wird dokumentiert, wie die Übungsaufgaben implementiert wurden und welche Tools dafür eingesetzt wurden. In einem ersten Schritt wird die Vorgehensweise definiert, wie die Labs implementiert wurden. Anschliessend werden die technischen Komponenten vertieft aufgelistet und dokumentiert. Die Übungen selbst werden im nächsten Abschnitt *Dokumenation der Übungen* detailliert beschrieben.

3.5.2 Abläufe

Die Übungsumgebungen wurden immer im selben Schema implementiert. Nachfolgend werden diese Schritte aufgelistet.

Stories definieren

Wie dem *Projektplan* zu entnehmen ist, wurden nach einem intensiven Research Teil in der vierten Iteration Stories definiert. Die Stories sollten einen Themenbereich und mögliche Aufgabentypen aufzeigen, an welchen sich später die Labs orientieren sollten. Es wurden bewusst mehr Stories definiert, als im Projektplan zur definitiven implementation angedacht war. Dies ermöglichte dem Team bei Problemen eine andere Story zu implementieren. Alle Stories sind im Abschnitt *Stories* beschrieben.

Erstellen einer Lab Umgebung

In der fünften Iteration wurde die Übungsumgebung aufgebaut. Diese bestand zunächst nur aus einem einzelnen Docker Container. In jeder darauffolgenden Iteration wurde diese stetig auf die Bedürfnisse der Labs angepasst und verbessert.

Implementation der Labs

Die Implementation der Labs erstreckte sich von der sechsten bis neunten Iteration. Ziel war es, dass durchschnittlich pro Iteration und Teammitglied ein neues Lab erstellt wird.

Zunächst wurde immer mit einem kurzen Research Teil gestartet und versucht, die zuvor definierten Stories direkt in MISP umzusetzen. War dies nicht möglich, so musste erneut Recherche betrieben werden, bis die Übung umgesetzt werden konnte. Falls das Lab trotzdem nicht durchgeführt werden konnte, musste von der Story abgewichen und der Themenbereich angepasst werden.

Im weiteren Verlauf der Iteration wurden die Texte erstellt werden, welche die Übungsteilnehmer während den Übungen lesen sollten. Dabei wurde darauf geachtet, dass die Texte verständlich sind und die Studierenden beim Lösen bestmöglich unterstützt. Des weiteren wurden je nach Lab Demo-Daten erstellt, welche durch das Konfigurationsskript beim ersten Start des Labs in MISP hinzugefügt werden.

Nach erfolgreichem Abschliessen des Labs wurden diese unter den Teammitgliedern ausgetauscht und getestet. Dabei wurden einerseits die Struktur der Aufgabenstellung sowie die Arbeiten in MISP selbst getestet. Ebenfalls wurden die Labs durch Externe getestet. Es stellte sich jedoch heraus, dass es sehr schwierig war Studierende zu finden, welche jede Woche neben der eigenen Studienarbeit zwei Übungsaufgaben lösen und bewerten sollten. Weitere Informationen zum Testen der Labs und das dabei erhaltene Feedback ist im Abschnitt *Testing* zu finden.

Abschluss

Alle Labs wurden in Iteration 12 erneut durchgetestet. Durch das Weiterentwickeln der Lab Infrastruktur gab es einige Übungen, welche nicht mehr zuverlässig funktionieren. Hauptsächlich mussten Änderungen in den Übungstexten vorgenommen werden. In gewissen Labs musste das Deployment weiter verfeinert werden, damit diese wieder gelöst werden konnten.

3.5.3 Konfiguration der Instanzen

Ziel dieser Arbeit war es, eine Übungsumgebung zu entwickeln, welche Studierende beim Kennenlernen von MISP unterstützt. Dabei sollten administrative Arbeiten einen möglichst kleinen Teil ausmachen. Die Studierenden sollten sich nach der Installation nicht mit Einstellungen und Konfigurationen der Lab Umgebung beschäftigen. Ebenfalls sollten Aufgaben nicht ständig wiederholt werden müssen. Beispiel dafür ist das Erstellen eines Events. Es wurden dafür zwei möglich Lösungsansätze verfolgt:

- Initialisierung während der Installation direkt in die Datenbank
- Kommunikation zu Laufzeit über die MISP API

Schlussendlich haben beide Varianten ihre Vor- und Nachteile. Wir entschieden uns jedoch, die Demo Daten und Konfigurationen über die MISP API hinzuzufügen. Hauptgrund für unsere Entscheidung war, dass weitere Labs zu einem späteren Zeitpunkt schneller und effizienter hinzugefügt werden können.

Skript

MISP stellt eine REST API zur verfügung, welche mittels GET oder POST Requests angesprochen werden kann. Dies ermöglicht uns, Daten an MISP automatisiert zu senden. Die REST Schnittstelle könnte direkt mit curl Befehlen in einem Shell Script angesprochen werden. Wir entschieden uns jedoch, stattdessen die pyMISP Bibliothek für Python3 zu verwenden. Dies ermöglicht uns verständlichen Code zu schreiben, welcher wiederverwendet werden kann.

Das Konfigurationsskript besteht aus mehreren Dateien. Alle Daten sind ebenfalls im GitHub Repository [17] einsehbar. Beim Start wird die `configuration.py` Datei im Docker Container ausgeführt. Das Programm versucht als erstes den API Key des Administrators zu speichern, da dieser zur Authentisierung benötigt wird. Um den API Key des Administrators zu erlangen, müssen zwei Shell Befehle ausgeführt werden. Der Key wird jedoch erst im zweiten Befehl zurückgegeben.

```
19 def get_api_key(email):
20     subprocess.getoutput(
21         "/var/www/MISP/app/Console/cake Admin change_authkey " + email + " | sed '1d'")
22     misp_key = subprocess.getoutput(
23         "/var/www/MISP/app/Console/cake user change_authkey " + email + " | cut -d ':' -f 2 | cut
               -d ' ' -f 2")
24     return misp_key
```

Listing 3.1: Abfrage API Key - configuration.py

In einem nächsten Schritt wird versucht mit dem erlangten API Key eine Verbindung zu MISP herzustellen. Es kann jedoch sein, dass das Pythonskript gestartet wird, bevor die Installation von MISP selbst abgeschlossen ist. Daher wird so lange erneut versucht eine Verbindung zu MISP herzustellen, bis diese erfolgreich ist.

Nachdem eine erfolgreiche Verbindung hergestellt wurde werden diverse Einstellungen auf dem MISP Server gesetzt, welche für die Übungen benötigt werden. In einem letzten Schritt werden die Konfigurationen und Demo Daten für die Labs zu MISP hinzugefügt. Um wiederverwendbaren Code zu schreiben wurde dafür eine Klasse erstellt, welche alle gängigen Funktionen für das Erstellen einer neuen Challenge abbildet.

Klasse Lab

Idee ist es, dass für jedes Lab eine neue Instanz eines Lab Objektes erstellt wird. Dafür können bei der Instanziierung einige Parameter bereits mitgegeben werden. Einerseits muss immer die Nummer der Challenge angegeben werden sowie der API Key, welcher beim Erstellen verwendet werden soll. Anderseits kann optional angegeben werden, auf welchen Server Instanzen die Konfiguration ausgeführt werden soll.

```
11 class Lab:
12     default_password = 'compass'
13
14     def __init__(self, lab_nr: int, api: PyMISP, instance: Instance = Instance.all):
15         self.__lab_nr = lab_nr
16         self.__api = api
17         self.__instance = instance
18         self.__users = []
19         self.__admins = []
```

Listing 3.2: Ausschnitt des Konstruktors der Lab Klasse - lab.py

Nach der Instanziierung können neue Organisationen, Nutzer oder Daten einfach für jedes Lab erstellt werden. Dies kann am besten anhand eines Beispiels erklärt werden:

Die Challenge 5 besteht aus einer Organisation, einem Organisations Administrator und einem Investigator Benutzer, welcher von den Studierenden für das Lösen verwendet wird. Das Lab soll auf der MISP Instanz A gelöst werden. Ebenfalls sollen aus einer JSON Datei vorbereitete Events und alle dazugehörigen Daten mit dem Organisations Administrator ausgelesen und zu MISP hinzugefügt werden.

Als Ersteller des Labs kann die gewünschte Konfiguration einfach direkt in der `configuration.py` Datei erstellt werden. Das Skript fügt selbstständig diverse Attribute wie E-Mail Adressen, Namen, Rechte, UUID's, etc. hinzu.

```
89 lab5 = Lab(5, misp, Instance.A)
90 lab5.add_org()
91 lab5.add_user(Role.org_admin)
92 lab5.add_user(Role.investigator)
93 lab5.import_events(lab5.get_admin())
```

Listing 3.3: Erstellung des Lab 5 - configuration.py

Environment Variablen

Um verschiedene Instanzen mit nur einem Dockerfile zu betreiben wurden unterschiedliche Lösungswege diskutiert. Der Docker Container muss zur Laufzeit erkennen, um welchen MISP Server es sich handelt und anschliessend entscheiden, welche Konfigurationsskripte ausgeführt werden müssen und welche nicht. Schlussendlich entschieden wir uns, in der `docker-compose.yml` Datei Environment Variablen hinzuzufügen, welche vom Skript ausgelesen werden. Die Environment Variablen werden in einer `.env` Datei gesetzt und können dort bei Bedarf angepasst werden.

3.5.4 CI/CD

Da der Build des Docker Images viel Zeit und Ressourcen beansprucht, hat man sich dazu entschieden, diesen Prozess zu automatisieren. Dies wurde mit Github Actions direkt im selben Repository umgesetzt. Dafür wurden zwei verschiedene Workflows angelegt.

latest Tag

`build-latest.yml` baut das MISP Docker Image gemäss Dockerfile und pusht dieses dann auf das Docker Hub Repository. Auf diesem Workflow wird dann der Tag `latest` und die Github Variable `RUN_NUMBER` gesetzt. `RUN_NUMBER` gibt die ID der Ausführung des Workflows von Github an. So kann nachgeschaut werden, welcher Tag zu welchem Commit gehört.

dev Tag

Damit Änderungen auch getestet werden können wurde der `dev` Tag eingeführt. Dieser wird durch den Workflow `build-dev.yml` erstellt und auf Docker Hub gepusht. Der Ablauf des Workflows ist sonst identisch.

Allgemein

Für die Authentisierung bei Docker Hub wurden Secrets von Github verwenden. Diese können innerhalb des Workflows abgerufen werden, jedoch im Repository nicht eingesehen werden. In den Secrets abgelegt wurde der Username `secrets.DOCKER_HUB_CI_CD_PIPELINE_MISP_USERNAME` und der dazugehörige API Key `secrets.DOCKER_HUB_CI_CD_PIPELINE_MISP_API_KEY`.

Die Workflows werden ausserdem nur bei relevanten Änderungen im Repository ausgeführt. Dies wurde mit dem `paths` Argument beschränkt. So wird der Workflow nur bei Änderungen in den Verzeichnissen `misp/*` und `data-shared/*` ausgeführt.

Der folgende Ausschnitt aus der Pipeline Konfigurationsdatei zeigt, welche Variablen für allfällige Anpassungen verändert werden müssten. In der aktuellen Konfiguration kann der Name des Docker Images, der Name des Dockerhub Repository's und die Authentifizierungsdaten hier angepasst werden.

```
11 env:
12     dockerhub_image_name: misp
13     dockerhub_repository_name: hackinglab
14     dockerhub_username: ${{ secrets.DOCKER_HUB_CI_CD_PIPELINE_MISP_USERNAME }}
15     dockerhub_token: ${{ secrets.DOCKER_HUB_CI_CD_PIPELINE_MISP_API_KEY }}
```

Listing 3.4: build-latest.yml

3.5.5 Docker-Compose Konfiguration

Damit die Installation der Lab Umgebung für den Studenten so simpel wie möglich ist, wurde mit einer docker-compose Datei gearbeitet. Die Datei hat knapp 250 Zeilen, darum werden nur die relevante Ausschnitte in der Dokumentation erklärt.
Total beinhaltet die Installation 10 Container.

Die unten beschrieben Datei kann hier [18] abgerufen werden.

MISP Instanz: MISP

Der MISP Container wird direkt vom offiziellen MISP Docker Hub Repository geholt. Der Container lässt sich über TCP Port 80 ansprechen. Da aber mehrere Instanzen davon laufen müssen, werden mehrere Netzwerke und einen Reverse Proxy benötigt. Mehr dazu später.

Damit die Daten persistent gespeichert werden können, muss erstens auf die Datenbank zugegriffen werden können und zweitens via Volume Daten abgelegt werden. Hier sind zwei Volumes hinterlegt. Das Volume **shared-data** wird für ein MISP Expansion Modul verwendet, welches noch eine externe Datenbank benötigt.

Über Umgebungsvariablen werden der Instanz Informationen zur Datenbank hinterlegt. Zusätzlich gibt es noch ein einige Parameter wie zum Beispiel Zeitzone und das Passwort des Admins welche mitgegeben werden.
Mit dem **depends on** Attribut stellt Docker sicher, dass die Datenbank vollständig initialisiert ist und erst dann die MISP Instanz gestartet wird.
Die Labels werden für den Traefik Reverse Proxy benötigt. Diese werden weiter unten genauer behandelt.
Zum Schluss wird hier noch ein Port Mapping Konfiguriert. Für ein Lab muss der Reverse Proxy aus technischen Gründen umgangen werden. Die **default** MISP Instanz ist somit auch auf dem TCP Port **10000** erreichbar.

```
misp-instance-default:
    image: hackinglab/misp:latest
    container_name: misp-instance-default
    restart: unless-stopped
    depends_on:
      - database-instance-default
    networks:
      - internal-all-instances
      - internal-instance-default
    volumes:
      - ./data-instance-default/misp:/var/www/MISP:rw
      - ./data-shared:/data-shared:rw
    environment:
      - MYSQL_HOST=${MYSQL_HOST_DEFAULT}
      - MYSQL_DATABASE=${MYSQL_DATABASE}
      - MYSQL_USER=${MYSQL_USER}
      - MYSQL_PASSWORD=${MYSQL_PASSWORD}
      - MISP_ADMIN_EMAIL=${MISP_ADMIN_EMAIL}
      - MISP_ADMIN_PASSPHRASE=${MISP_ADMIN_PASSPHRASE}
      - INSTANCE_TAG=${INSTANCE_DEFAULT_TAG}
      - POSTFIX_RELAY_HOST=${POSTFIX_RELAY_HOST}
      - TIMEZONE=${TIMEZONE}
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.misp-default.rule=Host(‘${INSTANCE_DEFAULT_URL}‘)"
      - "traefik.http.routers.misp-default.middlewares=test-errorpages"
      - "traefik.http.middlewares.test-errorpages.errors.status=500-599"
      - "traefik.http.middlewares.test-errorpages.errors.service=errorservice"
      - "traefik.http.middlewares.test-errorpages.errors.query=/inprogress.html"
    ports:
      - "10000:80"
```

Listing 3.5: docker-compose.yml (Ausschnitt)

Datenbank: MySQL

MISP benötigt ebenfalls auch eine Datenbank um Events, Tags, Galaxies etc. abzuspeichern. Jede MISP Instanz nutzt eine eigene Datenbank.
Damit die Daten persistent abgelegt werden können, wird ein Volume verwendet. Dieses legt die Daten im docker-compose Verzeichnis ab, wie auf Zeile 175 zu sehen ist.
Mit Umgebungsvariablen, welche in einer **.env** Datei abgelegt sind, geben den Namen, Benutzer und Passwörter für die Datenbank vor.

```
168 database-instance-default:
169     image: mysql/mysql-server:5.7
170     container_name: database-instance-default
171     restart: unless-stopped
172     networks:
173       - internal-instance-default
174     volumes:
175       - ./data-instance-default/database:/var/lib/mysql:rw
176     environment:
177       - MYSQL_DATABASE=${MYSQL_DATABASE}
178       - MYSQL_USER=${MYSQL_USER}
179       - MYSQL_PASSWORD=${MYSQL_PASSWORD}
180       - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
```

Listing 3.6: docker-compose.yml (Ausschnitt)

Reverse Proxy: Traefik

Traefik wird als Reverse Proxy eingesetzt. Es wird dadurch ermöglicht vier MISP Instanzen gleichzeitig zu starten. Die Instanzen lassen sich unter unterschiedlichen Hostnamen erreichen.
Der folgende Ausschnitt zeigt die Konfiguration auf dem Traefik Container selbst. Damit so wenig Logs wie möglich angezeigt werden wird das Log Level auf **FATAL** eingestellt. Traefik bietet auf dem TCP Port **8080** ein Dashboard an, um die Konfiguration und den Status zu überprüfen. Dieses wird mit **api.insecure=true** aktiviert.
Da der Provider Docker ist muss das auch definiert werden.
providers.docker.exposedbydefault=false bewirkt, dass auf jedem Container eine Traefik Konfiguration explizit das Routing einschalten muss.
entrypoints.web.address=:80 konfiguriert, dass der Reverse Proxy auf den TCP Port **80** reagiert.

Damit sich Traefik auf den Docker Service verbinden kann (um den aktuellen Status der anderen Container zu prüfen) muss auch das konfigurierte Volume eingerichtet sein.

```
5     traefik:
6     image: "traefik:latest"
7     container_name: "traefik"
8     command:
9       - "--log.level=FATAL"
10      - "--api.insecure=true"
11      - "--providers.docker=true"
12      - "--providers.docker.exposedbydefault=false"
13      - "--entrypoints.web.address=:80"
14
15    networks:
16      - internal-all-instances
17      - internal-instance-default
18      - internal-instance-A
19      - internal-instance-B
20      - internal-instance-E
21    ports:
22      - "80:80"
23      - "8080:8080"
24    volumes:
25      - "/var/run/docker.sock:/var/run/docker.sock:ro"
```

Listing 3.7: docker-compose.yml (Ausschnitt)

Es folgt noch die Konfiguration der Traefik Labels auf einem Container (hier misp-instance-default). Die Labels schalten Traefik auf dem erwähnten Container ein und definieren den Hostnamen via eine Umgebungsvariable. Es wird zusätzlich auch noch eine Middleware aktiviert. Diese verhindert, dass auf den Container zugegriffen kann, solange ein Error Code 500 bis 599 abgefangen wird. Falls dies zutrifft, wird auf den Container **error-service** die Webseite **inprogress.html** angezeigt.

```
65 labels:
66     - "traefik.enable=true"
67     - "traefik.http.routers.misp-default.rule=Host(‘${INSTANCE_DEFAULT_URL}‘)"
68     - "traefik.http.routers.misp-default.middlewares=test-errorpages"
69     - "traefik.http.middlewares.test-errorpages.errors.status=500-599"
70     - "traefik.http.middlewares.test-errorpages.errors.service=errorservice"
71     - "traefik.http.middlewares.test-errorpages.errors.query=/inprogress.html"
```

Listing 3.8: docker-compose.yml (Ausschnitt)

Error Service: Apache

Dieser Service ist sehr simpel. Er zeigt eine einzige Webseite an. Diese wird aufgerufen wenn Traefik von einem Container einen Fehlercode bekommt und die Error Middleware aktiviert ist. Dieser Container muss beim ersten Start von docker-compose zuerst noch gebaut werden. Dies geschieht automatisiert und dauert nur wenige Sekunden.
Der Webserver läuft mit einem Apache Image von Dockerhub.

```
27     errorservice:
28     build:
29       context: ./misp/traefik-custom-error-pages
30     networks:
31       - internal-all-instances
32       - internal-instance-default
33       - internal-instance-A
34       - internal-instance-B
35       - internal-instance-E
36     labels:
37           - "traefik.enable=true"
38           - "traefik.http.routers.error.rule=Host(‘error.localhost‘)"
39           - "traefik.http.routers.error.service=errorservice"
40           - "traefik.http.services.errorservice.loadbalancer.server.port=80"
41           - "traefik.http.routers.error.entrypoints=web"
```

Listing 3.9: docker-compose.yml (Ausschnitt)

Docker Netzwerke

Damit sich die verschiedenen Container nicht ungewollt beeinflussen wurden im docker-compose diverse Netzwerke eingerichtet. Jede MISP Instanz und sowie die dazugehörige Datenbank hat ihr eigenes Netzwerk. Damit die MISP Instanzen auch untereinander kommunizieren können (zb. für Event Synchronisierungen) regelt das Netzwerk **internal-all-instances** diesen Verkehr.
Internal ist bei allen Netzwerken auf **False**. Das bedeutet, dass man aus diesem Netzwerk auch auf externe Server (Internet) zugreifen kann.

```
networks:
  internal-all-instances:
    name: internal-all-instances
    driver: bridge
    internal: false
  internal-instance-default:
    name: internal-instance-default
    driver: bridge
    internal: false
  internal-instance-A:
    name: internal-instance-A
    driver: bridge
    internal: false
  internal-instance-B:
    name: internal-instance-B
    driver: bridge
    internal: false
  internal-instance-E:
    name: internal-instance-E
    driver: bridge
    internal: false
```

Listing 3.10: docker-compose.yml (Ausschnitt)

3.5.6 Installation der Labs

Die Installation der Labs soll gemäss Anforderungen sehr simpel aber effektiv sein. Als offizielle Umgebung wird die Hacking Lab Live CD unterstützt. Die benötigten Docker Komponenten sind bereits installiert.

Als erster Schritt muss das Repository von Github geklont werden. Dieses ist öffentlich zugänglich und kann hier eingesehen werden.

```
cd /home/hacker/
git clone https://github.com/Hacking-Lab/misp-docker-image.git
```

Listing 3.11: Clone Github MISP Repository

Das Repository selbst ist knapp 80 Megabytes gross, jedoch müssen nun im folgenden Schritt noch Images von mehreren Gigabytes heruntergeladen werden. Die Images werden direkt von hub.docker.com heruntergeladen. Die Installation kann aus diesem Grund mehrere Minuten dauern.
Mit **docker-compose up** wird aus der **docker-compose.yml** Datei entnommen welche Container wie konfiguriert werden. Total werden nun 10 Docker Container hochgefahren. Über einen Reverse Proxy verbindet sich der User dann mit den verschiedenen Instanzen. Mehr dazu kann im Kapitel *Docker-Compose Konfiguration* gefunden werden.

```
cd /home/hacker/misp-docker-image
docker-compose up
```

Listing 3.12: Start docker-compose

Wenn der Benutzer versucht auf die MISP Instanz zuzugreifen bevor die Konfiguration abgeschlossen wurde, wird ein **Loading Screen** angezeigt. Dem Benutzer stehen für verschiedene Labs unterschiedliche MISP Instanzen zur Verfügung. In den meisten Labs wird die Default Instanz verwendet. Die Instanzen unterscheiden sich in der Konfiguration und vorbereiteten Daten.

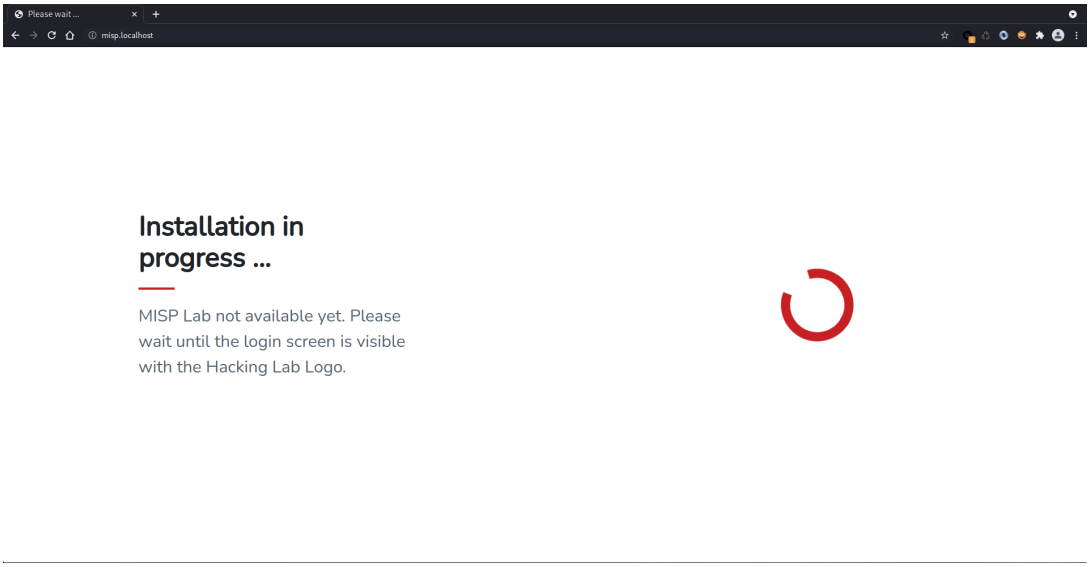

Abbildung 3.13: Installation Loading Screen

Sobald die grundlegende Konfiguration abgeschlossen wurde, wird eine weitere **Loading Page** angezeigt. Diese vermittelt dem Benutzer nun, dass die Konfiguration einen Schritt weiter ist und nun nur noch die Daten und Konfiguration der MISP Instanz im Hintergrund läuft. Da auch die Benutzer in diesem Schritt angelegt werden, kann der Benutzer sich erst anmelden nachdem auch dieser Schritt abgeschlossen wurde.

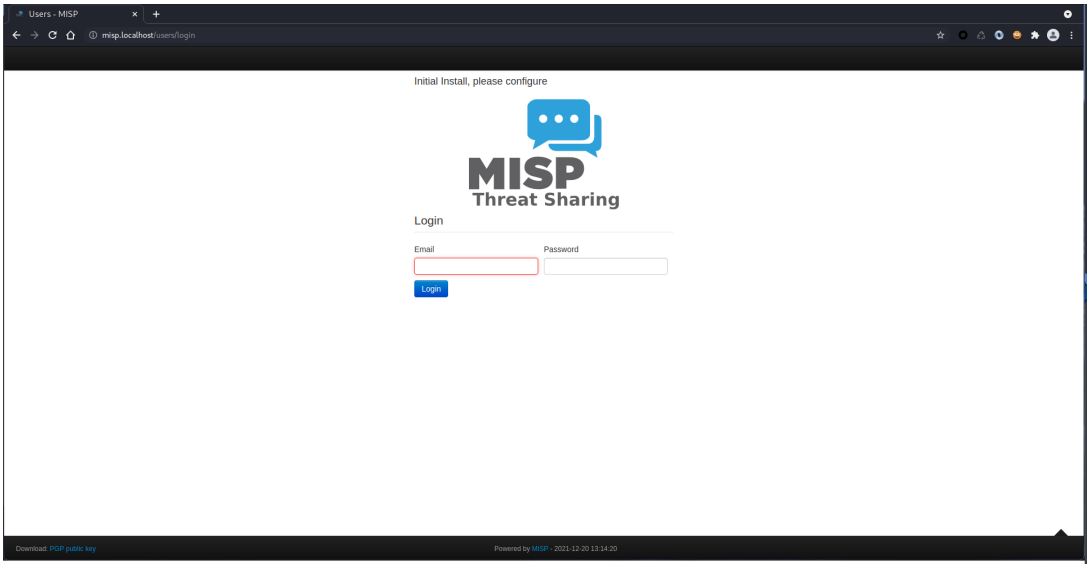

Abbildung 3.14: Installation Loading Screen 2

Sobald dann das Hacking Lab Logo auf der Startseite erscheint ist die Konfiguration vollständig abgeschlossen. Der Benutzer kann sich nun anmelden. In den Labs wird immer vorgegeben welcher Benutzer verwendet werden soll. Der User **admin@misp-lab.com** mit dem Passwort **compass** ist jedoch auf allen Instanzen verfügbar. Das Passwort ist auch bei den anderen Benutzern das selbe.

Abbildung 3.15: Installation complete

Die Instanzen sind verfügbar unter

- http://misp.localhost
- http://instance-a.misp.localhost
- http://instance-b.misp.localhost
- http://instance-e.misp.localhost

Mit **CTRL + C** (in der Shell) kann dann das docker-compose wieder gestoppt werden. Um alle erstellten Netzwerke und Container wieder zu entfernen kann mit **docker-compose down** die Umgebung aufgeräumt werden. Die Daten bleiben aber mit beiden Optionen persistent. Um auch die Daten zu löschen wird am einfachsten der ganze Order **misp-docker-image** gelöscht und nochmals von git geklont.
Das Starten der Lab Umgebung mit **docker-compose up** wird nun um einiges schneller gehen, da alle Images bereits lokal im Cache sind und die Konfiguration der Instanzen auch bereits vorgenommen wurde.

3.6 Dokumenation der Übungen

3.6.1 Übersicht

In diesem Unterkapitel wird die Aufgabenstellung an die Übungsteilnehmer definiert. In einem ersten Schritt werden Stories definiert, wie ein Student die Labs aus seiner Perspektive sieht. In einem zweiten Schritt wird die effektive Aufgabenstellung niedergeschrieben. Die Labs werden nach verschiedenen Deployments unterschieden. Eine genaue Beschreibung zu den Lab Umgebungen ist im Kapitel *Systemübersicht* definiert.

3.6.2 Stories

Einleitung

Das MISP Lab wird in mehrere kleine Übungen unterteilt. Mit Stories soll eine Richtung vorgegeben werden, in welche die Übungen später gehen werden. Dabei wird vor allem auf den Kontext aus Sicht der Studenten eingegangen. Ebenfalls wird ein grobes Lernziel pro Aufgabe, respektive Story definiert.

Story 1

Lernziel:	Der Student sollte in der Lage sein, selbständig eine MISP Instanz zu starten, neue Feeds zu abonnieren und Events anschliessend zu analysieren.
Deployment:	Setup 1 - Lokal als Docker-Compose in der Hacking-Lab Live CD
Gruppengrösse:	1 bis 2 Personen pro Gruppe
Zeit:	30 bis 45 Minuten
Inhalt:	Step 1: Der Student erhält einen Link zu einem Git Repository, unter welchem er ein Docker-Compose File mit den benötigten Ressourcen herunterladen kann. Er klont das Repository direkt auf die Hacking-Lab Live CD. Step 2: Im nächsten Schritt startet er die Ressourcen als Docker Container direkt in der Hacking-Lab Live CD. Mit den zur Verfügung gestellten Zugangsdaten kann er sich auf der Malware Information Sharing Platform über den Webbrowser anmelden. Step 3: Anschliessend kann sich der Student durch die Plattform klicken und sich damit vertraut machen. Step 4: Im vierten Schritt ist es seine Aufgabe, einen neuen vorgegeben Feed zu abonnieren. Step 5: Zum Schluss analysiert der Student ein Event, welches im vorgegeben wird.

Tabelle 3.1: Dokumenation der Übungen - Story 1

Story 2

Lernziel:	Nach erfolgreichem Abschluss der Übungen sollte der Student eigene neue Events erfassen können.
Deployment:	Setup 1 - Lokal als Docker-Compose in der Hacking-Lab Live CD
Gruppengrösse:	1 bis 2 Personen pro Gruppe
Zeit:	30 Minuten
Inhalt:	Step 1: Der Student erhält einen Link zu einem Git Repository, unter welchem er ein Docker-Compose File mit den benötigten Ressourcen herunterladen kann. Er klont das Repository direkt auf die Hacking-Lab Live CD. Step 2: Im nächsten Schritt startet er die Ressourcen als Docker Container direkt in der Hacking-Lab Live CD. Mit den zur Verfügung gestellten Zugangsdaten kann er sich auf der Malware Information Sharing Platform über den Webbrowser anmelden. Step 3: Der Student erhält ein phishing E-Mail, welches ein möglicher Angriff auf das Unternehmen wiederspiegelt. Seine Aufgabe ist es, dieses zu analysieren und mögliche wichtige Hinweise (aus Security Sicht) zu erkennen. Step 4: Die erkannten Hinweise sollen nun in MISP eingetragen werden.

Tabelle 3.2: Dokumenation der Übungen - Story 2

Story 3

Lernziel:	Der Student kann Malware Samples mithilfe einer Online Sandbox auswerten und diese Daten in MISP erfassen.
Deployment:	Setup 1 - Lokal als Docker-Compose in der Hacking-Lab Live CD
Gruppengrösse:	1 bis 2 Personen pro Gruppe
Zeit:	30 Minuten
Inhalt:	Step 1: Der Student lädt im Hacking Lab drei verschiedene Malware Samples herunter. Unbedingt die Live CD verwenden dafür. Step 2: Der Student lädt die bereitgestellte docker-compose Datei aus dem Hacking Lab herunter und startet dieses mit docker-compose up. Eine lokale MISP Instanz steht nun dem Studenten zur Verfügung. Step 3: Die Malware Samples sollen dann in einer Online Sandbox (VirusTotal oder Joesandbox (Registrierung notwendig)) analysiert werden. Step 4: Die Daten der Analyse sollen dann in ein MISP Event übertragen werden. Step 5: Korrelationen zwischen den Events können hergeleitet werden. Step 6: Der Student reicht einen Report ein, welche Korrelationen zwischen den Malware Samples gefunden werden konnte.

Tabelle 3.3: Dokumenation der Übungen - Story 3

Story 4

Lernziel:	Der Student benutzt MISP über die offizielle MISP Schnittstelle und kann darüber Daten beziehen und auslesen.
Deployment:	Setup 1 - Lokal als Docker-Compose in der Hacking-Lab Live CD
Gruppengrösse:	1 bis 2 Personen pro Gruppe
Zeit:	30 Minuten
Inhalt:	Step 1: Der Student erhält eine Einleitung über die API von MISP mit Hinweisen, wo Informationen zur Schnittstelle zu finden sind. Step 2: Der Student lädt das bereitgestellte docker-compose File aus dem Hacking Lab herunter und startet dieses mit docker-compose up. Eine lokale MISP Instanz steht nun dem Studenten zur Verfügung. Step 3: Der Student kann sich nun in seiner lokalen MISP Instanz anmelden und mit dem REST Client im Webinterface erste Queries erstellen. Mit dem Query Builder soll eine Query erstellt werden, in der alle Events in einem gewissen Zeitraum gesucht und angezeigt werden. Step 4: Nun soll auch noch mit einem REST Client auf die API zugegriffen werden. Das Tool "Postman" bietet sich dafür an. Dieses soll vom Student installiert werden. Step 5: Damit auf die API zugegriffen werden kann muss der Student in der MISP Instanz einen API Key generieren. Step 6: Der Student richtet Postman dann nach Vorgaben ein. Dies beinhaltet auch die Eingabe des API Keys. Nun können Queries abgesetzt werden. Mithilfe der Dokumentation soll dann eine Query erarbeitet werden. Diese soll alle Attribute eines gewissen Events anzeigen. Step 6: Als letzten Schritt soll ein Attribut via API Query (mit zum Beispiel einer IP Adresse) zum erwähnten Event hinzugefügt werden. Step 6: Am Ende der Übung gibt der Student ein Report mit dem Gelernten ab. Die geschrieben API Queries sollen auch darin vorkommen.

Tabelle 3.4: Dokumenation der Übungen - Story 4

Story 5

Lernziel:	Der Student versteht wie sich das MITRE ATT&CK in MISP integrieren und verwenden lässt. Er kann dieses Wissen praktisch anwenden.
Deployment:	Setup 1 - Lokal als Docker-Compose in der Hacking-Lab Live CD
Gruppengrösse:	1 bis 2 Personen pro Gruppe
Zeit:	30 Minuten
Inhalt:	Step 1: Der Student erhält eine Einführung in das MITRE ATT&CK Framework. Es sollen grundlegende Fragen beantwortet werden, welche Funktionen das Framework bietet und wieso es in Kombination mit MISP sinnvoll ist. Step 2: Der Student lädt die bereitgestellte docker-compose Datei aus dem Hacking Lab herunter und startet dieses mit docker-compose up. Eine lokale MISP Instanz steht nun dem Studenten zur Verfügung. Step 3: Aus dem Hacking Lab können nun einige Events heruntergeladen werden. Diese sollen nun durch den Studenten importiert werden. Step 4: Nach dem Import der Events sollen zu einem der Events eine MITRE ATT&CK Matrix hinzugefügt werden. Diese soll anhand der Daten, welche bereits im Event erfasst wurden, ergänzt werden. Step 5: Die generierte Matrix soll dann in einem Report dokumentiert und abgegeben werden.

Tabelle 3.5: Dokumenation der Übungen - Story 5

Story 6

Lernziel:	Ziel ist es, dass der Student Events zwischen sich und anderen Organisationen synchronisieren kann.
Deployment:	Setup 2 oder 3 - Im Hacking-Lab als Docker oder in der Azure Cloud
Gruppengrösse:	2 bis 6 Gruppen à 2 Personen
Zeit:	30 Minuten
Inhalt:	Step 1: Der Student startet im Hacking-Lab unter Ressourcen einen Docker Container, welcher ihm anschliessend eine vorbereitete MISP Instanz hochfährt. Step 2: Er sucht sich anschliessend eine Gruppe aus den Übungen, die an einer Synchronisation interessiert ist. Gegenseitig werden die public IP Adressen ausgetauscht. Step 3: In seiner MISP Instanz fügt er die IP Adresse der anderen Gruppe hinzu und verbindet die andere Organisation mit der eigenen.

Tabelle 3.6: Dokumenation der Übungen - Story 6

Story 7

Lernziel:	Der Student sollte in der Lage sein Events anderer Organisationen zu analysieren und anschliessend zu entscheiden, ob sie mit selbst erfassten Events korrelieren.
Deployment:	Setup 2 oder 3 - Im Hacking-Lab als Docker oder in der Azure Cloud
Gruppengrösse:	2 bis 6 Gruppen à 2 Personen
Zeit:	30 bis 45 Minuten
Inhalt:	Step 1: Der Student startet im Hacking-Lab unter Ressourcen einen Docker Container, welcher ihm anschliessend eine vorbereitete MISP Instanz hochfährt. Step 2: Anschliessend soll erneut eine Synchronisation zwischen sich und einer anderen Übungsgruppe erstellt werden. Dies wurde bereits in Story 6 erlernt und kann nochmals repetiert werden. Step 3: Nach einer erfolgreichen Synchronisation wird ein neues Event durch den Student erfasst. Ein Incident wird für das Event vorgegeben. Step 4: Die Aufgabe ist es nun, das Event an weitere Organisationen zu verteilen. Step 5: Andere Gruppen, respektive Organisationen teilen ebenfalls ihre Incidents. So sollten neue Events in der eigenen MISP Instanz erscheinen. Ziel ist es nun, diese mit den eigenen zu vergleichen und zu erkennen, ob diese kongruent sind. Step 6: MISP stellt Tools für das Erkennen von ähnlichen Events zur Verfügung. Der Student soll erneut die Events vergleichen und prüfen, wo sich diese unterscheiden und wieso.

Tabelle 3.7: Dokumenation der Übungen - Story 7

Story 8

Lernziel:	Der Student kann mit dem erlernten Wissen über MISP eine Schwachstelle im Netzwerk finden und diese aufgrund der Daten in MISP mitigiren.
Deployment:	Setup 3 - Azure Cloud
Gruppengrösse:	1 bis 2 Personen
Zeit:	45 Minuten
Inhalt:	Step 1: Der Student erhält eine Einleitung, um was es in dieser Aufgabe geht. Der Fokus liegt hier auf dem Finden und Beheben der Schwachstelle im System. Step 2: Der Student startet die Cloud Umgebung. Darin enthalten sind mehrere Windows Clients welche mit einem AD verbunden sind. Diese können auf einen File Server zugreifen. Ebenfalls ist eine MISP Instanz in der Umgebung vorhanden. Step 3: In der MISP Instanz befinden sich eine Auswahl an erfassten Events. Diese sollen durchsucht werden und dem Studenten Hinweise darauf geben, wo sich eine Schwachstelle im System finden lässt. Step 4: Die erste Schwachstelle die gefunden werden soll, ist ein fehlendes Update eines zum Beispiel Browsers. Um die Schwachstelle zu beheben ist auf den Clients das Update des Browsers einzuspielen. Step 5: Die zweite Schwachstelle die gefunden werden soll, ist eine alte Version des SMB Protokolls. Dieses soll auf dem Fileserver auf eine neuere Version umgestellt werden. Somit ist auch diese Schwachstelle behoben. Step 6: Der Student gibt ein Report ab, in welchem die beiden gefunden Schwachstellen dokumentiert werden. Speziell soll darauf eingegangen werden, wie MISP geholfen hat, die Schwachstellen zu finden.

Tabelle 3.8: Dokumenation der Übungen - Story 8

Story 9

Lernziel:	Nach Abschluss dieser Übung sollte der Student Informationen automatisiert aus MISP exportieren und diese in einem anderen Systemen weiter verarbeiten können.
Deployment:	Setup 3 - Azure Cloud
Gruppengrösse:	1 bis 2 Personen
Zeit:	45 Minuten
Inhalt:	Step 1: Der Student erhält eine Einleitung, um was es in dieser Aufgabe geht. Der Fokus liegt hier auf der Verbindung zwischen MISP und einer Firewall. Step 2: Der Student startet die Cloud Umgebung. Darin enthalten ist ebenfalls eine Firewall. Ebenfalls ist eine MISP Instanz in der Umgebung vorhanden. Step 3: Der Student hat nun die Aufgabe, die beiden Systeme miteinander zu verbinden, sodass IP Adressen automatisch von der Firewall blockiert werden.

Tabelle 3.9: Dokumenation der Übungen - Story 9

Story 10

Lernziel:	Der Student soll ein Tool wie Wazuh oder IDS an MISP anbinden
Deployment:	noch nicht definiert → vermutlich Cloud
Gruppengrösse:	vermutlich 1 bis 2 Personen
Zeit:	45 Minuten
Inhalt:	Step 1: Der Student erhält eine Einleitung in das gewählte System. Step 2: Der Student startet das Deployment der Cloud Umgebung. Step 3: Die MISP Instanz soll im gewählten System angebunden werden. Step 4: Konkrete Aufgabe muss noch definiert werden. Step 5: Konkrete Aufgabe muss noch definiert werden. Step 6: Konkrete Aufgabe muss noch definiert werden. Step 7: Der Student gibt einen Report der Arbeit ab.

Tabelle 3.10: Dokumenation der Übungen - Story 10

3.6.3 Aufgabenstellung

Einleitung

Es folgen die Labs 1 bis 8, wie sie effektiv umgesetzt wurden. Die User Stories dienen als Referenz und Vision. Viele der Stories wurden umgesetzt, jedoch angepasst und weiter verfeinert. Die Anpassungen werden ebenfalls in den folgenden Abschnitten dokumentiert.
Die folgenden Labs sind auch darauf ausgelegt, dass sie alleine lösbar sind. Der Lerneffekt ist beim Lösen in einer Gruppe aber grösser. Es empfiehlt sich eine Gruppengrösse von 1 bis maximal 3 Personen.

MISP Lab 1: Introduction

Übungsinhalt

Der Student startet in dieser Übung die MISP Lab Umgebung zum ersten Mal. Er wird durchs Setup und Installation geführt. Er kann sich dann an der Default Instanz anmelden und soll danach von einem Feed erste Events ziehen.
Nachdem erste Events in der Datenbank erscheinen soll sich der Student ein Event genauer anschauen. Dabei sind die Details nicht wichtig, sondern mehr die generelle Struktur des Events.

Begründung der Themenwahl

Die Lerninhalte dieser Übung gegen dem Studenten eine Grundlage für die weiteren Labs. Ein Event ist ein Grundlegendes Element in MISP und wird darum auch hier schon kurz behandelt. Natürlich wird im weiteren Verlauf der Übungen Events noch mehr im Detail betrachtet.

Veränderungen zur ursprünglichen Story

Dieses Lab wurde fast so umgesetzt wie in der User Story 1 beschrieben. Als einzige Ausnahme soll der Student nicht ein vorgegebenes Event anschauen, sondern ein Event von einem Feed.

Lernziele

- erster Kontakt mit MISP
- laufende Instanz einrichten
- erstes Event kennenlernen

Lernkontrolle

Als Lernkontrolle sollen folgende Fragen durch die Studierende beantwortet werden:

1. Describe in your own words: What is MISP?
2. Note what you think MISP is used for.
3. Explain what a feed is.

Zugang

Das Lab kann auf der Hacking-Lab Plattform oder unter folgendem Link abgerufen werden:
https://assets.vuln.land/challenges/db2c61b7-e28a-47bb-aa96-4aa13b725692.git

Das Hacking-Lab sowie der Link zum Lab ist nicht öffentlich zugänglich. Der Zugriff kann bei Ivan Bütler angefragt werden.

Übersicht auf der Hacking-Lab Plattform

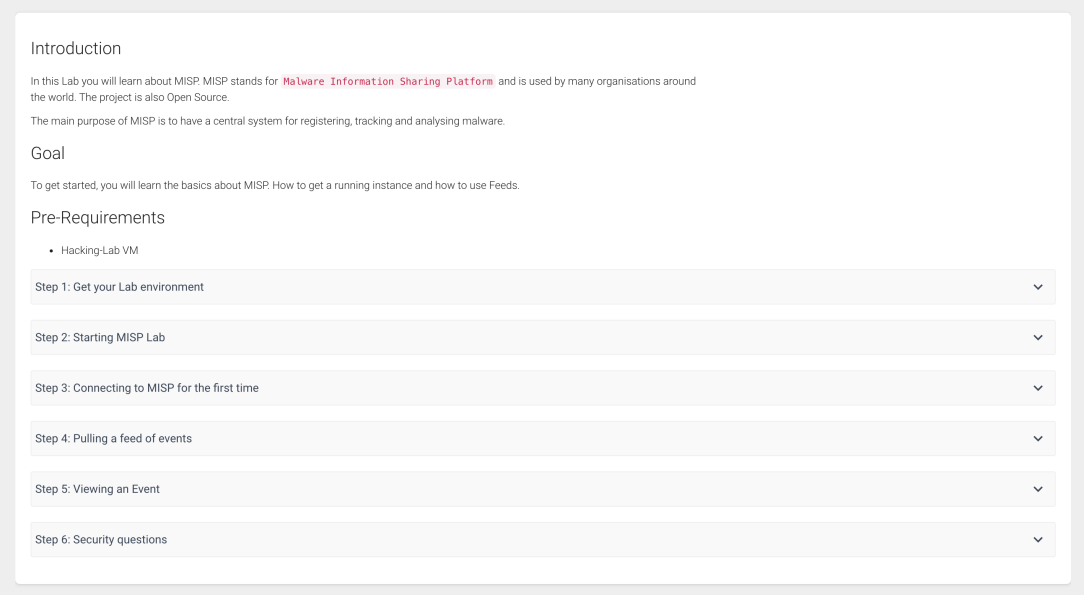

Abbildung 3.16: Übersicht MISP Lab 1

MISP Lab 2: Phishing Email

Übungsinhalt

In diesem Lab erlernt der Student, wie neue Events in MISP erstellt und anschliessend Daten hinzufügt werden. Dabei wird ihm für den Kontext ein Fall vorgegeben, bei welchem sein Unternehmen mit Phishing Emails konfrontiert wird.

Seine Aufgabe ist es, relevante Daten zu erkennen und als Attribute in MISP hinzuzufügen. Ebenfalls wird der Unterschied zwischen Attributen und Objekten in MISP aufgegriffen. Wie Attribute in Objekte zusammengeführt werden können, wird in einem nächsten Schritt erklärt. Weitere Tools wie der Freetext Import oder der Correlation Graph sollen angewendet und verstanden werden.

Begründung der Themenwahl

Attribute und Objekte werden in MISP für jedes Event benötigt und sind daher essenziell. Es ist daher wichtig, dass die Studierenden wissen wie diese eingesetzt werden können. Ebenfalls ist es wichtig, Korrelationen zwischen Events zu erkennen, um auch Nutzen ziehen zu können.

Veränderungen zur ursprünglichen Story

Die ursprüngliche Story 2 konnte übernommen werden, wurde jedoch um diverse Aspekte erweitert. Einerseits wurde das Lab um ein zweites Phishing Email ergänzt, welches mit dem Freetext Import Tool zu MISP hinzugefügt werden soll. Andererseits wird eine erste Einführung zu Event Korrelationen gegeben, da es sich bei diesen vorgegebenen Beispielen sehr einfach veranschaulichen lässt.

Lernziele

- Neue Events erfassen
- Attribute oder Objekte erfassen
- Vorteile von import Tools wiedergeben können
- Einfache Korrelationen zwischen Events erkennen

Lernkontrolle

Als Lernkontrolle sollen folgende Fragen durch die Studierende beantwortet werden:

1. What is the difference between attributes and objects in an event?
2. Why are you just able to merge to person object and not to a employee object?
3. Why does the IP address not show up as a correlation between the two events?

Zugang

Das Lab kann auf der Hacking-Lab Plattform oder unter folgendem Link abgerufen werden:
https://assets.vuln.land/challenges/b9035b4c-57da-4b10-993c-15b2985b37c5.git

Das Hacking-Lab sowie der Link zum Lab ist nicht öffentlich zugänglich. Der Zugriff kann bei Ivan Bütler angefragt werden.

Übersicht auf der Hacking-Lab Plattform

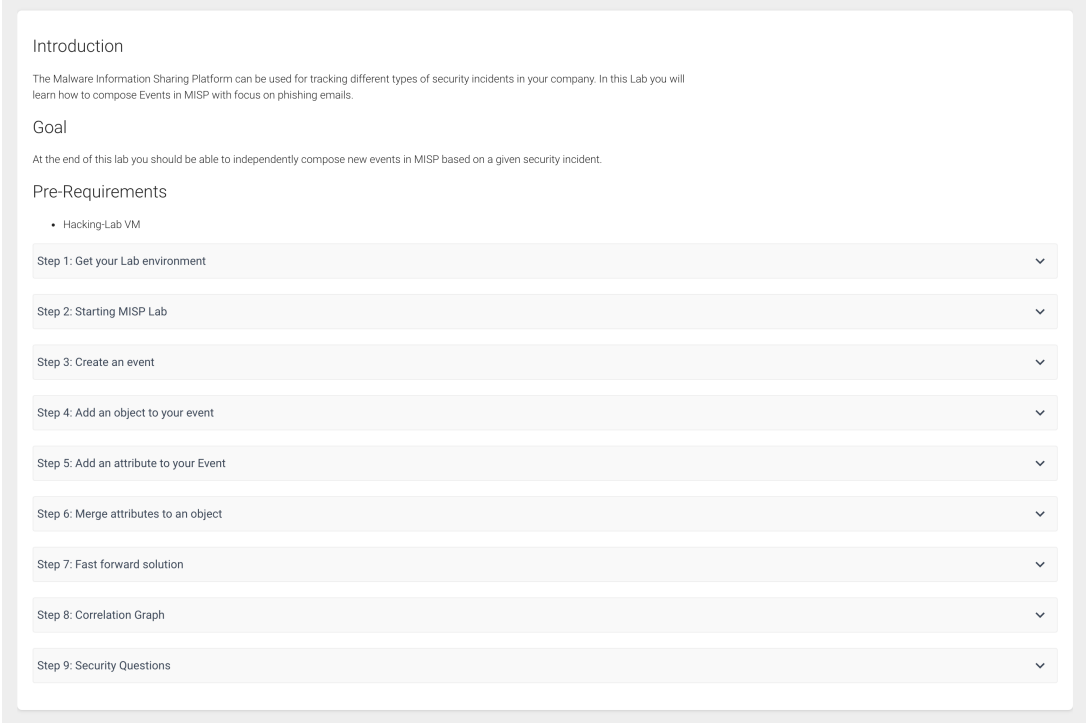

Abbildung 3.17: Übersicht MISP Lab 2

MISP Lab 3: Malware Sandbox

Übungsinhalt

Der Student analysiert in dieser Übung eine echte Malware und erfasst diese Daten in MISP.
Dazu lädt er zuerst vorgegebene Malware Samples herunter. Mithilfe einer Online Malware Sandbox names Hybrid Analysis (www.hybrid-analysis.com) kann die Malware dann analysiert werden. Die Resultate der Analyse sollen dann in MISP als Event übertragen werden. Durch ein zweites Event wird dann ersichtlich, welche Analyse MISP auch selbst machen könnte (hauptsächlich Hashes berechnen).
Durch die beiden Events entsteht nun eine Korrelation, welche durch den Correlation Graph sichtbar wird.

Begründung der Themenwahl

Dieses Lab entspricht einem realistischen Szenario. Eine Online Malware Sandbox zu kennen kann einem helfen, eine schnelle Analyse über Dateien / Links zu machen. Der Correlation Graph in MISP ist eine mächtige Funktion, mit welchem Korrelationen zwischen verschiedenen Event dargestellt werden können.

Veränderungen zur ursprünglichen Story

An diesem Lab wurde grundsätzlich wenig verändert. Als Online Sandbox wurde Hybrid Analysis ausgewählt. Diese Sandbox bietet alle benötigten Funktionen und kann ohne Registrierung benutzt werden.

Lernziele

- Umgang mit Online Sandbox erlernen
- Umgang und Verständnis für Events festigen
- Correlation Graph kennenlernen

Lernkontrolle

Als Lernkontrolle sollen folgende Fragen durch die Studierende beantwortet werden:

1. Explain what you have learned in this exercise.
2. Describe the advantages and disadvantages of uploading the file directly to MISP vs. uploading the file to a sandbox.
3. Please examine the following correlation graph. Describe what you can interpret from the given data. (Ein Correlation Graph wird angezeigt.)

Zugang

Das Lab kann auf der Hacking-Lab Plattform oder unter folgendem Link abgerufen werden:
https://assets.vuln.land/challenges/eba3edd5-e708-4322-8f4d-f637035ce709.git

Das Hacking-Lab sowie der Link zum Lab ist nicht öffentlich zugänglich. Der Zugriff kann bei Ivan Bütler angefragt werden.

Übersicht auf der Hacking-Lab Plattform

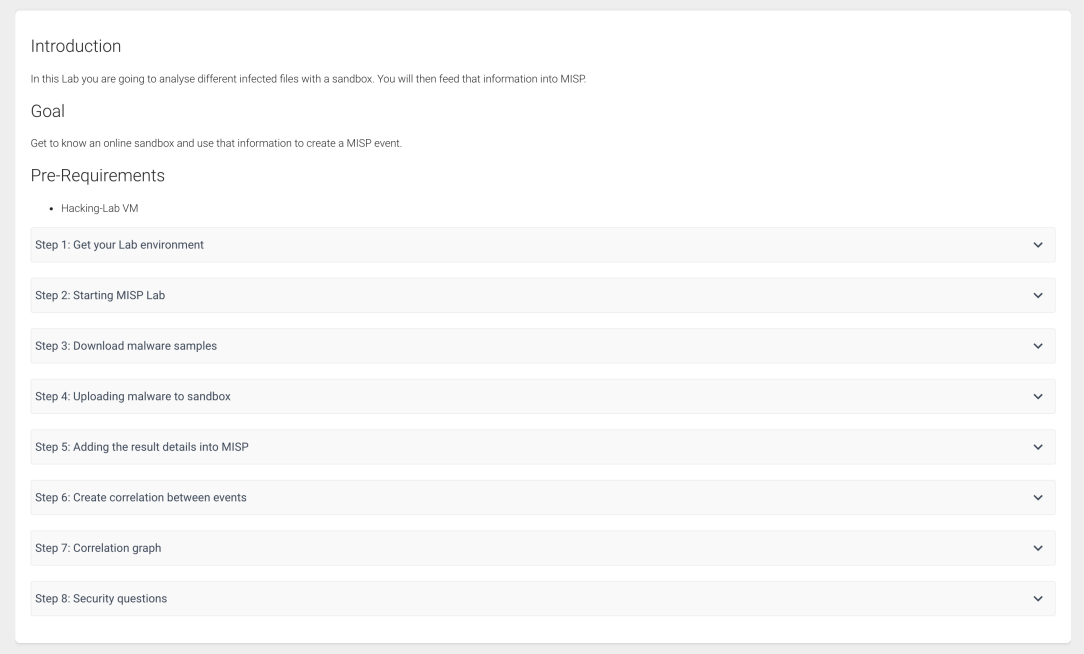

Abbildung 3.18: Übersicht MISP Lab 3

MISP Lab 4: API

Übungsinhalt

Als Einführung in die Übung wird zuerst erklärt wieso MISP eine API besitzt und wo die Dokumentation zu finden ist. Die eigentliche Übung beginnt dann mit dem Erstellen des API Keys. Dieser muss zwingend erstellt werden, da dieser für die Authentisierung der folgenden API Calls verwendet wird.
Danach kann mit einem Curl Befehl ein erster API Call gemacht werden. Dieser erstellt auf der Instanz ein neues Event. Dieses ist auch über das GUI sichtbar.
Im letzten Schritt wird dann PyMISP vorgestellt. Diese Python Library ermöglicht die einfache Kommunikation zu einer MISP Instanz über Python. Das Skript fügt ebenfalls ein Event hinzu.

Begründung der Themenwahl

Das Lab bietet einen Einblick "hinter die Kulussen" und zeigt, was für Möglichkeiten existieren, Vorgänge mit Skripts via API zu automatisieren. Auch wenn noch nie mit einer API gearbeitet wurde sollte dieses Lab möglich sein.

Veränderungen zur ursprünglichen Story

Kleine Änderungen wurden vorgenommen. Es wird nicht Postman sondern Python für die API Abfragen verwendet. Am Ende der Übung wird auch ein kurzes Python Skript in der Lernkontrolle gefordert.

Lernziele

- MISP API kennenlernen
- PyMISP Python Library kennenlernen

Lernkontrolle

Als Lernkontrolle sollen folgende Fragen durch die Studierende beantwortet werden:

1. Describe what you can do with the MISP API and who you think might use this API.
2. Please write a short script that will return some information about the event you just created. Please print if the event is published, how many attributes the event has and what distribution level it has. Use the API documentation and your knowledge from the steps in this lab to complete this challenge.

Zugang

Das Lab kann auf der Hacking-Lab Plattform oder unter folgendem Link abgerufen werden:
https://assets.vuln.land/challenges/82ad7d36-c538-4d62-b265-a92b58b546f9.git

Das Hacking-Lab sowie der Link zum Lab ist nicht öffentlich zugänglich. Der Zugriff kann bei Ivan Bütler angefragt werden.

Übersicht auf der Hacking-Lab Plattform

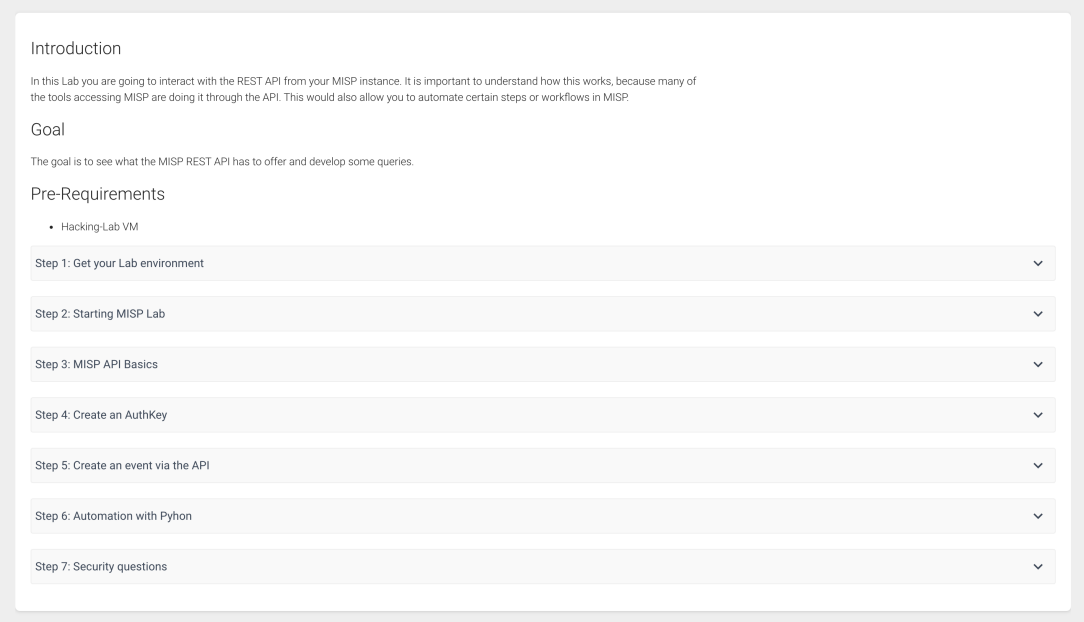

Abbildung 3.19: Übersicht MISP Lab 4

MISP Lab 5: Event Graph | MITRE ATT&CK

Übungsinhalt

Das Lab 5 umfasst einerseits den Event Graph, andererseits Galaxies und Taxonomies in MISP. In einem ersten Schritt wird ein Event und eine kurze Beschreibung dazu vorgegeben. Der Student soll beim betrachten des Events erkennen, dass es schwierig sein kann zu erkennen, welche Attribute und Objekte wie zusammenhängen. In einem nächsten Schritt soll aufgrund der vorgegebenen Beschreibung ein Event Graph erstellt werden, welcher eine grafische Übersicht über den Vorfall gibt. Ziel ist es, dass der Event Graph genügend Informationen zu Beziehungen darstellt, sodass weitere Researcher sich schnell zurechtfinden.

Zuletzt sollen Galaxies in MISP angewendet werden. Dafür wird als Beispiel das MITRE ATT&CK Framework verwendet. Es wird versucht, eine möglichst einfache Einführung in Galaxies zu geben, ohne alle Funktionen abzudecken. In der Lernkontrolle wird nochmals der Unterschied zwischen Galaxies und Taxonomien aufgegriffen, was bereits in der Vorlesung behandelt wurde. Ziel ist es, dass die Studierenden selbständig versuchen Tags hinzuzufügen und so die Möglichkeiten, Einschränkungen und Differenzen erkennen.

Begründung der Themenwahl

Galaxies und Taxonomies werden bei fast jedem Event eingesetzt. Sie sind essenzielle Features in MISP und werden generell verwendet. Um den Studierenden ein Beispiel zu liefern, welches sie bereits aus der Cyber Defense Vorlesung kennen wurde das MITRE ATT&CK Framework gewählt. Taxonomies wurden in der Vorlesung bereits ebenfalls aufgegriffen. Die Studierenden sollten jedoch selbständig versuchen diese hinzuzufügen.

Der Event Graph wird zwar nicht so oft eingesetzt wie Galaxies und Taxonomies, ist jedoch ein sehr nützliches Tool das von MISP bereitgestellt wird.

Veränderungen zur ursprünglichen Story

Die ursprüngliche Story 5 wurde teilweise übernommen werden. Dank des von uns bereitgestellten Konfigurationsskript werden Events beim ersten Start direkt zu MISP hinzugefügt und müssen nicht vom Studenten im Hacking-Lab manuell heruntergeladen und in MISP wieder importiert werden. Da das Lab vom Grössenumfang eher zu klein geworden wäre, musste darauf reagiert werden. Das Thema Event Graph wurde so zusätzlich hinzugefügt.

Lernziele

- Visualisierungen mit dem Event Graph erstellen
- Galaxies kennenlernen und anhand des MITRE ATT&CK Framework in MISP einsetzen
- Unterschiede zwischen Galaxies und Taxonomies rekapitulieren

Lernkontrolle

Als Lernkontrolle sollen folgende Fragen durch die Studierende beantwortet werden:

1. What do you have learned in this lab?
2. What are advantages of using the MISP event graph?
3. You didn't used the left world icon in step 6. This would add tags to your attribute. What is the difference between tags and galaxies in MISP?

Zugang

Das Lab kann auf der Hacking-Lab Plattform oder unter folgendem Link abgerufen werden:
https://assets.vuln.land/challenges/140e25bf-5172-4142-bcb0-3788532d74dd.git

Das Hacking-Lab sowie der Link zum Lab ist nicht öffentlich zugänglich. Der Zugriff kann bei Ivan Bütler angefragt werden.

Übersicht auf der Hacking-Lab Plattform

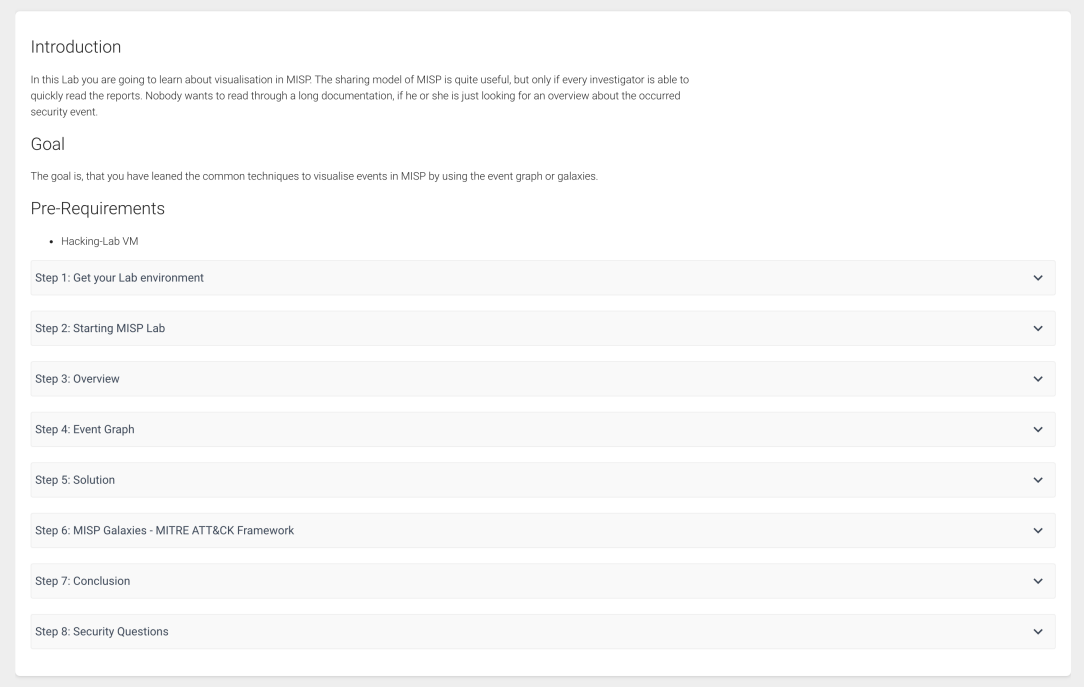

Abbildung 3.20: Übersicht MISP Lab 5

MISP Lab 6: Sharing

Übungsinhalt

Im Lab 6 wird auf die Synchronisation zwischen mehreren MISP Server eingegangen. Ziel ist es, dass Studierende selbstständig nach dem Erfassen von Events ein Distribution Level setzten können und verstehen, in welcher Community die Informationen verbreitet werden. Zu Beginn der Übung wird eine Einführung in das Sharing Model gegeben, welche die Studierenden erneut mit einem kurzen Theorieblock ins Thema einführt. Darauf werden Events mit verschiedenen Distribution Levels geteilt. Beginnend mit der Verteilung auf Organisationsebene bis zum Verbreiten von Informationen auf alle verbundenen Instanzen. Die Studierende sollen dabei auch neue Benutzer, Organisationen und MISP Server anlegen, respektive zum eigenen System hinzufügen. Bei erfolgreichem Abschluss sollten drei verschiede MISP Server miteinander verbunden sein und das Teilen von Information einwandfrei funktionieren.

Um die Grösse dieses Labs zu begrenzen werden die Server nur über das Pull-Verfahren miteinander verbunden. Eine mögliche Erweiterung dieses Labs wäre es, weitere Server mit dem Push-Verfahren anzubinden. Zusätzliche wäre es in nachfolgenden Labs möglich, Synchronisationsregeln mittels Taxonomies umzusetzen. Um noch weitere Themen den Studenten näher zu bringen, entschieden wir uns in den nachfolgenden Labs das MISP Sharing Modell nicht weiter zu vertiefen. Die Infrastruktur ist jedoch für Erweiterungen vorbereitet.

Begründung der Themenwahl

Teilen von Informationen ist einer der Hauptfunktionen in MISP. Durch das Verbreiten von Informationen über erkannte Cyberangriffe können Unternehmen sowie auch Researcher sich auf zukünftige Angriffe besser vorbereiten. Durch die Sharing-Funktion wird dies ermöglicht und gehört daher auf jeden Fall zu den wichtigsten Themen. Ebenfalls wird das Rollen- sowie Organisations- Modell in diesem Lab aufgegriffen, da dieses ebenfalls für die Synchronisierung durch die Studierenden verstanden werden muss.

Veränderungen zur ursprünglichen Story

Beim Entwerfen der Story 6 gab es einige Unklarheiten, wie das Teilen von Informationen in MISP genau funktioniert und in Form eines Labs umgesetzt werden soll. Ursprüngliche Idee war es, das Lab in mehreren Gruppen zu lösen und gegenseitig sich Events zuzusenden. Leider war die Teilnehmerzahl in den Übungen eher niedrig während dem Semester. Ein möglicher Grund dafür könnte die aktuelle Covid-19-Pandemie sein. Der Präsenzunterricht an der OST wurde zusätzlich ende Semester komplett eingestellt. Es wurde daher frühzeitig reagiert und die Übungen so entwickelt, dass die Studierenden das Lab komplett alleine lösen können, ohne auf andere Teilnehmer angewiesen zu sein.

Lernziele

- Synchronisation zwischen mehreren MISP Server praktisch anwenden
- Distribution Levels auf Event und Attribut Level verstehen
- Organisations und Rollen Modell wiedergeben können
- Unterschiede zwischen Synchronisation und Feeds erkennen und beschreiben können

Lernkontrolle

Als Lernkontrolle sollen folgende Fragen durch die Studierende beantwortet werden:

1. Step 8: On your instance A, you defined the distribution level of this event as Community only, but on instance B the distribution level is set to Organisation - Why?
2. Step 10: Why is it NOT necessary to update the distribution level of this event?
3. Step 11: To which other MISP instances will the MISP instance B forward the Hello World from Org A event? Justify your answer thoughtfully.
4. What do you think is the key difference between the learned synchronisation philosophy (learned in this lab) and feeds in MISP?

Zugang

Das Lab kann auf der Hacking-Lab Plattform oder unter folgendem Link abgerufen werden:
https://assets.vuln.land/challenges/38b35bea-1a63-4885-95d4-222fd35cfe8e.git

Das Hacking-Lab sowie der Link zum Lab ist nicht öffentlich zugänglich. Der Zugriff kann bei Ivan Bütler angefragt werden.

Übersicht auf der Hacking-Lab Plattform

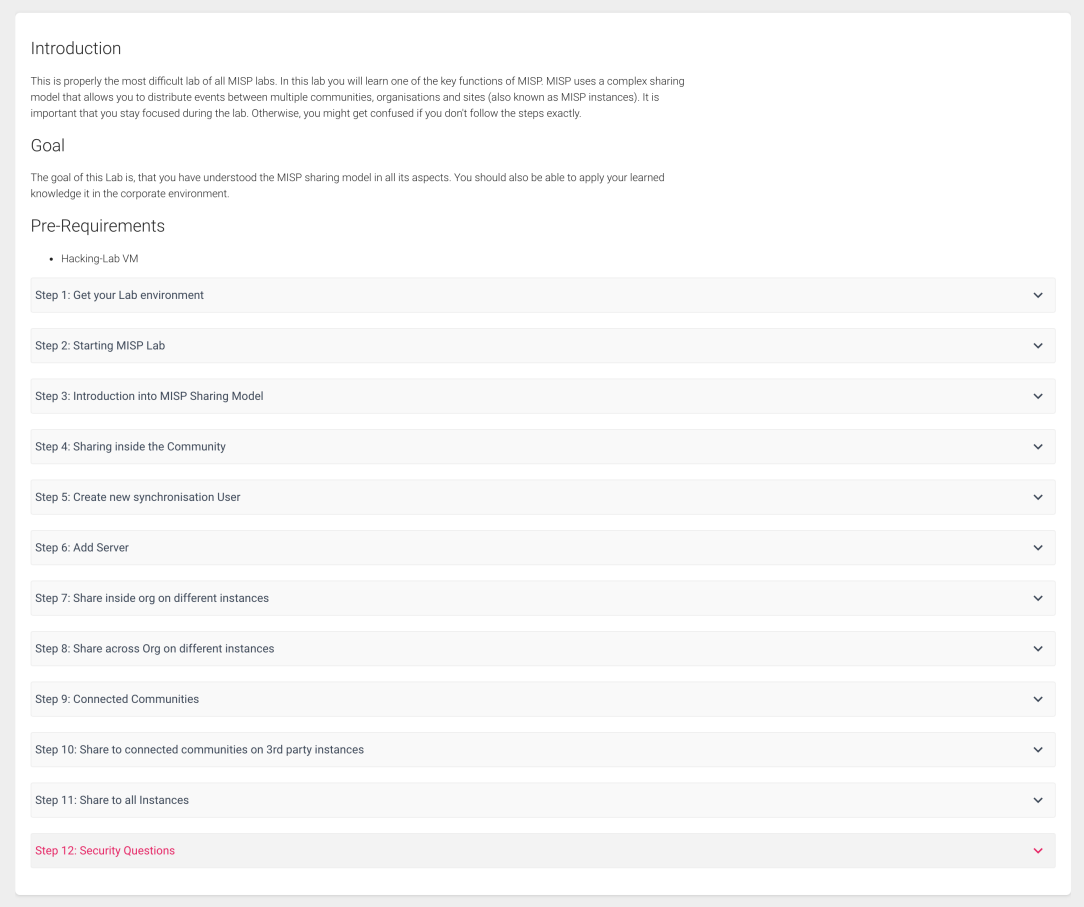

Abbildung 3.21: Übersicht MISP Lab 6

MISP Lab 7: Expansion Modules

Übungsinhalt

Im Lab 7 werden die Erweiterungen für MISP abgedeckt. Die Studierenden lernten bereits, dass MISP ein Open Source Projekt ist, welches dementsprechend mit eigenen Bedürfnissen an Funktionen erweitert werden kann. Dies ist jedoch eher aufwendig, da die Funktionsweise des Kerns der Applikation zuerst verstanden werden muss und anschliessend umgebaut werden muss. Ein anderer Ansatz wäre es die API zu verwenden und die Logik in eine externe Softwarekomponente auszulagern. Nachteil ist dabei die fehlende Integration in das User Interface von MISP. Die Lösung für dieses Problem sind daher Expansion Modules.

Die Studierende sollen sich mit bereits existierenden Expansion Modules auseinandersetzten und erkennen wie sie eingesetzt werden können. In einem ersten Schritt geht es hauptsächlich um die direkte Anwendung auf Event Ebene. Später werden Events mit Expansion Modules Enriched und dabei Informationen in Form von Attributen oder Objekten hinzugefügt. Als Abschluss dieses Labs wird ein weiteres Export Module hinzugefügt, damit die Studierende erkennen, wie Expansion Modules verwaltet werden können.

Wir überlegten uns zusätzlich, die Übungsteilnehmer ein eigenes Expansion Module erstellen zu lassen. Dies wäre technisch auch umsetzbar, jedoch wollten wir das zeitintensive Lab 6 (Sharing Model) kompensieren. Das Lab könnte jedoch um diesen Aspekt weiterentwickelt oder erweitert werden.

Begründung der Themenwahl

Expansion Modules sind sehr nützlich für die Erweiterung der MISP Funktionalität. Wir entschieden uns deshalb, den Studierenden eine kleine Einführung zu geben.

Veränderungen zur ursprünglichen Story

Die Story 7 wurde nicht in diesem Lab übernommen. Grund dafür war es, dass die Studierende bereits genug oft in anderen Labs neue Events erstellten und wir so einen neuen Aspekt schaffen wollten. Ein weiterer Grund war es, dass die Labs auch ohne andere Teilnehmer gelösst werden können. Wir entschieden uns daher, das Lab neu zu orientieren.

Lernziele

- Erkennen des Nutzens von Expansion Modules
- Anwenden von Expansion Modules
- Enrichment von Events

Lernkontrolle

Als Lernkontrolle sollen folgende Fragen durch die Studierende beantwortet werden:

1. Why should even foreign (trustworthy) Modules be used?
2. What do you think, why does MISP introduced a feature like Modules, when it's already open source?
3. What would be another useful feature that is not yet implemented in MISP and could be added by yourself?

Zugang

Das Lab kann auf der Hacking-Lab Plattform oder unter folgendem Link abgerufen werden:
https://assets.vuln.land/challenges/38b35bea-1a63-4885-95d4-222fd35cfe8e.git

Das Hacking-Lab sowie der Link zum Lab ist nicht öffentlich zugänglich. Der Zugriff kann bei Ivan Bütler angefragt werden.

Übersicht auf der Hacking-Lab Plattform

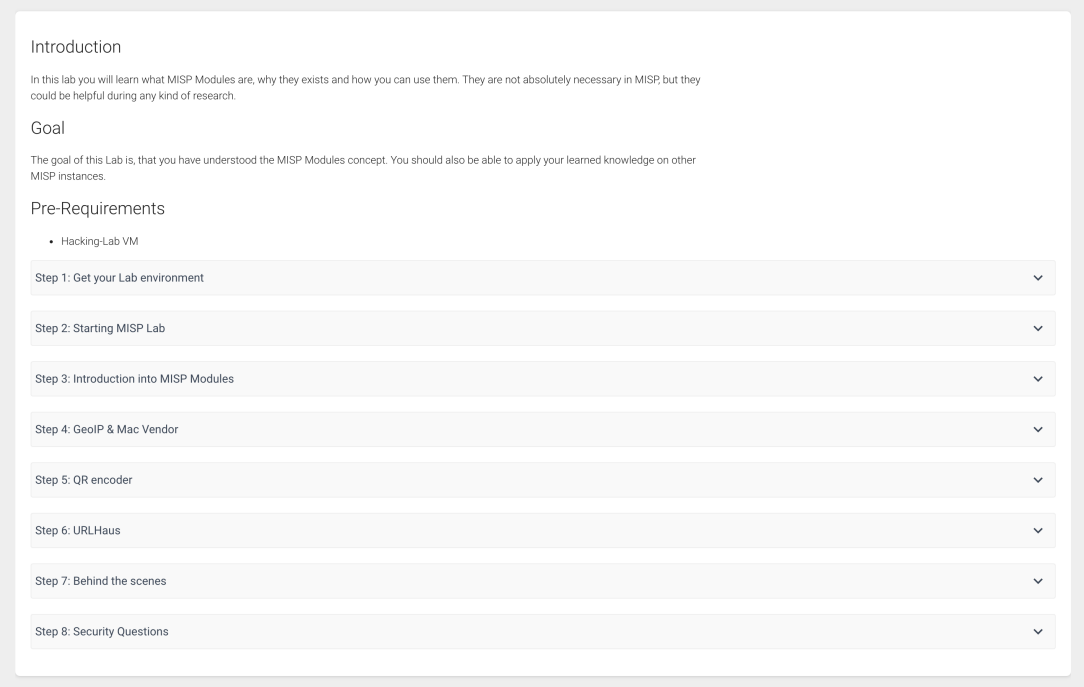

Abbildung 3.22: Übersicht MISP Lab 7

MISP Lab 8: Warning Lists

Übungsinhalt

Der Student lernt Warninglists in MISP kennen. Er navigiert dazu zur Auswahl von vielen verschiedenen Warninglists. In der Übung wird dann vorgegeben welche zu aktivieren sind. Damit die eingeschalteten Listen nun einen Effekt zeigen, muss der Student noch ein Event erstellen. Mit präparierten Werten erhält man dann eine Warnung. Das hinzugefügte Attribute taucht in mehreren Warninglists auf.

Begründung der Themenwahl

Warninglists können einfache Indizien sein, welche in MISP auf Attributlevel angezeigt werden. Diese Funktion erlaubt es in MISP Daten effizienter analysieren zu können, da mehr Daten zur Verfügung und Korrelationen stehen.

Veränderungen zur ursprünglichen Story

Dieses Lab war nicht so als User Story geplant. Die Funktion kam erst später in einer Research Phase zur Geltung und wurde daher auch erst am Schluss der Arbeit implementiert.

Lernziele

- Warninglists kennenlernen
- Warninglists benutzen / interagieren

Lernkontrolle

Als Lernkontrolle sollen folgende Fragen durch die Studierende beantwortet werden:

1. Explain why this information is useful and how investigators might use this feature.
2. Why does the value bit.ly set off two warning? Explain.
3. Try to add a second attribute to the event and add a value also in one of the enabled warninglists. Does it also show up as a warning? Please add a screenshot.

Zugang

Das Lab kann auf der Hacking-Lab Plattform oder unter folgendem Link abgerufen werden:
https://assets.vuln.land/challenges/e0e73d95-d320-4189-8fa8-875aef568a9b.git

Das Hacking-Lab sowie der Link zum Lab ist nicht öffentlich zugänglich. Der Zugriff kann bei Ivan Bütler angefragt werden.

Übersicht auf der Hacking-Lab Plattform

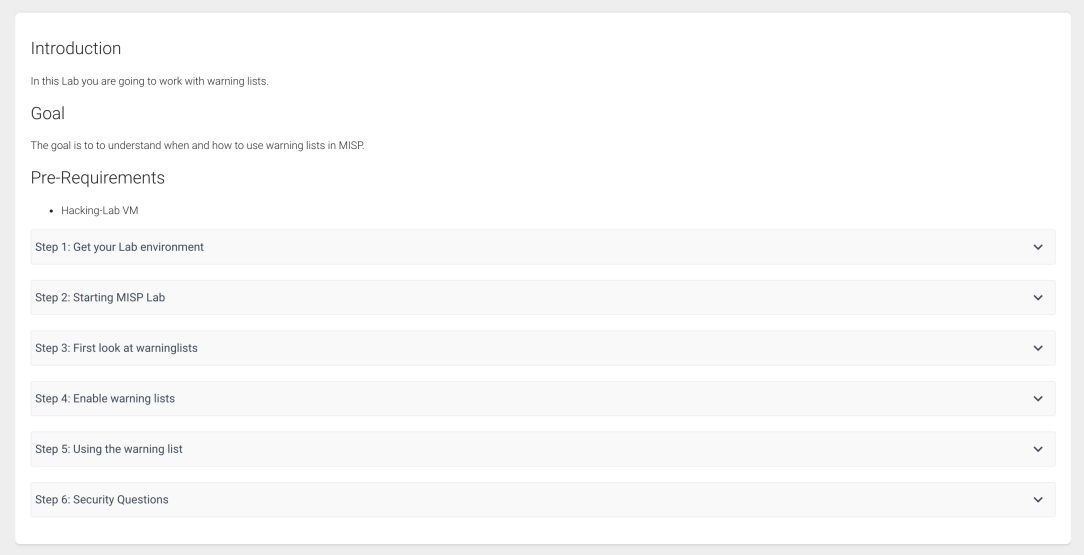

Abbildung 3.23: Übersicht MISP Lab 8

3.7 Testing

In diesem Kapitel wird beschrieben, wie die Funktionalität und Qualität der Labs getestet wird und welche Rückmeldungen wir erhalten haben.

3.7.1 Einleitung

Damit die Labs unseren Anforderungen entsprechen, wurden die Labs ausführlich getestet. Das Testing kann in drei Teile unterteilt werden.

Lab-Tests Intern

Nach dem Entwickeln eines Labs wurde das Lab zuerst nochmal vollständig vom Autor geprüft. Sobald dieser Test abgeschlossen wurde, hat der SA Partner das Lab ebenfalls durchgearbeitet und allfällige Fehler, Unklarheiten oder Feedback direkt behoben oder an den Autor gemeldet. Während der nächsten Sitzung wurde noch Feedback vom Betreuer eingeholt. Allfällige Anpassungen wurden jeweils bis zur nächsten Sitzung umgesetzt.

Lab-Tests mit kleiner Testgruppe

Um von Studenten so realistisch wie möglich Feedback abzuholen haben wir in unserem Umkreis Studenten gebeten, die Labs zu lösen und Feedback zu geben. Diese Feedbacks wurden mit Survey Monkey [19] gesammelt. Das Tool erlaubt es, einfache Umfragen zu erstellen, welche vom Team dann übersichtlich ausgewertet werden können.
Einige der Labs wurden von dieser Testgruppe gelöst. Das erhaltene Feedback war hilfreich. Da jedoch viele dieser Studenten selbst auch mit einer Semesterarbeit beschäftigt waren, fiel es uns etwas schwer die Motivation aufzubringen. Das Testen der Labs bedeutet einen deutlichen Mehraufwand. Dadurch wurden diese Tests leider mit einer tiefen Quote erfüllt.

Lab-Tests mit Cyber Defense Studenten

Die MISP Labs im Cyber Defense Modul durchzuführen war der dritte Testdurchlauf. Durch die grosse Menge an Testpersonen fiel das Feedback deutlich Informatiker aus.

3.7.2 Feedbacks

Die Feedback werden folgend zusammengefasst.

Feedback zur Vorlesung

Nach der Durchführung der Vorlesungs im Cyber Defense Modul konnten die Studenten Feedback via ein Online Formular hinterlassen. Die Feedbacks waren mehrheitlich sehr positiv, jedoch auch zum Teil widersprüchlich.

- *Die Grafiken waren sehr hilfreich und verständlich.*
- *Die Inhalte der Vorlesung waren sinnvoll und nachvollziehbar organisiert.*
- *Der Inhalte wurden verständlich vermittelt.*
- *Roter Faden hat in der Vorlesung gefehlt.* → ev. ein Beispiel einbauen, welches sich über die gesamte Vorlesung erstreckt.
- *Teilweise offene Fragen, welche am Schluss der Vorlesung in der Fragerunde geklärt wurden.*

Feedback zur Lab Übungen

Da nach der Vorlesung die Übungen von den Studenten im Cyber Defense Modul gelöst werden soll, wurden sie darauf hingewiesen, dass Feedback auch auszufüllen ist.
Auch hier war das Feedback mehrheitlich gut, allerdings tauchen ein paar Verbesserungspunkte auf.

- *Aufgabenstellungen sind klar.*
- *Funktional hat bis jetzt alles geklappt.*
- *Screenshots sind teilweise etwas klein.* → ev. Screenshots mit Vergrösserung anbringen oder auf Zoomfunktion in Browser hinweisen
- *Das docker-compose dauert zwar am Anfang beim Starten etwas lang und die Dateigrösse sei etwas gross.* → Das stimmt, ist aber sehr schwierig zu vermeiden, da das MISP Image von MISP selbst schon sehr gross ist.
- *Die Übungen sind sehr gut lösbar, allerdings ist der Umfang zu gross für 2 Lektionen (2x 45 Minuten).* → Das ist leider so. Die Übungen waren für mindestens 4x 45 Minuten, also 4 Lektionen ausgelegt.

3.7.3 Fazit

Allem in allem sind wir mit dem Feedback sehr zufrieden. Die Vorlesung hat den meisten Studenten gefallen und wir haben wenige Verbesserungsvorschläge erhalten.
Soweit wir mitbekommen haben, sind die Übungen lösbar und meistens gut beschrieben. Der eingeplante Slot von 2 Lektionen reicht allerdings nicht aus. Das war uns bewusst, und bestätigt unsere Annahme, dass eher 4 Lektionen dafür aufgewendet werden müssten.

3.8 Ergebnisse

Die definierten Ziele der Arbeit wurde erreicht.

Während dem Research Prozess wurden acht Labor Umgebungen für Informatikstudenten erstellt. Die dazugehörige Infrastruktur mit Docker Images, welche benötigt wird für die Labs wurde automatisiert und erweiterbar eingerichtet. Die 8 Labs wurden in der Hacking Lab Platform deployed und mit einer Gruppe von Studenten im Modul Cyber Defense getestet.
Zusätzlich wurde die Vorlesung im Cyber Defense Modul zum Thema MISP gehalten (2x 45 min). Die Übungen zur Vorlesung wurden auch betreut.
Wie vorgegeben wurde der Entwicklungsprozess von Start bis Schluss dokumentiert.

3.9 Schlussfolgerungen

Die definierten Ziele der Aufgabestellung wurden alle erreicht.
Die optionale Vorlesung im Cyber Defense Modul wurde durchgeführt. Zusätzlich wurde die Übungsbetreuung ebenfalls übernommen.

Eventuelles Feedback zu den Modulen kann, dank der modularen und dokumentierten Infrastruktur, auch nach der Abgabe der Arbeit noch vom Betreuer umgesetzt werden. Die acht erstellten Labs können somit für ein nächstes Modul verwendet werden.

KAPITEL 4

Projektdokumentation

4.1 Projektplan

4.1.1 Projekt Übersicht

Ziel dieser Arbeit ist es, ein Produkt zu erarbeiten, welches Studierende an der Ostschweizer Fachhochschule (OST) beim kennenlernen der Malware Information Sharing Platform (MISP) unterstützt. Dazu sollen diverse Übungen für die Teilnehmer des Cyber Defense Modul erstellt werden. Die Übungen werden anschliessend an eine Vorlesung im Security Lab an der OST absolviert.

Ziel und Zweck

MISP ist eine Sharing Platform für Cyberangriffe jeglicher Art. Diese wird weltweit eingesetzt, um Informationen über Sicherheitsrisiken mit anderen Unternehmen oder Staatlichen Institutionen zu teilen. Ziel und Zweck unserer Arbeit ist es, Studierenden an der OST MISP näher zu bringen sowie praktische Erfahrungen mit der Platform zu erlernen. Unsere Anwender sollten dabei die Möglichkeit haben, MISP selbst in den Labs einzusetzen, ohne vorhergehendes Wissen über deployment in einer Serverumgebung oder in der Cloud zu besitzen.

Abgabe

Die Abgabe unserer Studienarbeit erfolgt gemäss Studiengangleitung Informatik unter den folgenden Seiten:

- http://abstract.rj.ost.ch/
  Erfassung des Abstract im Online Tool sowie Abgabe an Betreuer zur Kontrolle.
- https://archiv-i.ost.ch/Overview/All
  Abgabe des Berichts an den Betreuer und hochladen aller Dokumente.

Abgaben zu Meilensteinen

Nach dem Erreichen eines Meilensteins werden die geleisteten Arbeiten dem Betreuer an der nächsten Sitzung präsentiert sowie gemeinsam diskutiert. Allfällige Änderungen werden priorisiert und möglichst zeitnah ergänzt.

4.1.2 Management Abläufe

Kostenvoranschlag

Das Projekt startet in der 1. Semesterwoche am 20. September 2021 und endet in der 14. und letzten Semesterwoche am 24. Dezember 2021 mit der Abgabe aller Dokumente.
Für eine Studienarbeit wird ein Zeitaufwand von 240 Stunden für jeden Studierenden erwartet. Das Autorenteam besteht aus zwei Studenten, welches einen Totalen Zeitaufwand von 480 Stunden für das ganze Projekt ergibt.

	ECTS	Stundenaufwand pro Woche	Stundenaufwand Gesamt
Janis Wolf	8	17.1	240
Marius Zindel	8	17.1	240
Total	16	34.2	480

Tabelle 4.1: Stundenaufwand

Pro Woche und pro Person müssen durchschnittlich etwas über 17 Arbeitsstunden geleistet werden.

Zeitliche Planung

An der OST werden in den Modulen Software Engineering 1 sowie Software Engineering 2 unterschiedliche Methoden kennengelernt, wie Projekte in der Informatik organisiert werden können. Zwei Beispiele dafür sind der Rational Unified Process oder Scrum, welche beide ihre Vor- sowie Nachteile in gewissen Aspekten haben.

Rational Unified Process (RUP)

RUP ist ein sehr umfassender Prozess und eignet sich sehr gut für Langzeitplanung. Dabei werden vier verschiedene Phasen unterschieden:

- Inception
  - Ungefähre Vision
  - Festlegen des Umfangs
  - Grobe Schätzungen für den Aufwand
- Elaboration
  - Identifizierung der meisten Anforderungen
  - Iterative Umsetzung der Kernarchitektur
  - Beseitigung von hohen Risiken
  - Realistischere Schätzungen für den Aufwand
- Construction
  - Iterative Umsetzung der Funktionalität
  - Behebung geringerer Risiken
  - Vorbereitung auf das Deployment
- Transition
  - Tests
  - Finales Deployment

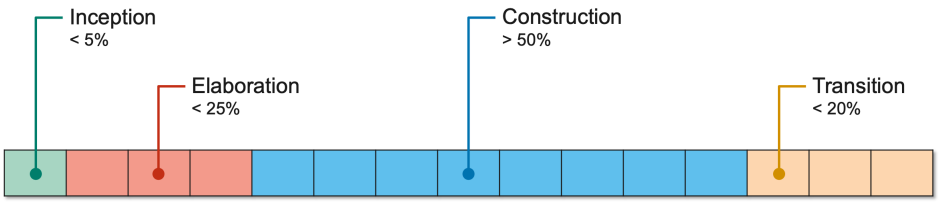

Abbildung 4.1: Rational Unified Process [20]

Scrum

Scrum hingegen ist ein leichtgewichtiges Framework, das Teams hilft, Lösungen für komplexe Probleme zu erreichen. Dies ermöglicht eine genauere Planung auf kurze Zeit. Scrum besteht hauptsächlich aus 3 Rollen, 5 Events, 3 Artefakten sowie 5 Typen von Werten:

- Scrum Team
  - Developers
  - Product Owner
  - Scrum Master
- Scrum Events
  - The Sprint
  - Sprint Planning
  - Daily Scrum
  - Sprint Review
  - Sprint Retrospective
- Scrum Artifacts
  - Product Backlog
  - Sprint Backlog
  - Increment
- Scrum Values
  - Commitment
  - Focus
  - Openness
  - Respect
  - Courage

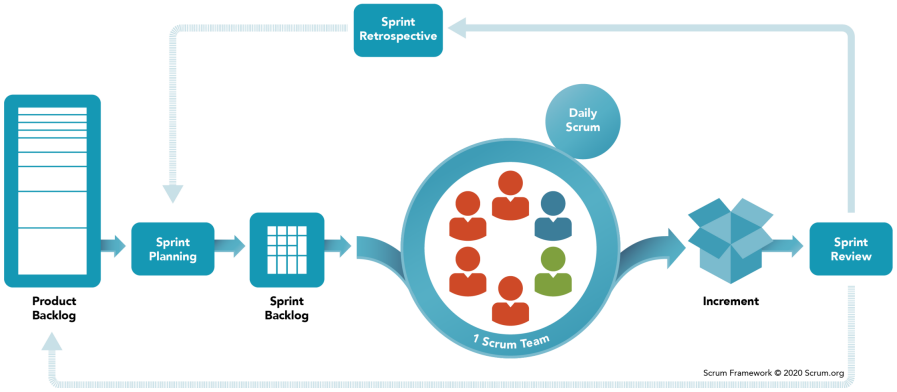

Abbildung 4.2: Scrum [20]

Scrum+

Eine weitere Methode, wie Projekte organisiert werden können ist Scrum+. Dies ist eine Kombination aus den besten Teilen von RUP und Scrum. Einsatzort der beiden Modelle ist hauptsächlich die Softwareentwicklung, jedoch können beide (wie auch die Kombination Scrum+) auf fast jedes andere Projekt adaptiert werden. Ziel ist es so, das Projekt in vier Phasen aufzuteilen, welche dann mit mehreren agilen Sprints durchgeführt werden.Dieses Projekt wird daher nach Scrum+ organisiert.

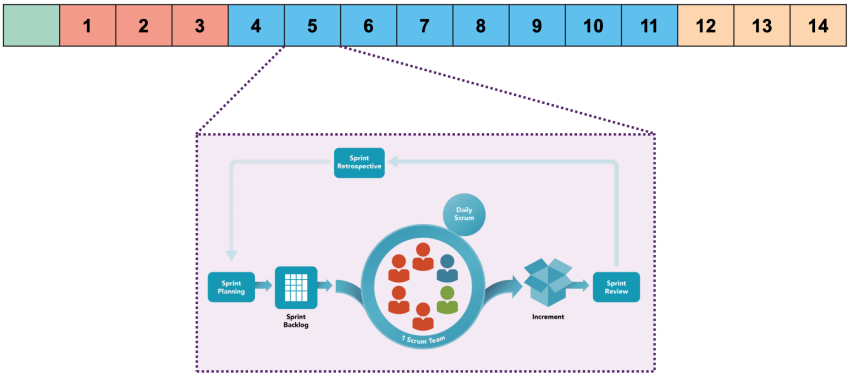

Abbildung 4.3: Scrum+ [20]

Projektorganisation

Autoren

Das Autorenteam besteht aus zwei Personen: Janis Wolf und Marius Zindel. Die Arbeit wird so als Team geschrieben. Dadurch entsteht eine flache Struktur und alle beteiligten sind gleichermassen am Projekt beteiligt. Wesentliche Entscheidungen können im Projektteam gemeinsam getroffen werden. Für beide gilt diese Arbeit als Studienarbeit an der Ostschweizer Fachhochschule.

Betreuung

Das Team wird zusätzlich durch einen Betreuer der Hochschule durch die Studienarbeit geführt. Für die Betreuung der Studienarbeit ist Ivan Bütler zuständig, welcher als Lehrbeauftragter an der OST im Bereich Security tätig ist.

Organisationsstruktur

Zuständigkeiten

Das Autorenteam besteht aus zwei Personen. Dies führt dazu, dass die beiden Autoren für mehrere Aufgaben zuständig sind. Um bei Unstimmigkeiten die Meinungsunterschiede aufzulösen wurde eine verantwortliche Person für die Dokumentation und das Produkt definiert.

- Janis Wolf
  - Scrum Product Owner
  - Verantwortlich für die Qualität des Produkts
  - Scrum Developer
- Marius Zindel
  - Scrum Master
  - Verantwortlich für die Qualität der Dokumentation
  - Scrum Developer

Externe Schnittstellen

Über das Projekt gibt es keine Externen Schnittstellen. Auftraggeber für die Arbeit ist der Betreuer Ivan Bütler, welcher zur internen Projektorganisation gehört.

Phasen und deren Iterationen

Diese Arbeit wird in vier Phasen aufgeteilt. Jede Phase enthält mehrere Iterationen, welche in Sprints abgearbeitet werden. Die Besprechungen mit dem Betreuer werden jeweils auf einen Donnerstag fallen. Meilensteine werden jeweils auf einen Mittwoch terminiert, damit die Fortschritte anschliessend besprochen werden können. Alle Iterationen entsprechen einem Sprint mit einer Länge von sieben Tagen. Einzige Ausnahmen sind die Iterationen 0, 1 und 13, welche aus administrativen Gründen verlängert, respektive verkürzt wurden.

Inception

Mit Iteration 0 startet die Studienarbeit bereits in der Woche vor Semesterstart mit einem Meeting, um sich untereinander besser kennenzulernen. Die Aufgabenstellung wird jedoch erst in Iteration 1 ausgegeben. Ziel ist es, dass sich das Team ein Grundverständnis zu MISP aneignet. Ebenfalls wird das neu erlernte Wissen direkt in einer erstellten MISP Lab-Umgebung angewandt.

Iteration	Start	Ende	Beschreibung
0	15.09.2021	19.09.2021	Kickoff Event, kennenlernen des Teams
1	20.09.2021	29.09.2021	Ausgabe der Aufgabenstellung, Einlesen ins Thema, Aufbau von Infrastruktur und MISP Test Lab-Umgebung

Tabelle 4.2: Inception-Phase

Elaboration

In der Elaboration Phase wird das Projekt selbst durchgeplant, sowie Research zu MISP selbst betrieben und dokumentiert. Verschiedenen Technologien werden in einem nächsten Schritt evaluiert, sodass in der nächsten Phase die Labs erstellt werden können.

Iteration	Start	Ende	Beschreibung
2	30.09.2021	06.10.2021	Projektplan und Risikoanalyse erstellen
3	07.10.2021	13.10.2021	Research zu MISP
4	14.10.2021	20.10.2021	Technologien evaluieren & Stories definieren

Tabelle 4.3: Elaboration-Phase

Construction

Die eigentliche Erstellung der Labs wird in dieser Phase durchgeführt. In Iteration 5 wird zunächst die verschiedenen Setups erstellt, welche der Übungsteilnehmer während dem Lösen der Aufgaben im Hacking Lab benötigt. Das MISP Lab selbst wird anschliessend in Iteration 6 bis 9 iterativ erstellt. Ziel ist es, ein Lab abzuschliessen, bevor mit dem nächsten begonnen wird. So kann sichergestellt werden, dass die erledigte Arbeit sicher verwendet werden kann. Zum Schluss jeder Iteration werden direkt Tests mit Nutzern durchführt, sodass Feedback zeitnah analysiert und in die Arbeit einfliessen kann.

Iteration	Start	Ende	Beschreibung
5	21.10.2021	27.10.2021	Verschiedene Setups erstellen
6	28.10.2021	03.11.2021	Erstellen & testen des 1. MISP Lab
7	04.11.2021	10.11.2021	Erstellen & testen des 2. & 3. MISP Lab
8	11.11.2021	17.11.2021	Erstellen & testen des 4. & 5. MISP Lab
9	18.11.2021	24.11.2021	Erstellen & testen des 6. & 7. MISP Lab
10	25.11.2021	01.12.2021	Erstellen & testen des 8. MISP Lab

Tabelle 4.4: Construction-Phase

Transition

Aufgrund der Ergebnisse der Risikoanalyse wurde eine Woche Puffer eingeplant, sodass auf Probleme bei der Implementation der Labs reagiert werden kann. Zusätzlich wird die Dokumentation abgeschlossen. Ziel ist es, dass in Iteration 12 und 13 die Labs bereits im Modul Cyber Defense an der OST eingesetzt werden können. In Iteration 13 müssen diverse Dokumente abgegeben werden.

Iteration	Start	Ende	Beschreibung
11	02.12.2021	08.12.2021	Puffer
12	09.12.2021	15.12.2021	Dokumentation finalisieren
13	16.12.2021	24.12.2021	Abstract verfassen sowie Abgabe

Tabelle 4.5: Transition-Phase

Meilensteine

Um die Ziele dieser Arbeit zu erreichen wurden Meilensteine definiert und zeitlich terminiert.

Bezeichnung	Deadline	Beschreibung
M1 - Grundverständnis	29.09.2021	Das Team weiss, was MISP ist/kann
M2 - Projektplan	06.10.2021	Projektplan erstellt
M3 - Research	20.10.2021	Research & Stories wurden ausarbeitet
M4 - Lab 1	03.11.2021	1. Lab ist fertig und kann ausgeliefert werden
M5 - Lab 2 - 5	17.11.2021	2. bis 5. Lab sind fertig und können ausgeliefert werden
M6 - MISP Labs	01.12.2021	Alle MISP Labs sind fertig und können ausgeliefert werden
M7 - Abgabe	24.12.2021	Finale Auslieferung des Produktes sowie der Dokumentation

Tabelle 4.6: Übersicht aller Meilensteine

Gantt Diagramm

Basierend auf vorherigen Phasen und deren Iterationen soll folgendes Gantt Diagramm den Zeitplan des Projektes zur Verständlichkeit besser visualisieren.

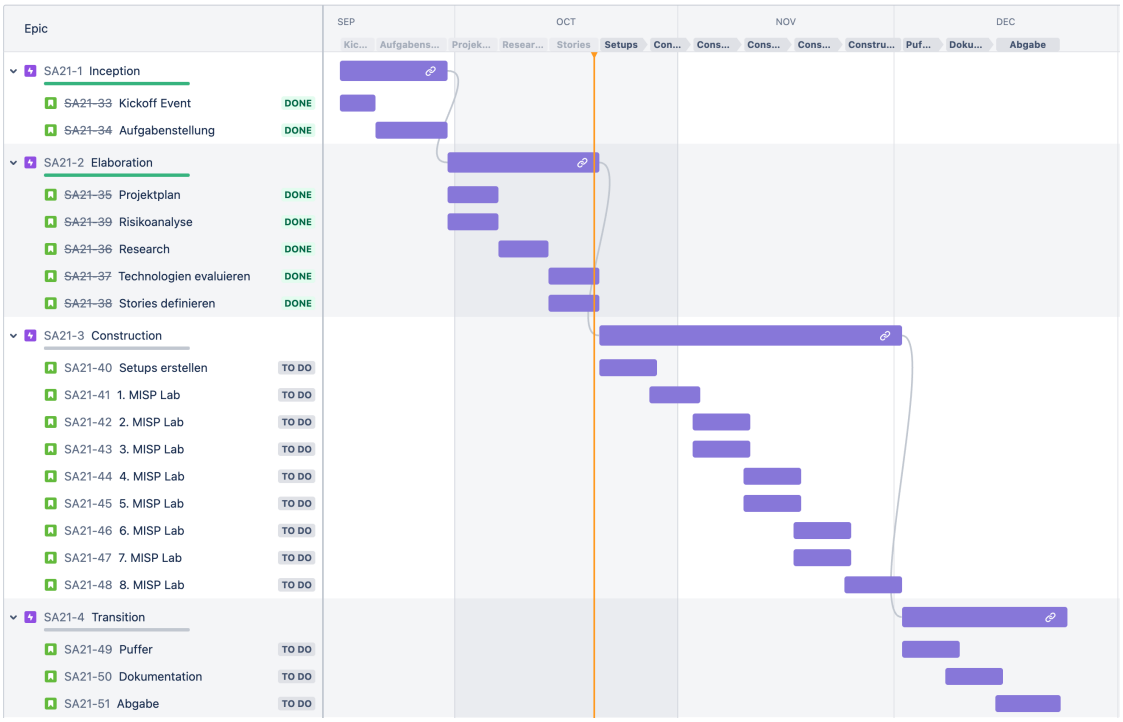

Abbildung 4.4: Gantt Diagramm

Zeiterfassung

Für die Zeiterfassung wird mit der Webapplikation Clockify verwendet. Vorteil ist die Integration von Clockify mit Jira und GitLab, welche ebenfalls für das Projekt verwendet werden. Im Team hat bisher noch keiner Erfahrungen mit dem Tool gemacht. Trotzdem wurde entschieden, dass das Tool verwendet wird, da das Tool sehr beliebt ist und einen guten Eindruck macht. Um die aufgewendete Zeit bei Projektabschluss besser analysieren zu können werden folgende Tags eingeführt:

- Administratives
  In dieser Kategorie wird die Zeit gebucht, welche für das Erstellen von Infrastruktur (Server, GitLab, Jira, etc.) aufgewendet wird.
- Dokumentation
  Sämtliche Zeit, welche für Arbeiten an der Dokumentation wird unter diesem Tag verbucht.
- Implementation
  Das erstellen der Labs selber wird unter dem Tag Implementation erfasst.
- Meeting
  Interne Meetings sowie Besprechungen mit dem Betreuer werden unter dieser Kategorie gebucht.
- Research
  Zeit die für die Informationsbeschaffung aufgewendet wird ist in dieser Kategorie erfasst.
- Testing
  Tests mit externen Personen sowie deren Auswertung wird unter diesem Tag erfasst.

Besprechungen

Das Team arbeitet jeweils montags und freitags am Projekt. An diesen Tagen werden auch Teaminterne Besprechungen gehalten, an welchen Unklarheiten diskutiert und geklärt, sowie offene Aufgaben mit Deadlines verteilt werden.

Donnerstags findet jeweils eine Besprechung mit dem Betreuer Ivan Bütler statt, an welcher der Fortschritt präsentiert und weiteres Vorgehen kurz besprochen wird. Dieses Meeting findet jeweils um 11:00 Uhr im Sitzungsraum 8.225 (Forschungszentrum) statt, sofern nicht anders abgemacht.

Alle Besprechungen werden jeweils dokumentiert und anschliessend in einem Protokoll niedergeschrieben. Um die Qualität der Protokolle zu gewährleisten, werden diese nach vier-Augen-Prinzip vor dem versenden geprüft. Zum Schluss jedes Meetings werden die Aufgaben, welche die Teilnehmer zu erledigen haben nochmals kurz besprochen und ebenfalls in das Protokoll übernommen.

Risikomanagement

Eine genaue Auflistung aller identifizierten Risiken ist im Abschnitt *Risikoanalyse* ersichtlich.

Umgang mit Risiken

Um die Risiken so gut wie möglich zu umgehen ist bei der Risikoanalyse pro Risiko jeweils eine Vorbeugung- und Eintrittsstrategie definiert worden. So kann im optimalen Fall das Risiko ganz umgangen werden. Beim Eintritt ist aber auch bereits eine Strategie zur Lösung vorhanden. So verlieren wir keine Zeit bei der Lösungssuche.

Zeitreserven

Wie im Zeitplan ersichtlich sind Pufferzeiten eingerechnet. Phasen welche schwierig einzuschätzen sind haben daher mehr Zeitreserven und erlauben dadurch mehr Spielraum und Sicherheit in der Planung.

Risikoüberwachung

Die Risiken verändern sich über den Verlauf des Projekts. Es ist darum wichtig, die Risiken immer wieder zu aktualisieren. In diesem Projekt wird die Risikoanalyse jeweils am Ende eines Meilensteins aktualisiert und die Risiken neu berechnet. Ein Verlauf dieser Aktualisierungen findet sich im Kapitel Risikoanalyse.

Arbeitspakete

Um die Arbeitspakte im Team sinnvoll, effizient und dokumentiert aufzuteilen verwenden wir JIRA in Kombination mit Gitlab. Jira wird verwendet um die gesamte Projektplanung zu organisieren. Die Arbeitspakete selbst werden im Gitlab als Issues erfasst. So werden diese dann einem Teammitglied zugewiesen. Nach Absprache kann auch die verantwortliche Person gewechselt werden.

Infrastruktur

In diesem Projekt werden diverse Komponenten an Infrastruktur verwendet. Für Dokumentation und Testing verwenden wir persönliche Laptops. Als IDE setzen wir hauptsächlich VS Code ein.
Zusätzlich verwenden wir zwei virtuelle Linux Server der OST. Auf dem einen Server läuft ein Gitlab Runner und auf dem anderen eine MISP Instanz für diverse Tests.

CI/CD

Unsere Repositories sind auf dem OST Gitlab abgelegt. Wir verwenden darum für das Kompilieren der LaTeX Dokumentation eine Gitlab Pipeline. Nach jedem Push in das Repository wird die Dokumentation kompiliert und als Artefakt abgespeichert. So findet sich im Gitlab immer eine aktuelle PDF Version.
Mögliche Erweiterungen der Pipeline schliessen wir nicht aus und können im Verlaufe des Projekte vorkommen. Aus persönlicher Erfahrung in einem anderen Projekt verzichten wir auf die Benutzung des Shared Runners der OST. Ein eigener Runner lauft darum auf einem der beiden virtuellen Linux Server der OST. So sind wir sicher, dass der Runner zuverlässig und stabil läuft.

Test Infrastruktur

Das finale Produkt dieser Arbeit wird schlussendlich im Hacking Lab zur Verfügung gestellt. Aus diesem Grund besteht unsere Testplatform auch aus dem Hacking Lab selbst. Die Tests lassen sich zu einem grossen Teil nicht automatisieren. Der Fokus liegt daher auch mehr auf Usability Tests.

Dokumentation

Die Dokumentation wird in einem Gitlab Repository versioniert und verwaltet. Jedes Thema wird in einem eigenen Branch editiert. Sobald die Änderungen abgeschlossen sind wird ein Merge Request erstellt. Ein anderes Teammitglied prüft den Request dann auf inhaltliche, grammatikalische und sonstige Fehler. Werden keine Fehler dieser Art gefunden wird der Merge Request akzeptiert und in den Master Branch gepusht. Ist die Qualität aber nicht zufriedenstellend werden Kommentare hinterlassen, was verbessert werden muss. Diese Vorschläge werden dann vom Verfasser geprüft, angepasst und erneut zum Review veröffentlicht. Dieser Prozess wird so oft wiederholt, bis der Reviewer den Merge Request akzeptiert.

Kommunikation

Um die Qualität unseres Projekts so hoch wie möglich zu halten, werden in regelmässigen Abständen Sitzungen durchgeführt. Die Sitzung findet gemäss definiertem Plan jeweils am Donnerstag um 11.00 wenn möglich physisch an der OST statt. Teilnehmer dieser offiziellen Sitzung sind Ivan Bütler, Janis Wolf und Marius Zindel.
Bei Bedarf können weitere Sitzungen vereinbart werden. Ist keine Sitzung notwendig wird dies frühzeitig kommuniziert und von allen Teilnehmern bestätigt.
Übrige Kommunikation findet via MS Teams statt.

Projektmanagement

Für die Dokumentation wird ein Gitlab Repository verwendet. Für allfälligen Code wird ein separates Repository angelegt.
Für Organisatorisches verwenden wir Jira in Kombination mit Gitlab Issues.

Testen

Schlussendlich wird unser fertiges Projekt im Hacking Lab für Informatik Studenten zum Lernzweck veröffentlicht. Daher macht es Sinn, dass wir mit dieser Zielgruppe Usability Tests durchführen. Wichtig ist dabei, dass die Testpersonen eine klare Aufgabenstellung haben, was zu erledigen ist.
Feedback dieser Tests wird dann ausgewertet und das Produkt entsprechend optimiert.

4.2 Risikoanalyse

Das Kapitel Risikoanalyse beschreibt alle von uns identifizierten Risiken, welche in unserem Projekt bestehen. Die Risikobewertung wird laufend neu evaluiert und der Verlauf dokumentiert.

- Projekt: MISP SA
- Erstellt: 04.10.2021
- Zuletzt aktualisiert: 22.12.2021
- Autor: Janis Wolf
- Aktueller gewichteter Schaden: 0h

Nr	Titel	Beschreibung	Schaden [h]	Wahrscheinlichkeit	gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten
R01	Anforderungen	Neue und ändernde Anforderungen	25h	40%	10h	Scope in der Planung klar definieren und priorisieren.	Scope den neuen Anforderungen anpassen.
R02	Komplexität	Komplexität der Funktionalität zu hoch.	20h	20%	4h	Komplexität mit Prototypen in der Elaborationsphase abschätzen.	Funktionalität verringern.
R03	Neue Technologien	Fehlendes Know-How von MISP, Azure etc und der externen Komponenten im Projektteam.	30h	10%	3h	Know-How in der Elaborationsphase aneigenen und festigen.	Fehlendes Know-How aufarbeiten.
R04	Kommunikation	Ungenügende Kommunikation im Projektteam.	20h	30%	6h	Wöchentliche Meetings und Zuweisung der Verantwortung.	Zusätzliche Meetings einführen.
R05	Externe Komponenten	Fehlende Dokumentation und Support von externen Komponenten.	40h	30%	12h	Dokumentation und Support bei Evaluierung der Komponenten berücksichtigen.	Andere Komponenten evaluieren und auf diese wechseln oder Anforderungen anpassen.
R06	Kompatibilitäten	Fehlerhafte Kompatibilität zwischen internen und externen Schnittstellen.	30h	50%	15h	Genaues Prüfen der Schnittstellenspezifikation bei der Evaluierung.	Spezifikation erneut prüfen und interne Schnittstelle anpassen oder wechseln.
R07	Go-Live	Live Umgebung entspricht nicht den Anforderungen.	60h	20%	12h	Anforderungen genau definieren und umsetzten.	Fehlerhafte Komponenten anpassen.
R08	Datenverlust	Verlust von projektrelevanten Daten durch Hardware oder Anwendungsfehler.	60h	5%	3h	Persönliche Computer regelmässig sichern und zur Versionierung Gitlab verwenden.	Daten aus Backup zurückholen.
R09	Überschätzung	Die zu erledigende Arbeit wird mit zu tiefem Aufwand eingeschätzt.	30h	10%	3h	Anforderungen sauber und genau definieren. Pufferzeiten einplanen.	Zeitaufwand durch eingeplante Pufferzeiten ausgleichen.
R10	Krankheit, Unfall	Durch die aktuelle Pandemiesituation können Ausfälle krankheitshalber nicht ausgeschlossen werden.	240h	5%	12h	Empfehlungen bezüglich Gesundheit bestmöglich einhalten.	Anforderungen des Projekts anpassen.
	Total:				**92h**		

Tabelle 4.7: Risikoanalyse (Stand: Projektstart)

Nr	Titel	Beschreibung	Schaden [h]	Wahrscheinlichkeit	gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten
R01	Anforderungen	Neue und ändernde Anforderungen	25h	32%	8h	Scope in der Planung klar definieren und priorisieren.	Scope den neuen Anforderungen anpassen.
R02	Komplexität	Komplexität der Funktionalität zu hoch.	20h	20%	4h	Komplexität mit Prototypen in der Elaborationsphase abschätzen.	Funktionalität verringern.
R03	Neue Technologien	Fehlendes Know-How von MISP, Azure etc und der externen Komponenten im Projektteam.	30h	40%	12h	Know-How in der Elaborationsphase aneigenen und festigen.	Fehlendes Know-How aufarbeiten.
R04	Kommunikation	Ungenügende Kommunikation im Projektteam.	20h	30%	6h	Wöchentliche Meetings und Zuweisung der Verantwortung.	Zusätzliche Meetings einführen.
R05	Externe Komponenten	Fehlende Dokumentation und Support von externen Komponenten.	40h	30%	12h	Dokumentation und Support bei Evaluierung der Komponenten berücksichtigen.	Andere Komponenten evaluieren und auf diese wechseln oder Anforderungen anpassen.
R06	Kompatibilitäten	Fehlerhafte Kompatibilität zwischen internen und externen Schnittstellen.	30h	30%	9h	Genaues Prüfen der Schnittstellenspezifikation bei der Evaluierung.	Spezifikation erneut prüfen und interne Schnittstelle anpassen oder wechseln.
R07	Go-Live	Live Umgebung entspricht nicht den Anforderungen.	60h	20%	12h	Anforderungen genau definieren und umsetzten.	Fehlerhafte Komponenten anpassen.
R08	Datenverlust	Verlust von projektrelevanten Daten durch Hardware oder Anwendungsfehler.	60h	5%	3h	Persönliche Computer regelmässig sichern und zur Versionierung Gitlab verwenden.	Daten aus Backup zurückholen.
R09	Überschätzung	Die zu erledigende Arbeit wird mit zu tiefem Aufwand eingeschätzt.	30h	10%	3h	Anforderungen sauber und genau definieren. Pufferzeiten einplanen.	Zeitaufwand durch eingeplante Pufferzeiten ausgleichen.
R10	Krankheit, Unfall	Durch die aktuelle Pandemiesituation können Ausfälle krankheitshalber nicht ausgeschlossen werden.	150h	4%	6h	Empfehlungen bezüglich Gesundheit bestmöglich einhalten.	Anforderungen des Projekts anpassen.
	Total:				**75h**		

Tabelle 4.8: Risikoanalyse (Meilenstein 3)

Nr	Titel	Beschreibung	Schaden [h]	Wahrscheinlichkeit	gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten
R01	Anforderungen	Neue und ändernde Anforderungen	25h	10%	2.5h	Scope in der Planung klar definieren und priorisieren.	Scope den neuen Anforderungen anpassen.
R02	Komplexität	Komplexität der Funktionalität zu hoch.	20h	20%	4h	Komplexität mit Prototypen in der Elaborationsphase abschätzen.	Funktionalität verringern.
R03	Neue Technologien	Fehlendes Know-How von MISP, Azure etc und der externen Komponenten im Projektteam.	30h	30%	9h	Know-How in der Elaborationsphase aneigenen und festigen.	Fehlendes Know-How aufarbeiten.
R04	Kommunikation	Ungenügende Kommunikation im Projektteam.	20h	20%	4h	Wöchentliche Meetings und Zuweisung der Verantwortung.	Zusätzliche Meetings einführen.
R05	Externe Komponenten	Fehlende Dokumentation und Support von externen Komponenten.	40h	30%	12h	Dokumentation und Support bei Evaluierung der Komponenten berücksichtigen.	Andere Komponenten evaluieren und auf diese wechseln oder Anforderungen anpassen.
R06	Kompatibilitäten	Fehlerhafte Kompatibilität zwischen internen und externen Schnittstellen.	30h	25%	7.5h	Genaues Prüfen der Schnittstellenspezifikation bei der Evaluierung.	Spezifikation erneut prüfen und interne Schnittstelle anpassen oder wechseln.
R07	Go-Live	Live Umgebung entspricht nicht den Anforderungen.	60h	15%	9h	Anforderungen genau definieren und umsetzten.	Fehlerhafte Komponenten anpassen.
R08	Datenverlust	Verlust von projektrelevanten Daten durch Hardware oder Anwendungsfehler.	60h	5%	3h	Persönliche Computer regelmässig sichern und zur Versionierung Gitlab verwenden.	Daten aus Backup zurückholen.
R09	Überschätzung	Die zu erledigende Arbeit wird mit zu tiefem Aufwand eingeschätzt.	30h	5%	1.5h	Anforderungen sauber und genau definieren. Pufferzeiten einplanen.	Zeitaufwand durch eingeplante Pufferzeiten ausgleichen.
R10	Krankheit, Unfall	Durch die aktuelle Pandemiesituation können Ausfälle krankheitshalber nicht ausgeschlossen werden.	120h	4%	4.8h	Empfehlungen bezüglich Gesundheit bestmöglich einhalten.	Anforderungen des Projekts anpassen.
	Total:				**55.3h**		

Tabelle 4.9: Risikoanalyse (Meilenstein 4)

Nr	Titel	Beschreibung	Schaden [h]	Wahrscheinlichkeit	gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten
R01	Anforderungen	Neue und ändernde Anforderungen	25h	5%	1.25h	Scope in der Planung klar definieren und priorisieren.	Scope den neuen Anforderungen anpassen.
R02	Komplexität	Komplexität der Funktionalität zu hoch.	20h	10%	2h	Komplexität mit Prototypen in der Elaborationsphase abschätzen.	Funktionalität verringern.
R03	Neue Technologien	Fehlendes Know-How von MISP, Azure etc und der externen Komponenten im Projektteam.	30h	10%	3h	Know-How in der Elaborationsphase aneigenen und festigen.	Fehlendes Know-How aufarbeiten.
R04	Kommunikation	Ungenügende Kommunikation im Projektteam.	20h	5%	1h	Wöchentliche Meetings und Zuweisung der Verantwortung.	Zusätzliche Meetings einführen.
R05	Externe Komponenten	Fehlende Dokumentation und Support von externen Komponenten.	40h	15%	6h	Dokumentation und Support bei Evaluierung der Komponenten berücksichtigen.	Andere Komponenten evaluieren und auf diese wechseln oder Anforderungen anpassen.
R06	Kompatibilitäten	Fehlerhafte Kompatibilität zwischen internen und externen Schnittstellen.	30h	10%	3h	Genaues Prüfen der Schnittstellenspezifikation bei der Evaluierung.	Spezifikation erneut prüfen und interne Schnittstelle anpassen oder wechseln.
R07	Go-Live	Live Umgebung entspricht nicht den Anforderungen.	60h	10%	6h	Anforderungen genau definieren und umsetzten.	Fehlerhafte Komponenten anpassen.
R08	Datenverlust	Verlust von projektrelevanten Daten durch Hardware oder Anwendungsfehler.	60h	5%	3h	Persönliche Computer regelmässig sichern und zur Versionierung Gitlab verwenden.	Daten aus Backup zurückholen.
R09	Überschätzung	Die zu erledigende Arbeit wird mit zu tiefem Aufwand eingeschätzt.	30h	0%	0h	Anforderungen sauber und genau definieren. Pufferzeiten einplanen.	Zeitaufwand durch eingeplante Pufferzeiten ausgleichen.
R10	Krankheit, Unfall	Durch die aktuelle Pandemiesituation können Ausfälle krankheitshalber nicht ausgeschlossen werden.	90h	4%	3.6h	Empfehlungen bezüglich Gesundheit bestmöglich einhalten.	Anforderungen des Projekts anpassen.
	Total:				**28.85h**		

Tabelle 4.10: Risikoanalyse (Meilenstein 5)

Nr	Titel	Beschreibung	Schaden [h]	Wahrscheinlichkeit	gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten
R01	Anforderungen	Neue und ändernde Anforderungen	25h	2%	0.5h	Scope in der Planung klar definieren und priorisieren.	Scope den neuen Anforderungen anpassen.
R02	Komplexität	Komplexität der Funktionalität zu hoch.	20h	0%	0h	Komplexität mit Prototypen in der Elaborationsphase abschätzen.	Funktionalität verringern.
R03	Neue Technologien	Fehlendes Know-How von MISP, Azure etc und der externen Komponenten im Projektteam.	30h	0%	0h	Know-How in der Elaborationsphase aneigenen und festigen.	Fehlendes Know-How aufarbeiten.
R04	Kommunikation	Ungenügende Kommunikation im Projektteam.	20h	2%	0.4h	Wöchentliche Meetings und Zuweisung der Verantwortung.	Zusätzliche Meetings einführen.
R05	Externe Komponenten	Fehlende Dokumentation und Support von externen Komponenten.	40h	0%	0h	Dokumentation und Support bei Evaluierung der Komponenten berücksichtigen.	Andere Komponenten evaluieren und auf diese wechseln oder Anforderungen anpassen.
R06	Kompatibilitäten	Fehlerhafte Kompatibilität zwischen internen und externen Schnittstellen.	30h	0%	0h	Genaues Prüfen der Schnittstellenspezifikation bei der Evaluierung.	Spezifikation erneut prüfen und interne Schnittstelle anpassen oder wechseln.
R07	Go-Live	Live Umgebung entspricht nicht den Anforderungen.	60h	0%	0h	Anforderungen genau definieren und umsetzten.	Fehlerhafte Komponenten anpassen.
R08	Datenverlust	Verlust von projektrelevanten Daten durch Hardware oder Anwendungsfehler.	60h	0%	0h	Persönliche Computer regelmässig sichern und zur Versionierung Gitlab verwenden.	Daten aus Backup zurückholen.
R09	Überschätzung	Die zu erledigende Arbeit wird mit zu tiefem Aufwand eingeschätzt.	30h	0%	0h	Anforderungen sauber und genau definieren. Pufferzeiten einplanen.	Zeitaufwand durch eingeplante Pufferzeiten ausgleichen.
R10	Krankheit, Unfall	Durch die aktuelle Pandemiesituation können Ausfälle krankheitshalber nicht ausgeschlossen werden.	60h	4%	2.4h	Empfehlungen bezüglich Gesundheit bestmöglich einhalten.	Anforderungen des Projekts anpassen.
	Total:				**3.3h**		

Tabelle 4.11: Risikoanalyse (Meilenstein 6)

Nr	Titel	Beschreibung	Schaden [h]	Wahrscheinlichkeit	gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten
R01	Anforderungen	Neue und ändernde Anforderungen	25h	0%	0h	Scope in der Planung klar definieren und priorisieren.	Scope den neuen Anforderungen anpassen.
R02	Komplexität	Komplexität der Funktionalität zu hoch.	20h	0%	0h	Komplexität mit Prototypen in der Elaborationsphase abschätzen.	Funktionalität verringern.
R03	Neue Technologien	Fehlendes Know-How von MISP, Azure etc und der externen Komponenten im Projektteam.	30h	0%	0h	Know-How in der Elaborationsphase aneigenen und festigen.	Fehlendes Know-How aufarbeiten.
R04	Kommunikation	Ungenügende Kommunikation im Projektteam.	20h	0%	0h	Wöchentliche Meetings und Zuweisung der Verantwortung.	Zusätzliche Meetings einführen.
R05	Externe Komponenten	Fehlende Dokumentation und Support von externen Komponenten.	40h	0%	0h	Dokumentation und Support bei Evaluierung der Komponenten berücksichtigen.	Andere Komponenten evaluieren und auf diese wechseln oder Anforderungen anpassen.
R06	Kompatibilitäten	Fehlerhafte Kompatibilität zwischen internen und externen Schnittstellen.	30h	0%	0h	Genaues Prüfen der Schnittstellenspezifikation bei der Evaluierung.	Spezifikation erneut prüfen und interne Schnittstelle anpassen oder wechseln.
R07	Go-Live	Live Umgebung entspricht nicht den Anforderungen.	60h	0%	0h	Anforderungen genau definieren und umsetzten.	Fehlerhafte Komponenten anpassen.
R08	Datenverlust	Verlust von projektrelevanten Daten durch Hardware oder Anwendungsfehler.	60h	0%	0h	Persönliche Computer regelmässig sichern und zur Versionierung Gitlab verwenden.	Daten aus Backup zurückholen.
R09	Überschätzung	Die zu erledigende Arbeit wird mit zu tiefem Aufwand eingeschätzt.	30h	0%	0h	Anforderungen sauber und genau definieren. Pufferzeiten einplanen.	Zeitaufwand durch eingeplante Pufferzeiten ausgleichen.
R10	Krankheit, Unfall	Durch die aktuelle Pandemiesituation können Ausfälle krankheitshalber nicht ausgeschlossen werden.	60h	0%	0h	Empfehlungen bezüglich Gesundheit bestmöglich einhalten.	Anforderungen des Projekts anpassen.
	Total:				**0h**		

Tabelle 4.12: Risikoanalyse (Meilenstein 7)

Verlauf

Neubeurteilung Meilenstein 2 & 3

Die Risikoanalyse wurde für die Meilensteine 2 und 3 neu evaluiert. Änderungen werden folgend protokolliert. Der total gewichtetet Schaden sinkt von 92 auf 75 Stunden.

R01 - Anforderungen

Die Anforderungen wurden genauer definiert. Die Wahrscheinlichkeit wird daher kleiner. Der gewichtete Schaden wird um 2 Stunden reduziert.

R03 - Neue Technologien

Auch R03 ist nun besser einzuschätzen. Die Wahrscheinlichkeit sinkt von 50% auf 40%. Die reduziert den gewichteten Schaden auf 12 Stunden.

R03 - Kompatibilität

Dieses Risiko kann nun auch dank den genaueren Anforderungen besser eingeschätzt werden. Der gewichtete Schaden sinkt von 15 auf 9 Stunden.

R10 - Krankheit, Unfall

Die Chancen sinken, dass dieses Ereigniss eintritt. Der gewichtete Schaden reduziert sich.

Neubeurteilung Meilenstein 4

Durch erste Erfahrungen mit den Labs und viel Research konnte der gewichtete Schaden weiter verkleinert werden.

Neubeurteilung Meilenstein 5

Der gewichtete Schaden konnte weiterhin verkleinert werden.

Neubeurteilung Meilenstein 6

Da die MISP Labs nun erreicht und abgeschlossen wurde konnte die Eintrittswahrscheinlichkeit bei vielen Punkten minimiert werden. Durch die vergrösserten Anforderungen stieg der zeitliche Aufwand gegen Ende des Projekts noch an. Es wird mit einem Schaden von 34.2 Stunden gerechnet, was der Länge einer Iteration entspricht.

Neubeurteilung Meilenstein 7

Der gewichtete Schaden konnte zum Schluss des Projekts auf 0h minimiert werden. Wir sind froh darüber, dass wir mit den Risiken gut umgehen konnten und unserem Zeitplan mehrheitlich folgen konnten.

4.3 Projektmonitoring

4.3.1 Überblick

In diesem Kapitel wird der Soll- mit dem Ist-Zustand des Projektes verglichen. Dabei werden hauptsächlich auf die Planung der Arbeit eingegangen und was sich im Laufe der Iterationen veränderte.

4.3.2 Meilensteine

Alle Meilensteine wurden grundsätzlich eingehalten. Die Labs wurden erfolgreich im Meilenstein 6 wie geplant fertiggestellt, jedoch gab es aufgrund Änderungen in der Infrastruktur einige Anpassungen, welche vor der Auslieferung umgesetzt werden mussten. In der Planungsphase wurde für solche Fälle ein Puffer in Iteration 11 eingeplant. So konnten die Labs erfolgreich in der folgenden Iteration 12 durch Ivan Bütler im Hacking-Lab final deployed werden.

Der Ivan Bütler machte den Autoren dieser Arbeit das Angebot, das erlernte Wissen in einer 90 minütigen Vorlesung direkt an die Studierenden weiterzugeben. Nach einer kurzen Risikoabschätzung wurde die Iteration 12 neu organisiert und die Finalisierung der Dokumentation musste in Iteration 13 weichen. Aufgrund bereits geleisteter Vorarbeit war dies trotz Mehraufwand zeitlich umsetzbar. Das Angebot konnte so angenommen werden.

Bezeichnung	Deadline	Status
M1 - Grundverständnis	29.09.2021	Meilenstein wurde eingehalten
M2 - Projektplan	06.10.2021	Meilenstein wurde eingehalten
M3 - Research	20.10.2021	Meilenstein wurde eingehalten
M4 - Lab 1	03.11.2021	Meilenstein wurde eingehalten
M5 - Lab 2 - 5	17.11.2021	Meilenstein wurde eingehalten
M6 - MISP Labs	01.12.2021	Meilenstein wurde teilweise eingehalten
M7 - Abgabe	24.12.2021	Meilenstein wurde eingehalten

Tabelle 4.13: Auswertung aller Meilensteine

4.3.3 Zeiterfassung

Allgemein

Für die Arbeit wurde ursprünglich ein Zeitaufwand von 480 Stunden geschätzt. Der effektive Zeitaufwand beträgt 542 Stunden und 29 Minuten.

Durch die zusätzliche angenommene Arbeit für das Vorbereiten und Halten der Vorlesung, sowie dem Betreuen der Übungslektion wurde zusätzlich Zeit beansprucht. Diese beläuft sich auf 37 Stunden. Ebenfalls wurde auf die Qualität der Labs sehr wert gelegt, welche zusätzliche Zeit in der Puffer Iteration beanspruchte.

Zeitaufwand pro Teammitglied

Die aufgewendete Zeit im Vergleich zwischen den Teammitgliedern ist ausgeglichen und akzeptabel.

	Stundenaufwand durchschnittlich pro Woche	Stundenaufwand geleistet
Janis Wolf	18.8h	263h 30min
Marius Zindel	19.9h	278h 59min
Total	38.7h	542h 29min

Tabelle 4.14: Stundenaufwand geleistet

Zeitaufwand pro Kategorie

Die bereits im Kapitel *Zeiterfassung* erwähnten Tags wurden um eine weitere Kategorie Präsentation ergänzt. Darin sind alle Arbeiten enthalten welche für das Vorbereiten und Halten der Vorlesung, sowie dem Betreuen der Übungslektionen beansprucht wurde.

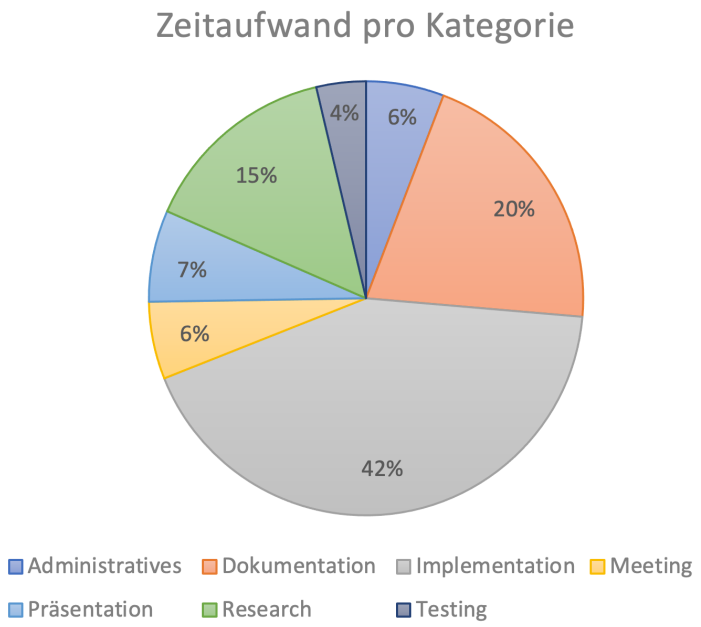

Abbildung 4.5: Zeitaufwand pro Kategorie

Kategorie	Aufwand in Stunden
Administratives	31.5
Dokumentation	111.5
Implementation	231.2
Meeting	31.3
Präsentation	37.0
Research	80.0
Testing	20.0

Tabelle 4.15: Zeitaufwand pro Kategorie

Auffallend ist, dass die Kategorie Implementation bei weitem den grössten Teil der Arbeit ausmacht. Die Analyse ist jedoch nur mit Vorsicht zu bewerten, da bei der Implementation der Labs immer auch indirekt ein Teil an Research dabei war.
Das Kapitel Dokumentation hingegen viel relativ klein aus. Aufgrund von Vorarbeiten aus anderen Projekten konnte die Struktur der Dokumente grösstenteils wiederverwendet werden.

4.3.4 Zeitaufwand pro Iteration

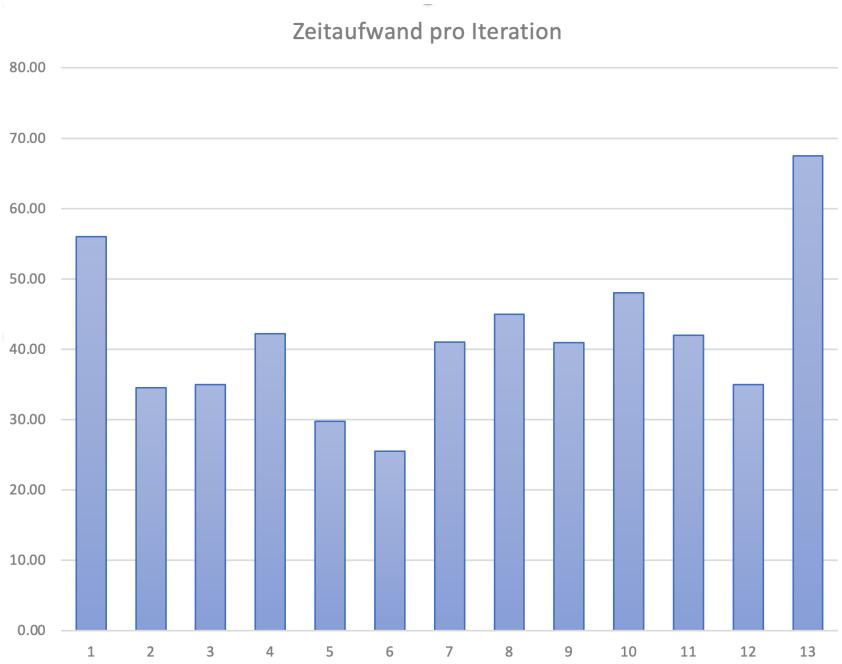

Abbildung 4.6: Zeitaufwand pro Iteration

Iteration	Aufwand in Stunden
1	56.0
2	34.5
3	35.0
4	42.2
5	29.7
6	25.5
7	41.0
8	45.0
9	40.9
10	48.0
11	42.0
12	35.0
13	67.5

Tabelle 4.16: Zeitaufwand pro Iteration

An die Grafik ist anzumerken, dass die Iteration 1 sowie 13 länger als die restlichen Iterationen dauern. Dies wurde aus organisatorischen Gründen so gewählt und im Abschnitt *Phasen und deren Iterationen* dokumentiert.

In Iteration 6 waren alle Teammitglieder privat sehr ausgelastet und konnten so kaum am Projekt weiterarbeiten. Dies wurde jedoch bei der Planung berücksichtigt und konnte in den folgenden Iterationen kompensiert werden. Der darauf folgende Meilenstein konnte trotzdem eingehalten werden.

4.4 Persönliche Berichte

In diesem Kapitel werden die Autoren der Arbeit rückblickend auf den Verlauf der Arbeit einen kurzen persönlichen Bericht abliefern.

4.4.1 Janis Wolf

Der Start der Semesterarbeit erschien mir etwas schwierig, da ich mir vom vorjährigen Engineering Projekt eine sehr klare Struktur und Vorgabe gewöhnt war. Nachdem dann aber die ersten Unklarheiten geklärt wurden konnten wir mit der Arbeit beginnen. Bzw. wir haben mit viel Research und Tests gestartet. Ich denke diese Phase hätte sogar fast noch etwas länger sein können. Teilweise habe ich mitten im Entwickeln eines Labs eine neue Funktion gefunden, von welcher ich noch nie gehört habe.
Nach Zeitplan zu Arbeiten hat im Rückblick gut funktioniert. Ich bin aber froh, dass wir Pufferzonen eingeplant haben. So konnten wir gegen Ende der Arbeit all unsere Abgaben pünktlich einreichen.
Die Zusammenarbeit mit Ivan Bütler als Betreuer hat sehr gut funktioniert. Die wöchentlichen Meetings waren angebracht und die Inputs hilfreich.
Es war auch spannend eine Vorlesung halten zu dürfen.

Schlussendlich war die Semesterarbeit anstrengend, aber ich bin froh, dass ich nun ein Tool wie MISP kennengelernt habe. Vielleicht trifft man dieses in der Zukunft ja mal wieder an und ich denke dann hilft das gut fundierte Wissen, welches wir uns angeeignet haben, sicher weiter.

4.4.2 Marius Zindel

Die Studienarbeit stellte uns Herausforderungen, welche anders nicht während einer Ausbildung erlernt werden können. Selbständigkeit und Eigenwille wurde von Beginn bis zum Schluss wie noch nie zuvor in einem Projekt gefordert. In die Studienarbeit starteten wir ohne Vorwissen zu MISP und mussten und dieses uns zuerst in den ersten Wochen selbst aneignen. Zwar mit mehr Wissen, jedoch immer noch vielen Unklarheiten starteten wir mit der Implementierung der Übungsaufgaben. Umso länger ich mich mit MISP beschäftigte, umso mehr verstand ich alle Zusammenhänge. Unser Betreuer Ivan Bütler liess uns dabei viele Freiheiten. So konnte ich meine Kreativität völlig entfalten und die Challenges so gestalten, wie ich sie selbst interessant finden würde.

Schwierigkeiten hatte ich beim Implementieren der ersten Labs, praxisnahe Beispiele über die Platform zu finden. Im Nachhinein würde ich so zum Start des Projektes einen Experten interviewen, welcher Hinweise zu den "best practices" weitergeben kann. Der zu Beginn definierte Projektplan unterstützte mich sehr. Stets ein Fahrplan sowie ein Ziel vor Augen zu haben bestärkte mich enorm beim Arbeiten. Der Zeitplan war eher straff getaktet, was wiederum einige Freinächte forderte. Dies lohnte sich aber auf jeden Fall.

Die Vorlesung vor den Studierenden zu halten war für mich persönlich der krönende Abschluss dieser Arbeit. Es bereitete mir einerseits Freude die wichtigsten Themen zu MISP der Klasse zu präsentieren andererseits war es auch eine Herausforderung auf die kritischen Fragen möglichst einfache und präzise Erklärungen zu liefern.

Rückblickend konnte ich während der Studienarbeit etliche neuen Erfahrungen in diversen Bereichen sammeln. Die Arbeit werte ich daher aus persönlicher wie auch aus Sicht des Projektes als erfolgreich.

KAPITEL 5

Verzeichnisse

5.1 Glossar

Begriff	Erklärung
API	Application Programming Interface
Bitcoin	Kryptowährung
CIRCL	Organisation welche MISP entwickelt
CSV	Textbasiertes Dateiformat für strukturierte Daten
Curl	Command Line Tool um Webseiten per Konsole abzurufen
Docker-compose	Datei welche mehrere Docker Container in einem Paket verbindet
Dockerfile	Bau- und Konfigurationsdatei für ein Docker Image
Environment Variablen	Umgebungsvariablen
Flag (bezogen auf HL)	Token welches als Lernkontrolle im HL verwendet werden kann
GDPR	General Data Protection Regulation
Hacking Lab	Auch HL genannt, eine Cyber Range Umgebung
IOC	Indicator of Compromise
JSON	Dateiformat für strukturierte Daten
Jump Host	Computer welcher zum Verbinden zu einem anderen Host verwendet wird
Live CD	offizielle virtuelle Maschine vom Hacking Lab Team
Markdown	Einfach Markup Sprache die für simple Dokument
MISP	Malware Information Sharing Platform
MS Teams	Microsoft Teams (Kommunikationssoftware)
Network Intrusion Detection System	Auch IDS genannt, detektiert Einbrüche in ein Netzwerk
OCR	Optical character recognition (Texterkennung via Kamera)
OST	Ostschweizer Fachhochschule
Open Source	Code welcher zur freien Verfügung
PyMISP	Python Library welche auf die MISP API zugreift
REST	Representational state transfer (Protokoll für API Calls)
SOC	Security Operation Center (Zentrale für IT Sicherheit in einer Firma)
Security Information and Event Management	Verwaltung von Sicherheitsinformationen und Angriffe
Terraform	Infrastructure as code software tool
Threat Intelligence Sharing	Teilen von Informationen bei Angriffen
VM	Virtuelle Maschine
VS Code	Ausgebauter Texteditor von Microsoft

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] OST. (2021, 12) Studienschwerpunkte ost. [Online]. Available: https://www.ost.ch/de/studium/informatik/bachelor-informatik/studieninhalt-und-aufbau/studienschwerpunkte

[2] S. C. GmbH. (2021, 12) Hacking-lab livecd. [Online]. Available: https://livecd.hacking-lab.com/

[3] Leaderdigital. (2021, 12) Die gefahr von cyberangriffen wird weiter steigen. [Online]. Available: https://www.leaderdigital.ch/news/die-gefahr-von-cyberangriffen-wird-weiter-steigen-6085.html

[4] CIRCL. (2021, 12) Dokumentation misp. [Online]. Available: https://www.circl.lu/doc/misp

[5] ——. (2021, 12) Github repository misp. [Online]. Available: https://github.com/MISP/MISP

[6] ——. (2021, 12) Dokumentation misp features. [Online]. Available: https://www.misp.software/features.html

[7] ——. (2021, 12) Dokumentation misp objekte. [Online]. Available: https://www.circl.lu/doc/misp/misp-objects/

[8] ——. (2021, 12) Dokumentation misp synchronisation. [Online]. Available: https://www.circl.lu/doc/misp/sharing/

[9] ——. (2021, 12) Dokumentation misp taxonomie. [Online]. Available: https://www.circl.lu/doc/misp/misp-taxonomy/

[10] ——. (2021, 12) Misp galaxies. [Online]. Available: https://www.circl.lu/doc/misp/galaxy/

[11] ——. (2021, 12) Dokumentation misp sightings. [Online]. Available: https://www.circl.lu/doc/misp/sightings/

[12] ——. (2021, 12) Dokumentation misp warninglists. [Online]. Available: https://www.circl.lu/doc/misp/warninglists/

[13] ——. (2021, 12) Dokumentation misp noticelists. [Online]. Available: https://www.circl.lu/doc/misp/noticelists/

[14] ——. (2021, 10) Misp api documentaion. [Online]. Available: https://www.circl.lu/doc/misp/automation/

[15] ——. (2021, 12) Dokumentation misp pymisp. [Online]. Available: https://github.com/MISP/PyMISP

[16] ——. (2021, 12) Misp docker repository. [Online]. Available: https://github.com/misp/misp-docker

[17] J. Wolf and M. Zindel. (2021, 12) Hacking lab docker repo. [Online]. Available: https://github.com/Hacking-Lab/misp-docker-image/

[18] ——. (2021, 12) docker-compose datei misp github. [Online]. Available: https://github.com/Hacking-Lab/misp-docker-image/blob/main/docker-compose.yml

[19] S. Monkey. (2021, 12) Survey monkey website. [Online]. Available: https://surveymonkey.com

[20] T. Kälin, "Project planning," *Software Engineering 2*, pp. 16–21, Februar 2021.

KAPITEL 6

Anhang

6.1 Eigenständigkeitserklärung

OST
Ostschweizer
Fachhochschule

Eigenständigkeitserklärung

Erklärung

Wir erklären hiermit,

- dass wir die vorliegende Arbeit selbst und ohne fremde Hilfe durchgeführt haben, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.
- dass wir keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt habe.

Janis Wolf

Marius Zindel

Rapperswil, 23. Dezember 2021

Abbildung 6.1: Eigenständigkeitserklärung

6.2 Nutzungsrechte

OST
Ostschweizer
Fachhochschule

Vereinbarung

1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Studienarbeit «MISP Malware Information Sharing Project» von Marius Zindel und Janis Wolf unter der Betreuung von Ivan Bütler geregelt.

2. Urheberrecht

Die Urheberrechte stehen den Studenten der Studienarbeit zu.

3. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl von den Studenten der Studienarbeit, durch den Betreuer der FH OST wie auch von der Compass Security Holding oder einer ihrer Compass Tochterfirmen (Compass Schweiz, Compass Deutschland, Compass Cyber Defense, Hacking-Lab AG) uneingeschränkt nach Abschluss der Arbeit verwendet und weiterentwickelt werden.

Rapperswil, den 28.10.2021 .....
Der Student Marius Zindel

Rapperswil, den 28.10.2021 .....
Der Student Janis Wolf

Rapperswil, den 28.10.2021 .....
Der Betreuer der Studienarbeit Ivan Bütler

Abbildung 6.2: Nutzungsrechte