Security Information & Event Management (SIEM) für Klein- und Mittelständische Unternehmen (KMU)

Best-Practice-Bericht

im Verbundprojekt SIMU

DOKUMENTINFORMATIONEN	
TYP	Bericht
TITEL	AP5: Best-Practice-Bericht
DATUM	30.11.2015
ARBEITSPAKET	AP5
FÖRDERKENNZEICHEN	16KIS0041K

SIMU-KONSORTIUM	
KOORDINATOR	DECOIT GmbH
PARTNER 1	NCPe GmbH Network Communications Products engineering
PARTNER 2	macmon secure gmbh
PARTNER 3	Hochschule Hannover
PARTNER 4	Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e. V.

DOKUMENTSTATUS		
AKTION	**DURCH**	**DATUM**
EINGEREICHT	DECOIT GmbH	30.09.2015
AP-LEITER	DECOIT GmbH	
GENEHMIGT	DECOIT GmbH	30.11.2015

ÄNDERUNGSSHISTORIE			
DATUM	**VERSION**	**AUTOR**	**KOMMENTAR**
12.10.2015	0.1	Thomas Rix	Dokumentstruktur angelegt
20.10.2015	0.2	Thomas Rix	Inhalte bis auf 4.2.5, 4.3 und 5 eingefügt, bereit zur Weitergabe an Projektpartner
21.10.2015	0.3	Thomas Rix	Fehlende Inhalte 3.1.5, 4 und 4.2.4
22.10.2015	0.4	Thomas Rix	Inhalt für 4.3.3
18.11.2015	0.5	Thomas Rix	Inhalt für 4.3.7, 4.3.8, 4.3.9 und 5
18.11.2015	0.6	Leonard Renners	Inhalt für 4.3.1 und 4.3.2
19.11.2015	0.7	Kai-Oliver Detken	Gesamtes Dokument durchgegangen und erweitert bzw. ergänzt
24.11.2015	0.8	Bastian Hellmann	Inhalt für 4.3.2, 4.3.4 und 4.3.5
25.11.2015	0.9	Leonard Renners	Inhalt für 4.3.6
27.11.2015	1.0	Bastian Hellmann	Kleinere Änderungen
30.11.2015	FINAL	Kai-Oliver Detken	Glossar eingefügt und Dokument abgeschlossen

KONTAKTINFORMATIONEN		
NAME	**ORGANISATION**	**E-MAIL**
Prof. Dr. Kai-Oliver Detken	DECOIT GmbH	detken@decoit.de
Thomas Rix	DECOIT GmbH	rix@decoit.de
Jürgen Westerkamp	macmon secure gmbh	juergen.Westerkamp@macmon.eu
Tim Felser	NCP engineering GmbH	tim.felser@ncp-e.com
Bernhard Walle	NCP engineering GmbH	bernhard.walle@ncp-e.com
Leonard Renners	Hochschule Hannover	leonard.renners@hs-hannover.de
Bastian Hellmann	Hochschule Hannover	bastian.hellmann@hs-hannover.de
Henk Birkholz	Fraunhofer SIT	henk.birkholz@sit.fraunhofer.de

Inhalt

1 Einleitung

Dieses Dokument beschreibt die Erfahrungen und die daraus resultierenden Best-Practice-Richtlinien, welche sich innerhalb des vom Bundesministerium für Bildung und Forschung (BMBF) geförderten Verbundprojektes SIMU im Rahmen der Entwicklungsarbeiten ergeben haben. Außerdem wird aufgezeigt, wie der finale Demonstrator, welcher einen exemplarischen Use-Case mit den Komponenten der Projektpartner darstellt, konzipiert und technisch umgesetzt wurde. Abschließend werden die Erkenntnisse, die durch die Entwicklung des Demonstrators gewonnen wurden, präsentiert und diskutiert.

Innerhalb dieses Dokuments wird in Kapitel 2 zunächst eine kurze, allgemeine Übersicht über die Inhalte und Ziele des SIMU-Projektes gegeben. Im darauf folgenden Kapitel 3 wird das Schlüsselkonzept des Projektes mit den auftretenden Problemen bei der Entwicklung und Implementierung der IF-MAP-Spezifikation aufgezeigt sowie die Ansätze, welche zur Lösung dieser Probleme zur Anwendung kamen.

In Kapitel 4 werden der finale Demonstrator und die durchgeführten Tests beschrieben, welche im Rahmen des Projekts umgesetzt wurden. Dabei werden die Konzepte und eingesetzten Technologien erläutert, auf deren Grundlage die Realisierung des Demonstrators letztendlich erfolgte sowie eine kurze Anleitung zum Aufsetzen und zur Benutzung des Demonstrators gegeben. In Kapitel 5 folgt eine abschließende Zusammenfassung der erzielten Ergebnisse des Projekts.

2 Beschreibung des SIMU-Projekts

Gerade in kleinen und mittelständischen Unternehmen (KMU) wird der zunehmenden Bedrohungslage durch Cyberkriminalität noch nicht ausreichend Rechnung getragen. Dabei spielt der Trend zu stark zunehmender geschäftlicher Nutzung von Smartphones, Tablets und Netbooks eine wichtige Rolle. Laut einer Umfrage von BITKOM erlauben fast 43% der ITK-Unternehmen ihren Mitarbeitern die Nutzung von private Smartphones, Notebooks oder Tablet-Computers. Demgegenüber berücksichtigt fast die Hälfte (42%) der Unternehmen mobile Endgeräte überhaupt nicht in ihren IT-Sicherheitskonzepten (Stand: 2012).

Sicherheitssysteme wie Firewalls, Virenscanner, Spamfilter und VPN-Gateways sind zwar auch bei KMU häufig im Einsatz, arbeiten aber typischerweise isoliert voneinander. Viele Angriffe können jedoch nur durch die Kombination von Daten verschiedenster Systeme erkannt werden. Selbst wenn ein Angriff erkannt wird, erfolgen Gegenmaßnahmen oft zu spät und der Angreifer hat bereits den Betrieb wichtiger Systeme gestört oder sensible Informationen erlangt. Eine kontinuierliche und proaktive Überwachung von IT-Systemen (Client-, Server-, Netzwerk-Komponenten, Firewall etc.) sowie den Vorgängen und Ereignissen im Netz findet meist nicht statt.

Große Unternehmen und Konzerne setzen für diese Überwachung sogenannte „Security Information and Event Management“ (SIEM) Komponenten ein. SIEM-Systeme werden dort mittlerweile als eine wichtige Komponente von Firmennetzen und IT-Infrastrukturen angesehen. SIEM-Systeme erlauben es, Meldungen und Warnungen einzelner Komponenten eines IT-Systems zusammen zu führen und auszuwerten. Hierbei können auch die Meldungen von spezialisierten Sicherheitssystemen (Firewall-Logs, VPN-Gateway-Logs etc.) mit in Betracht gezogen werden. In der Praxis hat es sich aber gezeigt, dass diese SIEM-Systeme äußerst komplex sind und nur durch einen erheblichen personellen Aufwand betreibbar sind. Oft werden daher SIEM-Systeme zwar installiert, im weiteren Betrieb aber vernachlässigt.

Für den Einsatz im KMU-Umfeld sind SIEM-Systeme typischerweise nicht gut geeignet, vor allem aus folgenden Gründen:

1. Hohe Kosten für Einrichtung und Wartung, da neue IT-Infrastrukturkomponenten (Kollektoren) installiert, konfiguriert und gewartet werden müssen.
2. Hohe Kosten für den Betrieb, da umfangreiches Expertenwissen für die Auswertung und richtige Interpretation der Meldungen und Ausgaben eines SIEM-Systems erforderlich ist.
3. Mangelnde Skalierbarkeit auf kleine und mittlere Netze.

Wesentliches Ziel des SIMU-Projektes ist die Entwicklung eines SIEM-artigen Systems zur signifikanten, mit geringem Aufwand erzielbaren Verbesserung der IT-Sicherheit in einem Unternehmensnetzwerk. Neben der leichten Integrierbarkeit in IT-Infrastrukturen von KMU und einer einfachen Nachvollziehbarkeit von relevanten Ereignissen und Vorgängen im Netz soll dies darüber hinaus mit einem geringen

Aufwand für Konfiguration, Betrieb und Wartung realisiert werden können. Funktional soll SIMU ähnlich zu SIEM-Systemen arbeiten, also im wesentlichen Vorgänge und Ereignisse im Firmennetzwerk überwachen und dort, wo es sinnvoll erscheint, automatisiert, in Echtzeit und proaktiv Maßnahmen zur Verbesserung der Sicherheit einleiten.

2.1 SIEM-Definition

Security Information and Event Management (SIEM) Lösungen stellen eine Kombination aus ehemals unterschiedlichen Produktkategorien dar: Security Information Management (SIM) und Security Event Management (SEM). SIEM-Technologie ermöglicht die Echtzeitanalyse von Security-Alarmen, die von Netzwerk-Komponenten oder Anwendungen generiert werden. SIEM-Lösungen kann es als reine Software-Systeme geben, aber auch Appliances und Managed Services sind möglich. Durch die Analyse von Loginformationen können zusammenhängende Reports (Berichte) generiert werden, die man auch für Compliance-Zwecke verwenden kann.

Die Akronyme SEM, SIM und SIEM werden teilweise gleichermaßen verwendet, obwohl es unterschiedliche Bedeutungen und Produktfähigkeiten zu beachten gilt. Der Bereich des Sicherheitsmanagements, der sich mit der Echtzeitüberwachung, Ergebniskorrelation, Eventbenachrichtigungen befasst, wird als Security Event Management (SEM) bezeichnet. Der zweite Bereich ermöglicht Langzeiterfassung, Analyse und Reporting von Logdaten und ist als Security Information Management (SIM) bekannt. Beide Segmente können unterschiedlich kombiniert werden, um je nach Anforderungen und Leistungsfähigkeit ein SIEM-System zusammenzustellen.

Eine grundlegende Aufgabe eines SIEM-Systems ist es also, Informationen aus vielen unterschiedlichen Quellen zu sammeln und Zusammenhänge zwischen diesen zu finden. Für diesen Zweck werden Kollektoren eingesetzt, die die Daten sammeln und aufbereiten, damit das SIEM-System mit diesen arbeiten kann.

Um sicherheitsrelevante Events unterschiedlicher Güte in einem SIEM zusammenzuführen, erbringen sogenannte Kollektoren folgende Leistungen:

1. **Extraktion**: Events sind in Rohform meist Einträge in Log-Dateien oder über das Netz versendete Systemmeldungen. Diese Informationen müssen aus den jeweiligen verwendeten Systemen oder Transportprotokollen extrahiert werden, um sie einem SIEM-System zugänglich zu machen.
2. **Homogenisierung/Mapping**: Events werden von unterschiedlichen Diensten erzeugt und aus unterschiedlichen Systemen extrahiert. Um eine Weiterverarbeitung in einem SIEM-System zu gewährleisten, müssen die relevanten Inhalte der einzelnen Events miteinander in Bezug gebracht werden können. Dafür sorgt ein entsprechendes „Umsortieren“ individueller Datenfelder in speziellen Eventformaten in ein standardisiertes, dem SIEM-System verständliches Eventformat (z.B. IDMEF nach RFC-4765).
3. **Aggregation**: Große Mengen gleichartiger Events über einen kurzen Zeitraum würden ein zentrales SIEM-System belasten, ohne einen signifikanten Mehrwert zu erzeugen. Kollektoren aggregieren daher große

Mengen gleichartiger Events über einen kurzen Zeitraum (sog. Bursts) zu einem einzigen Event mit höherer Aussagekraft (z.B. Event-Typ, Inhalt und Menge der ursprünglichen Meldungen).

Die Auswertung von Events wird anhand von Regelsätzen durchgeführt. Die Relevanz der Ergebnisse der Regelauswertung ist aber abhängig von den Eigenschaften bzw. dem Aufbau des jeweiligen Unternehmens. Beispiele hierfür sind primäre und sekundäre Geschäftsprozesse, organisatorische Prozesse, die Bedrohungslage oder eingesetzte IT-Assets. Die Operationalisierung übergreifender Strategien, aber auch bereits das Ableiten von Regelsätzen aus konkreten Sicherheitsrichtlinien, stellt für Unternehmen eine Herausforderung dar. Maschinenlesbare Sicherheitsrichtlinien sind komplex und deren manuelle Erstellung erfordert spezifisches Expertenwissen, das nur in begrenztem Maße zur Verfügung steht und dessen Bereitstellung kostenintensiv ist. Ohne ein wirksames Set an Regelsätzen ist ein SIEM-System in seiner Wirkung stark eingeschränkt und erbringt nicht die Leistungen, die dessen Anschaffungs- und Betriebskosten rechtfertigen würden. Daher ist die Verbreitung solcher SIEM-Systeme immer noch als gering zu bezeichnen. [DRS14]

2.2 IF-MAP-Spezifikation

Die technologische Basis für das SIMU Projekt ist das IF-MAP (Interface for Metadata Access Point) Protokoll der TCG. Dabei handelt es sich um ein offenes, hersteller-unabhängiges Client-Server-Netzprotokoll zum Austausch von beliebigen, in XML codierten Metadaten. IF-MAP ist ein substantieller Bestandteil des Trusted Network Connect (TNC) Frameworks. Die ursprüngliche Motivation für IF-MAP war die Integration von vorhandenen, sicherheitsrelevanten Infrastrukturdiensten wie Firewalls, Virtual Private Networks (VPN) und Intrusion-Detection-Systemen (IDS). Durch die Integration dieser Dienste erhält man eine ganzheitliche Sicht auf den aktuellen Status eines Netzwerkes, was Vorteile bei der Administration und der Erkennung von Bedrohungen verspricht. Die erste Version (1.0) von IF-MAP erschien im Mai 2008. Die Version 2.2 (Revision 10) wurde im März 2014 veröffentlicht[1].

Die TNC-Architektur (siehe Abbildung 1) selbst ist in fünf verschiedene Spalten unterteilt, die jeweils eine Rolle definieren, die Komponenten in einem Netzwerk übernehmen können. Die Bedeutung der Rollen wird im Folgenden kurz beschrieben:

1. Als **Access Requestor (AR)** wird ein Endgerät bezeichnet, das Zugriff auf ein durch TNC geschütztes Netzwerk erhalten möchte.
2. Ein **Policy Decision Point (PDP)** befindet sich innerhalb des zu schützenden Netzwerkes. Basierend auf dem Integritätszustand des Access Requestors entscheidet der PDP, in welchem Umfang Zugriff auf ein geschütztes Netzwerk gewährt wird.
3. Ein **Policy Enforcement Point (PEP)** befindet sich „am Rand“ des zu schützenden Netzwerkes. Dabei handelt es sich in der Regel um Wireless

[1] http://www.trustedcomputinggroup.org/resources/tnc_ifmap_binding_for_soap_specification

Access Points oder kabelgebundene Switche. Ein PEP setzt die Zugriffsentscheidung, die ein PDP getroffen hat, um.

4. Ein Server innerhalb des zu schützenden Netzwerkes, der so genannte **Metadata Access Point (MAP)**, ist dafür verantwortlich, den aktuellen Zustand des Netzwerkes abzubilden. Dieser Zustand wird anhand eines vorgegebenen Formates für Metadaten beschrieben und kann (sicherheitsrelevante) Informationen wie angemeldete Benutzer, verwendete IP-Adressen oder erkannte Anomalien enthalten.
5. Die letzte der definierten Rollen wird als **MAP-Client (MAPC)** bezeichnet. Über die standardisierte Schnittstelle **IF-MAP** können MAP-Clients Metadaten von einem MAP-Server abfragen oder neue Metadaten veröffentlichen.

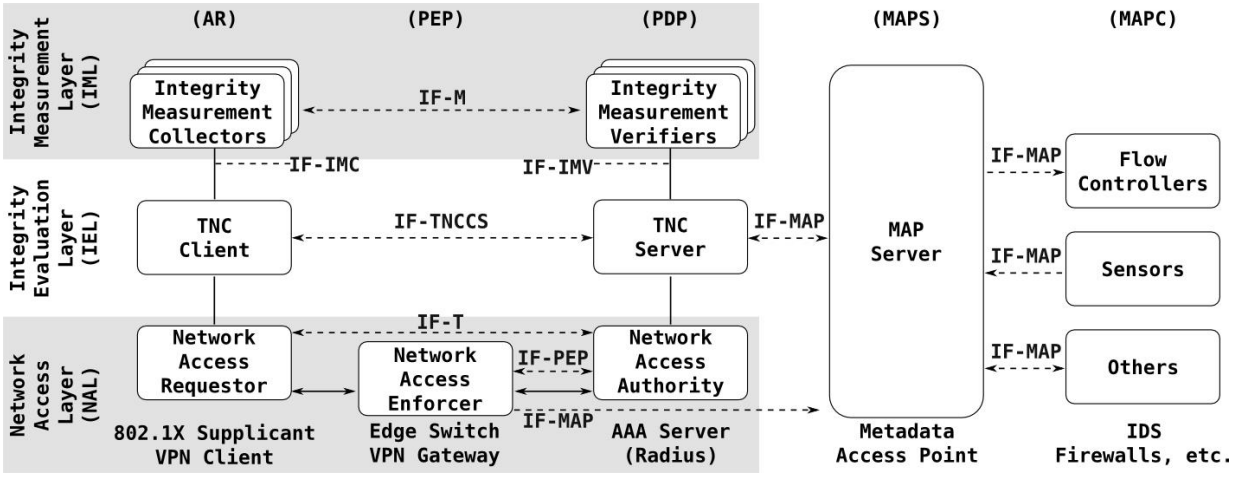

Abbildung 1: TNC-Architektur in der Version 1.4

Über die standardisierte IF-MAP-Schnittstelle können Metadaten von diesem Server abgefragt oder neue Metadaten veröffentlicht werden. Innerhalb des MAP-Servers werden die veröffentlichten Metadaten in Form eines Graphen verwaltet. Damit bietet sich die Möglichkeit, an zentraler Stelle eine Gesamtsicht auf den aktuellen Status eines Netzwerkes zu etablieren. Durch Korrelation der vorhandenen Metadaten können außerdem sicherheitsrelevante Informationen abgeleitet werden.

Systeme, die mit dem MAP-Server kommunizieren, werden als IF-MAP-Clients bezeichnet. Die Kommunikation basiert auf einem Publish-Search-Subscribe-Modell, bei dem sowohl synchron als auch asynchron Metadaten ausgetauscht werden können:

1. Neue Metadaten werden über die **Publish-Operation** veröffentlicht.
2. Nach vorhandenen Metadaten kann per **Search-Operation** gesucht werden.
3. Über die **Subscribe-Operation** können sich IF-MAP-Clients asynchron über Änderungen der im MAP-Server gespeicherten Metadaten informieren lassen. Dabei wird seitens des IF-MAP-Clients spezifiziert, welche Art von Metadatenänderungen relevant ist. Nur solche Änderungen haben eine Benachrichtigung durch den MAP-Server zur Folge.

Technologisch basiert IF-MAP auf einer Reihe von etablierten Standardtechnologien. Als Framework zur Übertragung der Metadaten kommt das SOAP-Protokoll in Kombination mit HTTP(S) zum Einsatz. Das Format der Metadaten ist durch XML-Schemata beschrieben. Auf diese Weise können etablierte Sicherheitssysteme, die um IF-MAP-Client-Funktionen erweitert worden sind, beliebige Metadaten über den aktuellen Status des Netzwerkes austauschen. Auf diese Weise können ansonsten isolierte Daten (IDS Events, DHCP Lease Informationen) verknüpft werden. [DSBW12]

Durch den Einsatz der IF-MAP-Architektur in dem geschilderten Szenario wäre es möglich, kompromittierte Endgeräte gezielt aufzuspüren und ggf. weitere Maßnahmen einzuleiten. Dies kann beispielsweise die Sperrung oder Isolierung dieser Geräte sein. Ein wichtiger Aspekt hierbei ist, dass durch den Einsatz der IF-MAP-Architektur ein kompromittiertes Endgerät auch nach den Authentifizierungs- und Autorisierungsvorgang anhand seines Verhaltens erkannt und somit flexibel auf die jeweilige Bedrohungslage reagiert werden kann. [DDB11]

2.3 SIEM-Technologien

Ein SIEM-System besteht aus diversen Modulen, die die folgenden Funktionen abbilden müssen:

a. **Event Correlation:** Logfiles werden aufgenommen, archiviert, normalisiert und korreliert, um eine Datenbasis zu schaffen, mit der Gefährdung erkannt werden können.

b. **Network Behaviour Anomaly Detection (NBAD):** Anomalien werden auf Netzwerkebene erkannt, Kommunikationsverhalten festgestellt und Abweichungen von der Normalität verfolgt bzw. bei Bedarf diese in die Korrelation der Problemstellung mit aufgenommen.

c. **Identity Mapping:** Sicherheitssysteme, die Anomalien oder Angriffe auf das Unternehmensnetz melden, beschreiben Angreifer bzw. Opfer stets mittels einer IP-Adresse. Das hilft aber nur bedingt weiter, weshalb die Identität der Person aufgelöst wird.

d. **Key Performance Indication:** Die IT-Sicherheit wird messbar gemacht, indem sicherheitsrelevante Informationen und Asset-Details zentral analysiert werden.

e. **Compliance Reporting:** Die IT-Compliance (u.a. Integrität, Risiko, Effektivität) des Unternehmens wird immer wieder hinterfragt und Ergebnisse miteinander verglichen.

f. **Application Programming Interface (API):** Anbindung vorhandener Sicherheitssysteme und Bereitstellung generischer Schnittstellen zur Integration unbekannter Geräte bzw. Systeme.

g. **Role Based Access Control:** Zentrale Sicht auf alle sicherheitsrelevanten Ereignisse innerhalb des Unternehmens, unter Berücksichtigung der Verantwortungsbereiche.

Diese Module machen die Intelligenz eines SIEM aus, die eine Risikoanalyse in Korrelation mit allen bekannten Events möglich macht. Über unterschiedliche Schnittstellen erhält das SIEM Informationen über Assets (z.B. Inventardaten), über Sicherheitslücken (Vulnerability und Patch Management) und kann auf die Dokumentation zugreifen. Zusätzliche Schnittstellen wird es zum Help-Desk oder Asset-Datenbanken geben. Auch das Zusammenspiel zum vorhandenen NAC-System muss über Schnittstellen ermöglicht werden. Abbildung 2 zeigt die wichtigsten Module eines SIEM und die Kommunikationsvariationen.

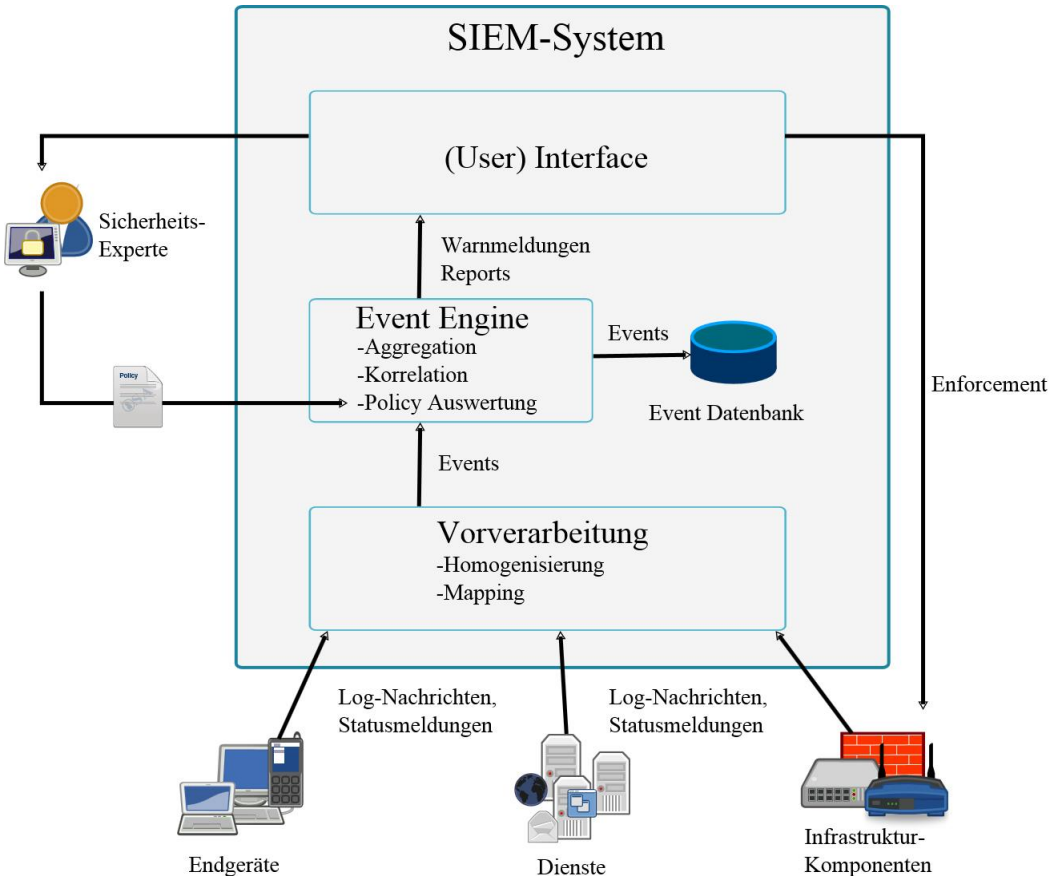

Abbildung 2: SIEM-Systemaufbau und Kommunikationsschnittstellen

Die fachlichen Anforderungen für ein SIEM-System orientieren sich anhand des Haupt-Informationsflusses, d.h. vom Sammeln der Informationen über das Verarbeiten bis zur Reaktion oder Alarmmeldung. Die folgenden Anforderungen müssen daher erfüllt werden:

1. Sammeln von Daten der zentralen Dienste und Netzkomponenten.
2. Aggregation dieser Daten zu aussagekräftigen Ereignissen anhand einer flexibel definierbaren Policy.
3. Darstellungen der sicherheitsrelevanten Ereignisse in verschiedenen Detailstufen.
4. Auslösen von Alarmmeldungen oder ggf. aktiver Eingriff (Alerting/Enforcement).

Um angemessene Entscheidungen zu treffen, ist es wichtig, eine aktuelle und korrekte Wissensbasis über die Geräte, Benutzer und Eigenschaften des Netzwerkes zu besitzen. Zu diesen Informationen gehören:

- Daten über Geräte, die sich im Netz befinden, wie IP-Adresse, MAC-Adresse, DNS-Name, Betriebssystem, Verfügbarkeit des Gerätes (Ping), offene Ports, angebotene Dienste, Datendurchsatz und Systemzustand (installierte Software, Integritätszustand, CPU- und RAM-Auslastung etc.).
- Informationen über Benutzer, die an Systemen angemeldet sind, wie Benutzername, Benutzergruppen, Berechtigungen oder Zuordnung zu Geräten im Netzwerk.
- Detail-Informationen über den Zustand der zentralen Dienste im Netzwerk, beispielsweise Informationen über aktuell angemeldete Benutzer, Fehlermeldung und Status-Reports der Dienste.

Wenn die oben aufgeführten Metadaten zentralisiert und für berechtigte Komponenten zugreifbar sind, können viele Ansätze zur Erkennung und Behandlung von Bedrohungen miteinander kombiniert werden. So können Verletzungen von Unternehmensrichtlinien frühzeitig erkannt und ggf. Maßnahmen eingeleitet werden. Auch kann die SIMU-Engine (siehe SIMU-Architektur) durch das stetige Sammeln dieser Informationen den „Normalzustand“ erlernen und außergewöhnliche und nicht vorhergesehene Bedrohungen erkennen. [DRS14]

2.4 SIEM-Architektur von SIMU

Abbildung 3 zeigt die SIEM-Architektur des SIMU-Projektes, die das IF-MAP-Protokoll als zentrales Protokoll zum Austausch von Metadaten in den Mittelpunkt stellt. Neben den Open-Source-Tools, die schwerpunktmäßig integriert werden, sind die deutschen Herstellerlösungen *NCP-VPN* und *macmon NAC* ebenfalls Teil der Architektur.

Die SIMU-Architektur teilt sich in zwei Schichten, die durch das IF-MAP-Protokoll miteinander verbunden werden:

a. Die **SIMU-Kollektoren- und Flow-Controller-Schicht**, welche für die Datensammlung und das Enforcement verantwortlich ist.
b. Die **SIMU-Engine**, welche die den zentralen Wissens- und Datenspeicher, Komponenten zur Korrelation, Aggregation und Darstellung von Daten sowie Protokollschnittstellen enthält.

Die SIMU-Engine muss u.a. die Ergebnisse der Datenanalyse durch die SIEM-GUI entsprechend aufbereitet darstellen. Die grafische Oberfläche muss dafür mit der Detection Engine (intelligente Analyse der MAP-Server-Daten) und VisITMeta (grafische Darstellung des IF-MAP-Graphens) direkt kommunizieren sowie indirekt mit dem IO-Tool (Erhebung der Daten der IT-Infrastruktur). Sie muss Events eindeutig anzeigen und ensprechende Mitteilungen an die IT-Administration schicken. Dieser SIEM-GUI kommt damit eine zentrale Bedeutung zu.

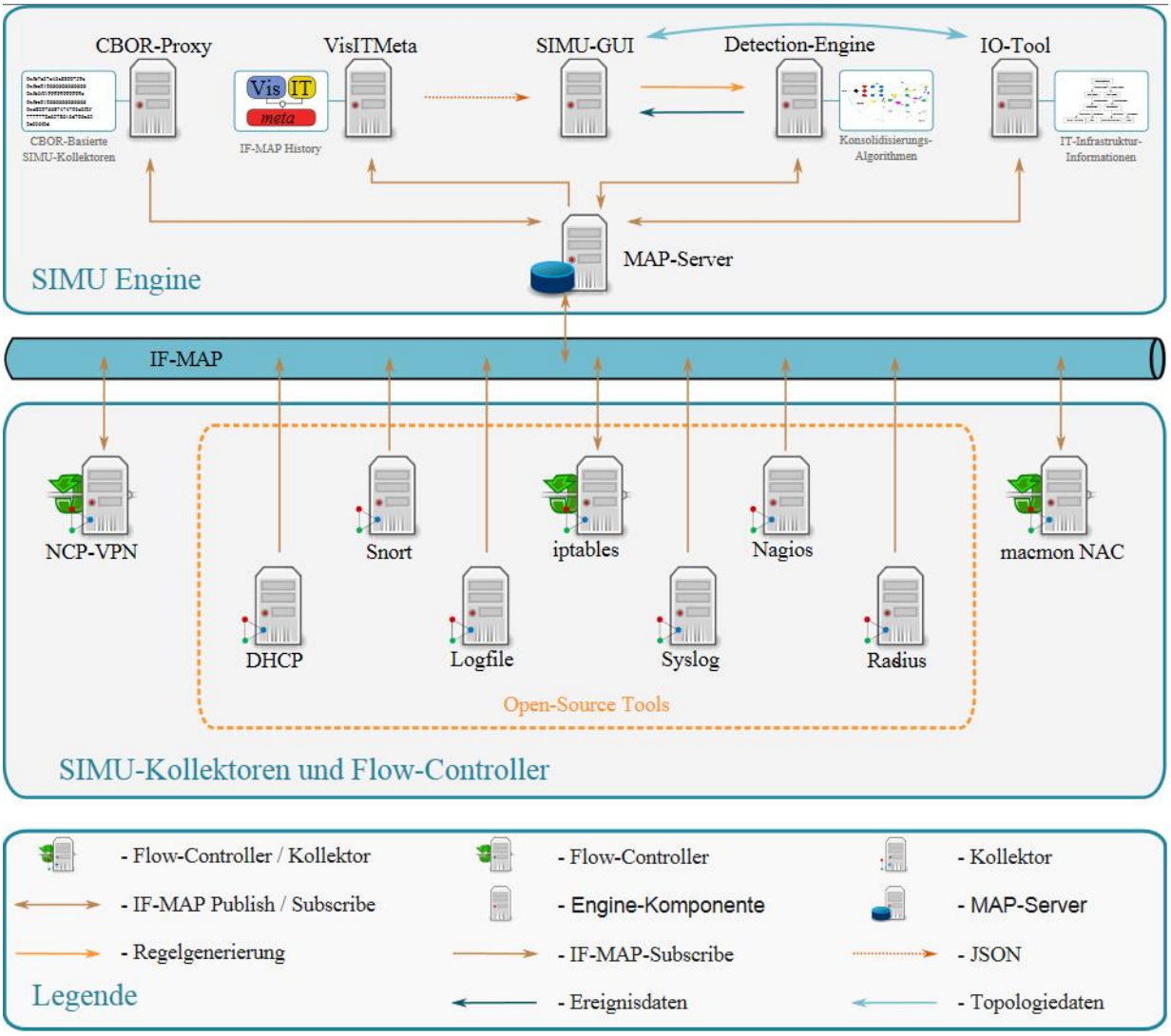

Abbildung 3: SIEM-Architektur im SIMU-Projekt

Flow-Controller und Kollektoren stellen die Schnittstelle zwischen der SIMU-Engine und den Diensten, Sicherheits- und Infrastrukturkomponenten des Netzwerks dar. Folgenden Kollektoren wurden für die Integration in SIMU analysiert und umgesetzt:

- **DHCP-Kollektor:** Extrahiert Metadaten zu aktuellen IP-Leases des DHCP-Servers aus dem Lease-File von DHCP-Servern.
- **Radius-Kollektor:** Liefert Metadaten zu Benutzeranmeldungen sowie Metadaten zu den Benutzern selbst (Gruppen, Berechtigungen).
- **Syslog-Kollektor:** Stellt voraggregierte Metadaten zum Zustand von Hosts und Diensten (CPU-Last, Fehlgeschlagene Log-Ins) zur Verfügung.
- **Nagios-Kollektor:** Veröffentlicht aus Nagios extrahierte Metadaten zum Zustand von Hosts und Diensten (u.a. Netzwerkverfügbarkeit).
- **Snort-Kollektor:** Übersetzt Snort-Alarme in IF-MAP-Metadaten.

- **Logfile-Kollektor:** Generischer Kollektor zur Auswertung beliebiger Log-Dateien und Transformation in IF-MAP Metadaten.

Zusätzlich sind die IF-MAP-Kollektoren Android, Icinga REST, LDAP und WMI (Windows Management Instrumentation) umgesetzt worden. Die Einbeziehung von Android wurde notwendig, da es auch möglich sein sollte, mobile Endgeräte mit in die SIEM-Betrachtung mit einbeziehen zu können. So kann über diesen Kollektor u.a. Firmware, Kernel-Version, Build-Nummer und SMS-/E-Mail-Informationen weitergeleitet werden. Der Nagios-Fork Icinga kann als Alternative zu Nagios über die REST-Schnittstelle verwendet werden und Anfragen bei Bedarf durchführen. Der LDAP-Kollektor soll eine Verbindung zum vorhandenen Verzeichnisdienst herstellen, während der WMI-Client auch die Windows-Clients mit einbezieht.

Des Weiteren ist die Integration folgender Flow-Controller-Komponenten umgesetzt worden, die zum Teil auch Kollektoren-Funktionalitäten besitzen:

- **Iptables-Flow-Controller:** Ermöglicht das automatische Anlegen von Firewall-Regeln für Hosts als Reaktion auf bestimmte Metadaten und veröffentlicht diese als Enforcement-Reports im MAP-Server.
- **macmon NAC:** Das NAC-System von macmon secure publiziert verschiedene Daten zu aktiven Endgeräten im Netzwerk. Hierzu gehören vor allem Autorisierungsinformationen zu bekannten Endgeräten, deren Standort im Netzwerk (physikalischer Port, WLAN-AP) und weitere Geräte-Charakteristika wie Betriebssysteminformationen und offene Ports.
- **NCP-VPN:** Die von NCP entwickelte VPN-Lösung kann eine Reihe relevanter Metadaten auf dem MAP-Server zur Verfügung stellen. Dies sind insbesondere Informationen über angemeldeter Benutzer sowie die IP-Adresse der Geräte, mit denen die Benutzer im VPN angemeldet sind, Datendurchsatz und Verbindungszeit der jeweiligen Benutzer. Es wird ein Enforcement auf VPN-Ebene ermöglicht.

Zusätzlich ist der Flow-Controller/Kollektor OpenVPN mit SSL verwendet worden, um zusätzlich eine Alternative zur Herstellerlösung von NCP anbieten zu können, die ausschließlich auf IPsec basiert.

Der MAP-Server stellt den zentralen Austauschpunkt von Informationen dar. Alle Sensoren und Flow-Controller veröffentlichen gesammelte Informationen via IF-MAP im MAP-Server. Die zweite Kernkomponente neben dem MAP-Server ist die Detection Engine, die später als *Graph Pattern Matching (GPM) Engine* beschrieben wird. Diese verwendet die Informationen aus dem MAP-Server, um zum einen Abweichungen von definierten Zuständen festzustellen (Pattern Matching) und zum anderen, um durch den Einsatz von Anomalie-Erkennungsalgorithmen Abweichungen vom Normalverhalten festzustellen. Dafür kann die Detection-Engine auf den gesamten Datenbestand des MAP-Servers zugreifen.

Durch die Anbindung des IO-Tools [BIRK12] wird der Datenbestand mit weiteren Asset-Informationen über die Netzwerkinfrastruktur angereichert, was eine Korrelation mit zusätzlichen Informationen ermöglicht. Das Normalverhalten des Systems wird

durch eine Trainingsphase mit zuvor bereitgestellten Trainingsdaten berechnet. Für den wartungsarmen Betrieb ist es dabei zwingend notwendig, dass die verwendeten Anomalie-Erkennungsverfahren möglichst zuverlässig und selbstständig arbeiten. Dies gilt auch für die Trainingsphase solcher Erkennungsverfahren. Deshalb sind speziell sogenannte „nicht-überwachte“ (unsupervised) Verfahren (vgl. [CHAN09]) für die Anwendung in SIMU geeignet.

Beim Einsatz von „überwachten“ (supervised) oder „teilüberwachten“ (semi-supervised) Verfahren besteht der Nachteil, dass in der Trainingsphase jedes Datum vom Benutzer als Normalfall oder Anomalie gekennzeichnet werden muss. Dieses hat offensichtlich mehrere Nachteile: zum einen ist es extrem aufwändig eine große Anzahl von Trainingsdaten auf diese Weise manuell korrekt zu kennzeichnen, zum anderen ist das dafür nötige Expertenwissen beim jeweiligen Anwender nicht vorhanden. Darüber hinaus ist es sehr schwierig, geeignete Trainingsdaten auszuwählen, die eine möglichst große Anzahl an Normalfällen und Anomalien abdecken.

Die VisITMeta-Komponente dient zur Langzeitarchivierung des IF-MAP-Graphen. Somit ist es möglich, jeden beliebigen Zustand des Graphen zu einem späteren Zeitpunkt – z.B. zu Analysezwecken – zu rekonstruieren. Außerdem bietet VisITMeta sowohl Schnittstellen als auch Visualisierungskomponenten, die in der SIEM-GUI verwendet werden können, um alle Aspekte des SIMU-Systems Zielgruppengerecht aufzubereiten und anzuzeigen. [DRS14]

2.5 Visualisierung der Informationen

Die Aufbereitung und Darstellung von sicherheitsrelevanten Informationen werden im Projekt durch zwei verschiedene Komponenten vorgenommen. Während die erste Komponente auf die Visualisierung der Metadaten mit deren Kommunikationsbeziehungen spezialisiert ist, gibt die SIEM-GUI eine Übersicht über die Vorfälle wieder, inkl. des auftretenden Sicherheitsrisikos. [DHRR15]

2.5.1 VisITMeta

VisITMeta (siehe auch [AHH14]) ist eine Software, die im gleichnamigen BMBF-geförderten Drittmittelprojekt (April 2012 bis März 2015) entwickelt wurde und die Speicherung von IF-MAP-Graphen und deren Visualisierung zur Verfügung stellt. Mit den Funktionen von VisITMeta können im SIMU-Projekt sowohl die IF-MAP-Rohdaten (veröffentlicht durch die verschiedenen Kollektoren) als auch die Ergebnisse von Korrelationen und anderen Auswertungen angezeigt werden.

Die Software selber teilt sich in zwei Komponenten auf:

a. Datenservice
b. Visualisierungskomponente

Der Datenservice ist ein Dienst, welcher IF-MAP-Daten wie ein gewöhnlicher IF-MAP-Client über eine oder mehrere parallele Abonnements (Subscriptions) erfasst und in einer Graph-Datenbank speichert. Die gespeicherten Daten können dann über eine REST-artige Schnittstelle abgerufen werden. Zu den möglichen Abfragen gehören der

aktuelle Graph-Zustand, Zustände zu beliebigen Zeitpunkten und Änderungen zwischen zwei beliebigen Zeitpunkten, wie die Abbildung 4 zeigt.

Hierzu wird auch eine Liste der Änderungen verwaltet und bereitgestellt, die Informationen enthält, wann wie viele und welche Änderungen an den Daten vorgenommen wurden. Die gespeicherten Daten zusammen mit der definierten Schnittstelle zum Abfragen der Daten können dann sowohl zur Visualisierung verwendet werden oder aber um darauf Auswertungen durchzuführen, welche die Historie der Daten miteinbeziehen können.

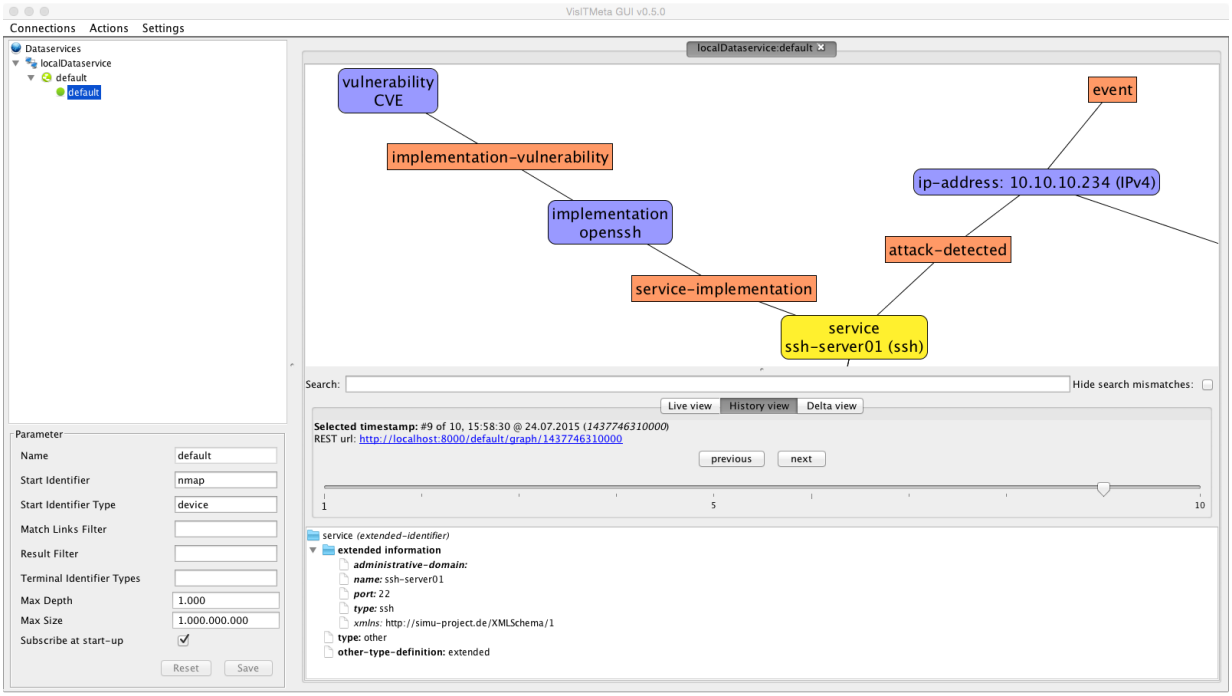

Abbildung 4: Benutzeroberfläche von VisITMeta

Die Visualisierungskomponente (Benutzeroberfläche, siehe Abbildung 4) holt sich über die REST-Schnittstelle des Datenservice die benötigten Daten, um den Zustand des Graphen darstellen zu können. Die Visualisierungskomponente hat also keine direkte Verbindung zu einem MAP Server, sondern immer nur zu einer (oder mehreren) Datenservice-Instanzen. Diese Verbindungen können über die GUI konfiguriert werden.

Zur Darstellung der MAP-Daten (siehe Abb. 3) werden Identifier und Metadaten als Knoten sowie Links als Verbindung zwischen diesen Knoten abgebildet. Metadaten auf einem Link erscheinen in der Darstellung als Knoten mit Verbindungen zu den Identifiern an beiden Enden eines Links. Die Visualisierung unterstützt verschiedene Graph-Layouting-Algorithmen (Force-Directed, Spring, Bipartit), die im laufenden Betrieb gewechselt werden können.

Die Navigation durch die Historie der Graph-Daten erfolgt durch die Aufteilung in drei Betriebsmodi: im Live-Betrieb wird immer der aktuelle Stand des Metadatengraphen beobachtet und bei Neuerungen entsprechend angepasst. Ein weiterer Modus ist die

Betrachtung des Graphen zu einem gegebenen Zeitpunkt. Über den Delta-Betrieb können die Änderungen zwischen zwei gewählten Zeitpunkten betrachtet werden. Einstellungsmöglichkeiten für die Farbgebung der Elemente und ein Highlighting bei Veränderungen werden eingesetzt, um eine übersichtliche Darstellung und Verfolgung der Ereignisse zu ermöglichen. Eine einfache Suche ermöglicht das Finden und Hervorheben von Elementen aufgrund ihrer Inhalte.

Im unteren Bereich des Anwendungsfensters werden detaillierte Informationen über einzelne Graph-Elemente angezeigt. Hierzu gehören alle durch IF-MAP vorgegebenen Attribute (Zeitpunkt der Veröffentlichung, ID des veröffentlichenden Clients, XML-Schema-Definitionen etc.) und auch die Werte und innere Struktur des konkreten Identifiers oder Metadatums. [DHRR15]

2.5.2 SIEM-GUI

Während VisITMeta die Möglichkeit bietet, die IF-MAP Rohdaten des Systems einzusehen, ist eine weitere Visualisierungskomponente namens **SIEM-GUI** notwendig, über die das System von IT-Administratoren verwendet werden kann. Diese wurde als Java-Webanwendung umgesetzt und kann mit allen gängigen Browsern verwendet werden. Das Web-Interface ist mit Hilfe von JavaScript und AngularJS als Single-Page-Application implementiert worden. Dieser Ansatz reduziert die Datenmenge, die zwischen Server und Client ausgetauscht werden muss, da das Grundgerüst der Seite nur einmalig geladen werden muss.

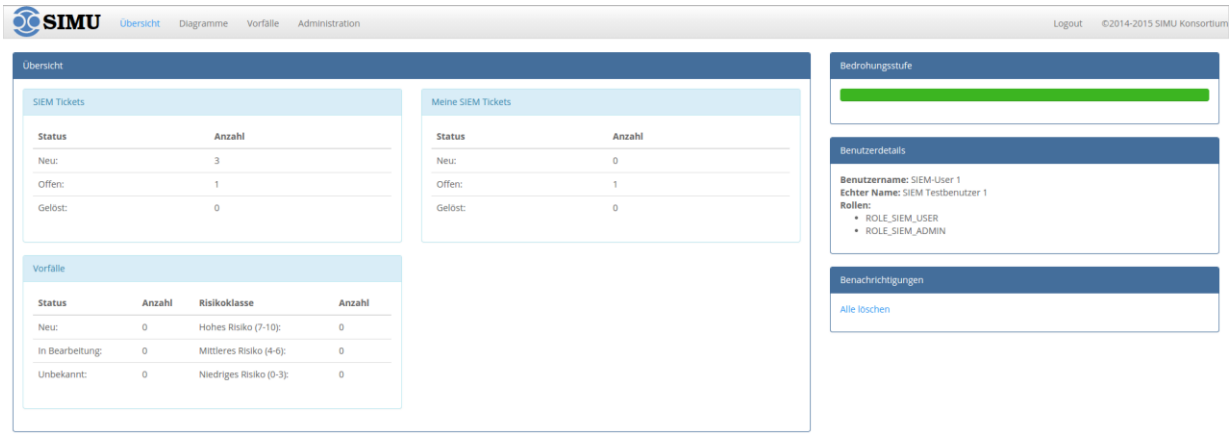

Abbildung 5: Oberfläche der SIEM-GUI

Die SIEM-GUI (siehe Abbildung 5) ermöglicht die Einsicht und Bearbeitung von erkannten Vorfällen. Weiterhin bietet sie die Möglichkeit, den Teil des IF-MAP Metadatengraphen einzusehen, der zur Meldung eines Vorfalls geführt hat. Dadurch lässt sich im Detail nachvollziehen, was genau von der Detection Engine erkannt wurde. Ein Vorfall kann außerdem einem Bearbeiter zugewiesen werden, der unter anderem in der Lage ist, den Status des Vorfalls zu ändern. Es ist jedoch auch anderen Benutzern möglich, die Bearbeitung eines Vorfalls zu kommentieren, wodurch eine gemeinsame Problemlösung vereinfacht wird.

Um die genannten Funktionen umzusetzen, integriert die SIEM-GUI verschiedene externe Dienste in das System. Diese werden über Konnektoren lose an das GUI-System gekoppelt und sind dadurch austauschbar. Dadurch lässt sich die SIEM-GUI bei Bedarf in bereits bestehende Systeme integrieren und benötigt somit nur wenige Dienste, die zusätzlich im Firmennetzwerk aufgebaut werden müssen. Als externe Dienste werden genutzt:

a. Ticketsystem,
b. Benutzerverwaltung,
c. Sicherheitsvorfälle,
d. VisITMeta Dataservice.

Die **Vorfälle** werden aus einer Datenbank gelesen, in der sie von der Detection Engine abgelegt werden. Dieser Weg ist vom SIMU-System vorgegeben, das verwendete DBMS kann jedoch variiert werden. Ebenfalls vorgegeben ist die Verbindung zum **VisITMeta Dataservice**, die benötigt wird, um die IF-MAP-Rohdaten zu einem Vorfall aus dem Metadatengraphen auszulesen. Der Dataservice kann dabei auf der gleichen Maschine wie die SIEM-GUI oder aber auch auf anderen Rechnern im lokalen Netzwerk installiert sein. Über ein VPN ließe sich auch auf einen Dataservice an einem anderen Standort zugreifen.

Das **Ticketsystem** und die **Benutzerverwaltung** sind generische Dienste, die von verschiedenen Systemen erfüllt werden können. Dieses Vorgehen hat den Vorteil, dass die SIEM-GUI bei Bedarf mit überschaubarem Aufwand in ein bestehendes Netzwerk integriert werden kann und vorhandene Dienste verwendet werden können. So ist es zum Beispiel nicht notwendig ein zweites Ticketsystem in Betrieb zu nehmen, wenn in der Firma bereits eines in Verwendung ist. Im einfachsten Fall werden beide Aufgaben vom gleichen System übernommen, so dass keine zusätzliche Verknüpfung zwischen SIEM-GUI- und Ticketsystem-Benutzern hergestellt werden muss. Dadurch ist im Idealfall kein Zugriff auf die Benutzeroberflächen der externen Dienste notwendig und alles kann direkt über die SIEM-GUI konfiguriert werden.

Die Tickets des Ticketsystems dienen der Nachverfolgung der Bearbeitung eines Vorfalls. So wird über ein Ticket der Status der Bearbeitung gespeichert, es werden Kommentare und Zeitbuchungen abgelegt und Deadlines für die Behebung definiert. Aus Benutzersicht wird allerdings ausschließlich mit Vorfällen interagiert, welche Informationen tatsächlich in der Vorfalldatenbank und welche im Ticketsystem hinterlegt werden ist für den Anwender nicht wichtig und daher auch nicht ersichtlich. Dies vereinfacht die Bedienung der SIEM-GUI zusätzlich, da prinzipiell nur mit einer einzigen Art von Information, den Vorfällen, umgegangen werden muss. Alles Weitere, zum Beispiel die Rohdaten und die Nachverfolgung über das Ticket, wird in die entsprechende Darstellung des Vorfalls integriert.

Zusätzlich ist die Verwendung von **WebSockets** für die Kommunikation zwischen dem Web-Interface und dem Backend der SIEM-GUI vorgesehen. Neben der üblichen Request-Response-Kommunikation, die auch über eine REST-Schnittstelle möglich wäre, lassen sich durch diese Technik Push-Benachrichtigungen vom Server an den

Client senden. Dies wird dazu verwendet, um den Anwender nahezu in Echtzeit über neue Vorfälle und andere wichtige Vorgänge im überwachten System über die SIEM-GUI zu informieren.

Die **Datenbank**, in der die erkannten Vorfälle von der Detection Engine abgelegt werden, wird von der SIEM-GUI periodisch auf neue Einträge durchsucht. Wird ein neuer Vorfall gefunden, so wird zunächst im Ticketsystem gesucht, ob es bereits ein Ticket zu diesem Vorfall gibt (z.B. beim Neustart des Systems). Wird keines gefunden, so erstellt die SIEM-GUI über den entsprechenden Konnektor ein neues Ticket und verknüpft dieses mit dem Vorfall. Die Detection Engine kann einem Vorfall eine Handlungsempfehlung mitgeben, um den Anwender bei der Behebung des Problems zu unterstützen. Ist diese in der Datenbank hinterlegt, so wird sie bei der Erstellung des Tickets als Text eingetragen und kann so bei der Ansicht des Vorfalls über das Web-Interface direkt eingesehen werden.

Neben der Bearbeitung von Vorfällen bietet die SIEM-GUI auch Funktionen, um einen schnellen Überblick über den Zustand des Netzwerks zu bekommen. Es gibt zwei Unterseiten, die eine Zusammenfassung des Systemzustands liefern. Die eine bietet eine tabellarische Übersicht über den Status unbearbeiteter Vorfälle und deren **Risikoeinschätzung**. Die zweite Seite stellt ähnliche Informationen in Form von Graphen und Diagrammen dar, was zum Beispiel das Erkennen von zeitlichen Entwicklungen gegenüber der tabellarischen Ansicht vereinfacht. Ein weiteres Feature ist eine graphische Darstellung der aktuellen **Bedrohungsstufe** (Threat Level) im Netzwerk. Diese wird aus der Risikobewertung der aktiven, also noch nicht behobenen, Vorfälle errechnet und auf jeder Unterseite der SIEM-GUI angezeigt. [DHRR15]

2.5.3 Darstellung von Regelverstößen

Verstöße gegen die Regeln der Detection Engine können auf zwei verschiedene Arten innerhalb der SIEM-GUI bzw. durch VisITMeta dargestellt werden:

a. **Darstellung der zu einer Regelauslösung gehörenden Teilgraphen:** Hierbei werden nur die Teile des gesamten MAP-Graphen dargestellt, die durch ein Graph-Muster der Detection Engine für das Auslösen einer konkreten Regel erkannt wurden, sozusagen die Instanz eines Graph-Musters. Die Detection Engine kann hierzu der SIEM-GUI diese Informationen mitsenden. Die VisITMeta-Visualisierungskomponente muss hierzu mitgeteilt bekommen, dass sie nur den entsprechenden Teilgraphen abrufen und darstellen soll, z.B. durch Übermittlung eines passenden Filterausdrucks.
b. **Regel-unabhängige Betrachtung:** Alternativ dazu kann auch der gesamte MAP-Graph dargestellt werden. Regelauswertungen werden hier gleichgestellt mit allen anderen MAP-Daten visualisiert. Mit den üblichen Mitteln zur Navigation, Suche und Filterung kann der Benutzer den Graphen betrachten. Diese Darstellung wird durch die gewöhnlichen Ansichten der VisITMeta-Visualisierungskomponente abgebildet.
c. **Kombination beider Ansätze:** Werden beide Ansätze kombiniert, sieht der Benutzer grundsätzlich alle Daten des MAP-Graphen wie in Variante 2,

allerdings werden die Graph-Muster-Instanzen hier speziell hervorgehoben. So kann direkt der Kontext der erkannten Graph-Muster-Instanzen in Bezug auf den Rest des Graphen bzw. die direkte Nachbarschaft betrachtet werden.

Unabhängig davon, ob nur Teilgraphen oder der gesamte Graph mit den Regelverstößen betrachtet wird, können für die eigentliche visuelle Darstellung der Regelauswertungen verschiedene grafische Konzepte verwendet werden. Im einfachsten Falle werden die von der Detection Engine publizierten Metadaten wie alle anderen Metadaten auch dargestellt (gleiche Form, Umrandung, etc.). Die GUI kann die Metadaten und Identifier, welche zu einer Regelauswertung gehören alternativ mit einer eigenen, speziell dafür konfigurierten Farbe und Form anzeigen, um sie so hervorzuheben und damit für den Benutzer leichter von anderen Elementen unterscheidbar zu machen. [DHRR15]

3 Schlüsselkonzepte des SIMU-Projekts

Um das Ziel des SIMU-Projekts, ein auf IF-MAP aufbauendes SIEM-System, zu erreichen, mussten einige Herausforderungen gelöst werden. Zunächst ist das IF-MAP-Metadatenschema für TNC nicht ausreichend, um alle benötigten Informationen darzustellen. Deshalb wurde das Schema über die von IF-MAP-spezifizierten Schnittstellen ergänzt. Das genaue Vorgehen wird im folgenden Abschnitt 3.1 erläutert.

Im Abschnitt 3.2 wird das Problem des hohen Netzwerkverkehrs durch IF-MAP angegangen, denn das klassische SOAP-basierte Format benötigt vergleichsweise viel Bandbreite im Netzwerk. Das dritte Schlüsselkonzept in Abschnitt 3.3 behandelt die einfache Integration des Systems in bestehende Infrastrukturen. Dies verringert die Notwendigkeit, bereits vorhandene Dienste erneut ausrollen zu müssen.

3.1 Erweiterung des IF-MAP-Metadatenschemas

Da IF-MAP ein Teil der Trusted Network Connect (TNC) Spezifikation ist, ist das grundlegende Metadatenschema auf die Feststellung ausgelegt, ob ein Gerät Zugriff auf das Netzwerk bekommen darf oder nicht. Diese Informationen sind zwar auch in einem SIEM-System wichtig, allerdings nicht ausreichend. Daher wurden im Rahmen des SIMU-Projekts Erweiterungen entwickelt, mit denen zusätzliche Informationen im MAP-Server gespeichert werden können.

Die folgenden Abschnitte beschreiben kurz die gemachten Erweiterungen und stellen die Struktur der entstehenden Teilgraphen dar. Für den genauen Aufbau der einzelnen Elemente kann die XSD-Datei *simu.xsd* herangezogen werden. Diese kann auf der Website des SIMU-Projekts heruntergeladen werden.

3.1.1 Service-Implementation-Vulnerability

Eine der Kernaufgaben für SIEM-Systeme ist die Erkennung von Angriffen auf Dienste im Netzwerk. Dabei kann es sich um vergleichsweise simple Angriffe wie DDoS oder Brute Force handeln, jedoch auch um gezielte Angriffe auf bekannte Schwachstellen bestimmter Dienste.

Ein IDS, wie zum Beispiel Snort, ist in der Lage solche Aktivitäten zu erkennen und zu melden. Um jedoch die richtigen Schlüsse daraus zu ziehen, ist die Korrelation mit zusätzlichen Informationen notwendig. Insbesondere wichtig ist dabei zu wissen, welche Dienste im Netzwerk überhaupt vorhanden sind und welche Schwachstellen diese aufweisen.

Um diese Zusammenhänge im MAP-Server abbilden zu können, wurde der Service-Implementation-Vulnerability Teilgraph entworfen. Dieser besteht aus den Extended Identifiern *simu:service*, *simu:implementation* und *simu:vulnerability*. Diese beschreiben entsprechend einen Typ von Service, eine spezifische Implementation bzw. Version und eine Schwachstelle, die für diese Implementation bekannt ist. In Abbildung 6 ist der Aufbau dieses Teilgraphs inklusive der Verbindungspunkte zum Standardschema zu sehen. Die *simu:service* und *simu:implementation* Identifier können zum Beispiel von Portscannern wie nmap oder dem Macmon NAC

veröffentlicht werden, die *simu:vulnerability* Identifier von einem Schwachstellenscanner wie OpenVAS. Ein *simu:implementation* Identifier kann dabei mit mehreren *simu:vulnerability* Identifiern verbunden sein.

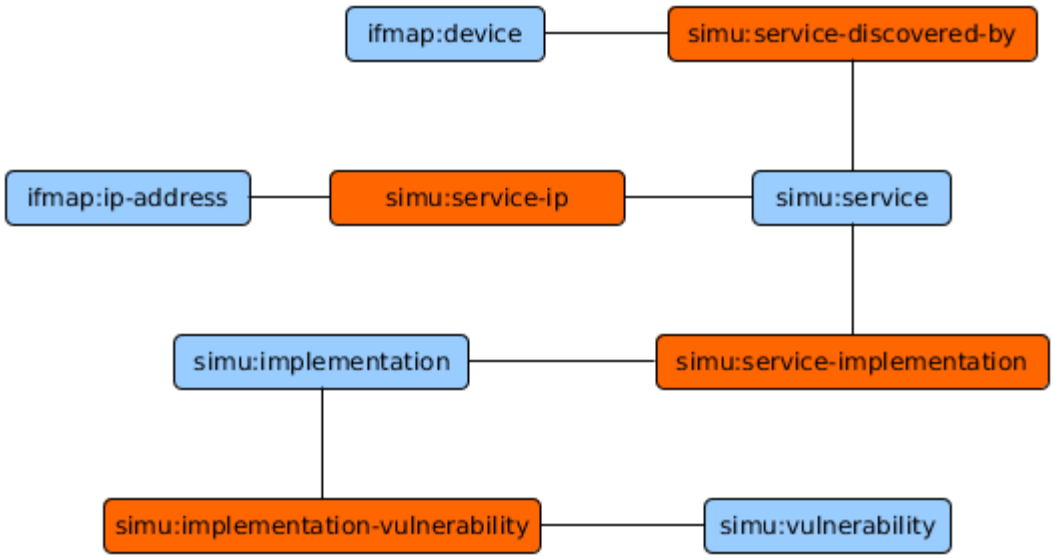

Abbildung 6: Service-Implementation-Vulnerability Teilgraph

3.1.2 Attack-Detected

Die Integration eines IDS wie Snort in eine IF-MAP-Infrastruktur stellt eine besondere Hürde dar. Die Datenmengen, die von einem Snort-Sensor produziert werden, bringen den MAP-Server schnell an seine Belastungsgrenze. Dies kann dazu führen, dass ein massiver Angriff auf das Netzwerk zu einem Ausfall des SIEM-Systems durch Überlastung führt.

Um dieses Problem zu umgehen, wurde im SIMU-Projekt entschieden nur erkannte CVE-Angriffe auf Schwachstellen an den MAP-Server weiterzuleiten. Dadurch gehen viele Informationen verloren, allerdings ist so eine Überlastung des MAP-Servers sehr unwahrscheinlich. Eine Lösung für die weiteren Informationen, die vom Snort erzeugt werden, konnte innerhalb des Projekts nicht gefunden werden.

Das Standardschema von IF-MAP bietet mit dem Metadatum *meta:event* eine Möglichkeit, solche Angriffe zu melden. Allerdings ist hier das Problem, dass Metadaten vom Typ *meta:event* flüchtig sind, da sie nur per *update-notify* veröffentlicht werden dürfen und somit nicht dauerhaft im MAP-Server gespeichert werden können. Zudem ist dieses Metadatum kein Link. Es wird an dem Identifier veröffentlicht, von dem der Angriff ausging. Was der Angriff zum Ziel hatte, ist nicht oder nur über Umwege in Freitext-Feldern erkennbar.

Da dies für eine einfache Korrelation nicht ausreichend ist, wurde das Metadatum *simu:attack-detected* entwickelt, dass genau diese Unzulänglichkeiten ausgleicht. Es wird per *publish-update* veröffentlicht, ist somit also persistent, und bietet als Link direkte Informationen darüber, von wem der Angriff ausging und welchen Dienst er

zum Ziel hatte. Abbildung 7 zeigt, wie das Metadatum im MAP-Graph eingefügt ist. Der Identifier *ifmap:ip-address* ist dabei der Ausgangspunkt des Angriffs, der *simu:service* Identifier ist das Ziel.

Abbildung 7: Attack-Detected Metadatum

3.1.3 Erweiterung des Access-Request Teilgraphen

Die Spezifikation des *ifmap:access-request* Identifiers und der damit verbundenen Metadaten im IF-MAP-Standard ist bereits sehr umfangreich und bietet viele Informationen über die Identität, mit der eine Anmeldung erfolgt. Allerdings ist in der Spezifikation nicht vorgesehen, dass fehlgeschlagene Anmeldungen im MAP-Graphen verbleiben. Diese werden direkt wieder entfernt oder nie veröffentlicht. Um aber zum Beispiel Brute-Force-Angriffe zuverlässig erkennen zu können, sind diese Informationen von unschätzbarem Wert.

Daher wurde im SIMU-Projekt der Access-Request-Teilgraph um Metadaten erweitert, die nicht unter Einschränkung der Spezifikation fallen und somit auch nach einem fehlgeschlagenen Login-Versuch erhalten bleiben. Weiterhin wird direkt zwischen erfolgreichen und fehlgeschlagenen Versuchen unterschieden, was die Konstruktion einer entsprechenden Korrelationsregel stark vereinfacht.

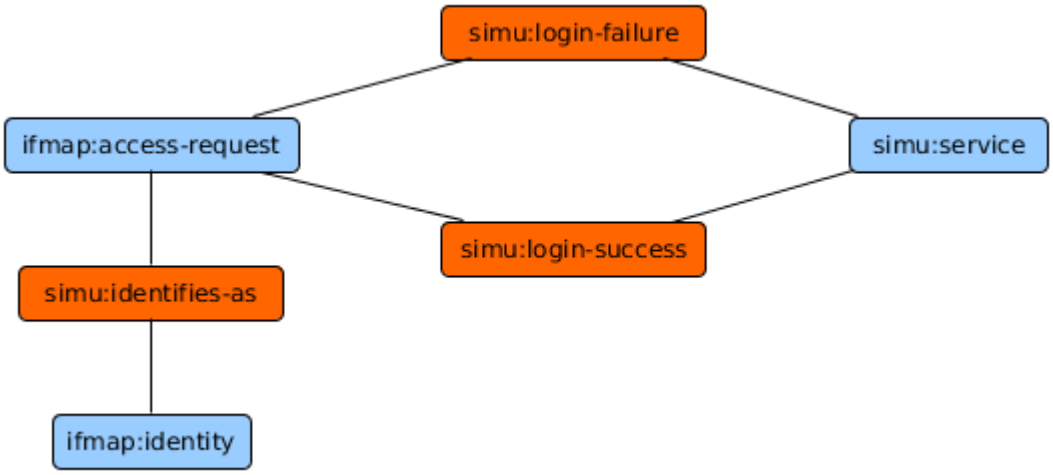

Abbildung 8: Access-Request Erweiterung

In Abbildung 8 ist die Struktur der neuen Metadaten dargestellt. Die beiden Metadaten *simu:login-failure* und *simu:login-success* stellen die Unterscheidung zwischen fehlgeschlagenen und erfolgreichen Versuchen dar. Dabei kann natürlich mit jedem Access-Request nur jeweils eines der beiden Metadaten verbunden sein. Da diese Metadaten als Link konzipiert sind, erlauben sie einen direkten Rückschluss darauf, an welchem Service der Anmeldeversuch stattgefunden hat.

Das Metadatum *simu:identifies-as* verfolgt im Grunde den gleichen Zweck wie das Metadatum *meta:authenticated-as* aus dem IF-MAP-Standard, wird jedoch im Gegensatz zu letzterem auch dann veröffentlicht, wenn der Login-Versuch fehlgeschlagen ist. Somit ist auch dann erkennbar, welche Identität für die Anmeldung verwendet wurde. [DSH15]

3.1.4 File-Integrity-Monitor

Die Überwachung von Dateien auf Änderungen ist ein vollständig neuer Einsatzbereich von IF-MAP, weshalb auch hier ein neuer Teilgraph entwickelt werden musste. Dieser knüpft am *ifmap:device* Identifier an und legt dort für jede überwachte Datei einen Teilgraph an, der die überwachte Datei und ihren Zustand abbildet.

In Abbildung 9 ist dieser Teilgraph sichtbar. Der *simu:file* Extended Identifier stellt eine einzelne Datei dar, die vom File Integrity Monitor überwacht wird. Dabei ist zu beachten, dass dieser Identifier über eine Administrative Domain klar seinem Device zugeordnet werden können muss. Wird dies nicht beachtet, dann ist der resultierende MAP-Graph nicht aussagekräftig und die Dateien lassen sich nicht immer explizit einem Device zuordnen.

Das Metadatum *simu:file-monitored* zeigt, dass diese Datei auf diesem Device gerade überwacht wird und über *simu:file-status* wird der aktuelle Zustand der Datei abgebildet.

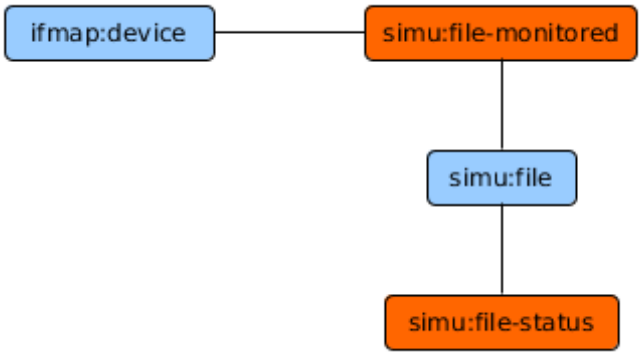

Abbildung 9: File-Integrity-Monitor Teilgraph

3.1.5 NCP VPN-Erweiterungen

Um die zusätzlichen Informationen, die der NCP VPN-Client auslesen kann, darzustellen, wurden auch hier einige Ergänzungen gemacht. Dazu wurden neue Metadaten entwickelt, die Informationen über das verbundene Client-System beinhalten. Diese Metadaten sind in Abbildung 10 abgebildet. Mit *simu:os-infos* werden detaillierte Informationen über das Betriebssystem gespeichert. Das Metadatum *simu:file-hashes* enthält Hashes bestimmter, zuvor festgelegter Dateien. In *simu:wsc-healths* sind die Zustände des Windows Safety Center abgelegt und über das Metadatum *simu:app-infos* werden Informationen über installierte Anwendungen bereitgestellt.

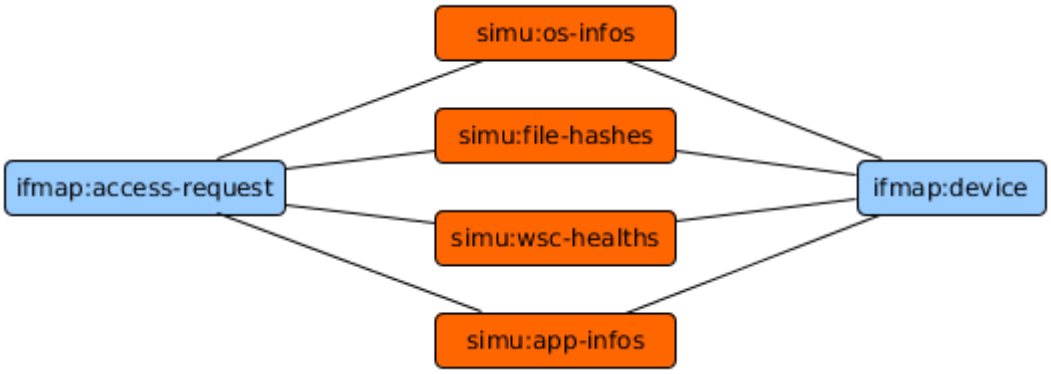

Abbildung 10: Metadaten des NCP VPN Clients

3.2 Verringerung der benötigten Bandbreite durch CBOR

Ein großes Problem beim Einsatz von IF-MAP ist, dass die im Netzwerk übertragenen Daten durch das SOAP-Format vergleichsweise viel Bandbreite benötigen. Dies ist bei einer reinen Statusüberwachung des Netzes vernachlässigbar, führt jedoch bei einem SIEM-System zu Problemen. Die große Menge an Ereignissen, die von Sensoren wie zum Beispiel Snort geliefert wird, führt durch die Verwendung von SOAP und HTTP zu einer nicht zu unterschätzenden Menge an Overhead-Daten.

Im SIMU-Projekt wurde die Verwendung eines alternativen Datenformats erörtert. Die Concise Binary Object Representation (CBOR) bietet sich an dieser Stelle an. Sie basiert auf einem JSON-ähnlichen strukturierten Datenformat, das jedoch nicht als Klartext, sondern in Binärdarstellung übertragen wird. Daten in CBOR sind immer typisiert, was die Möglichkeit, fehlerhafte Daten zu übertragen, im Gegensatz zu XML/SOAP deutlich reduziert. Die Typdefinition ist dabei so schlank gehalten, dass viele einfache Datentypen inklusive der Definition gerade einmal ein Byte belegen.

Um die Umwandlung zwischen SOAP und CBOR zu gewährleisten, wurde im Rahmen des SIMU Projekts ein CBOR Schema definiert, das jede beliebige IF-MAP Datenstruktur abbilden kann. Durch die Verwendung dieses Schemas lassen sich bereits ohne Optimierungen ca. 10-20% der benötigten Bandbreite einsparen, da CBOR ohne Steuerzeichen wie Klammern, Anführungs- oder Leerzeichen auskommt. Dabei werden die Namen der XML Elemente direkt in CBOR übernommen, was zu einer großen Anzahl von String-Felder im CBOR Bytestrom führt, die entsprechend viel Bandbreite benötigen. Lediglich Datentypen wie IP- oder MAC-Adressen, Zahlen etc. werden auf die passenden CBOR Datentypen abgebildet.

Der nächste Schritt, um die Bandbreite zu reduzieren, besteht darin, die Anzahl dieser String-Felder zu reduzieren. Dazu wurde ein Wörterbuchmechanismus entwickelt, mit dem feste Begriffe aus dem IF-MAP Umfeld auf andere Datentypen abgebildet werden. Dies können die Namen von Elementen und Attributen oder auch Werte von ENUM-Feldern sein. Das im Rahmen des SIMU Projekts entwickelte Wörterbuch verwendet dabei eine Abbildung auf Integer-Werte. Durch die bereits erwähnte schlanke Typisierung ist es in CBOR möglich, positive Integer bis zu einem Wert von 23 inklusive Typdefinition in einem Byte darzustellen.

Dieser Wertebereich reicht aus, um selbst die komplexesten IF-MAP Strukturen abzubilden, da das Wörterbuch hierarchisch aufgebaut ist. So lassen sich zum Beispiel der IF-MAP Metadata Namespace, das Metadatum *ifmap:access-request-ip*, das Attribut *ifmap-timestamp* dieses Metadatums und der ENUM-Wert *singleValue* des *ifmap-cardinality* Attributs allesamt auf den Wert 1 abbilden, da sie ineinander verschachtelt werden und so keine Kollisionen auftreten können.

Durch die Verwendung eines solchen Wörterbuchs kann die benötigte Bandbreite pro Übertragung, je nach Beschaffenheit der Datenstruktur, um bis zu 80% reduziert werden. Natürlich müssen Sender und Empfänger das gleiche Wörterbuch verwenden, um die Daten später wieder entschlüsseln zu können. Bei dieser Einsparung ist der Overhead durch eine Übertragung via HTTP noch nicht einberechnet. Dieser würde ebenfalls wegfallen, da auf einer CBOR-zu-CBOR Strecke kein HTTP notwendig ist.

Der momentane Internet-Draft CDDL befindet sich in der 6. Revision und geht deutlich über den Anwendungsbereich von IF-MAP hinaus. Schmalbandige (gegenseitige) Attestation, Sensor/Aktuator- Kommunikation im ICS-Kontext oder Car-2-Car Payloads mittels 802.1p sind weitere Anwendungsdomänen für die im SIMU-Projekt vorangetriebene CBOR/CDDL-Repräsentation. Der aktuelle Draft findet sich unter [BIVI15] und wurde stark von Fraunhofer SIT innerhalb des SIMU-Projektes vorangetrieben.

3.3 Integration in bestehende Infrastrukturen

Das im SIMU-Projekt entwickelte System soll keine in sich geschlossene Appliance darstellen, sondern sich in bestehende Infrastrukturen einpassen lassen. Wenn in einem Netzwerk zum Beispiel bereits Dienste wie ein Snort IDS oder ein OpenVAS Schwachstellenscanner im Einsatz sind, so sollen diese Dienste nicht durch das SIEM-System erneut ausgerollt werden müssen, sondern die bestehenden Dienste sollten an das neue System angeschlossen werden.

Um dieses Ziel zu erreichen, wurden nahezu alle Sensoren, die Daten an den MAP-Server liefern, als schlanke Dienste implementiert. Diese werden auf den Maschinen der Sensor-Dienste installiert und lesen dort die in Log-Dateien oder Datenbanken gespeicherten Ausgaben der Sensoren aus. Die so gesammelten Informationen werden umgewandelt, so dass sie auf das erweiterte IF-MAP Metadatenschema passen und dann an den MAP-Server gesendet. Die auswertenden Komponenten greifen danach auf die im MAP-Server gespeicherten Informationen zu. Ein ähnlicher Ansatz wurde für die Visualisierungskomponenten, insbesondere die SIEM-GUI als primäre Benutzeroberfläche, verfolgt. Die SIEM-GUI lässt sich mit verschiedenen externen Diensten verbinden, die bestimmte Aufgaben übernehmen. So kann beispielsweise ein bereits vorhandenes Ticketsystem für die Speicherung der SIEM-Tickets verwendet werden, es muss kein zweites Ticketsystem ausgerollt werden. Gleiches gilt für die Benutzerverwaltung, auch hier sind verschiedene Dienste integrierbar. In der Demonstrator-Implementierung wurden diese Aufgaben jeweils vom Ticketsystem Request Tracker (RT) übernommen.

4 Beschreibung des Demonstrators

Der Demonstrator wurde so wie hier beschrieben im Rahmen des SIMU-Projekts implementiert und diente als Proof-of-Concept für ein IF-MAP-basiertes SIEM-System. Er stellt einen exemplarischen Use-Case dar, in dem ein Angriff im Netzwerk erkannt wird und das System entsprechend reagiert.

Im folgenden Abschnitt wird zunächst das Szenario beschrieben. In den Abschnitten 4.2 und 4.3 folgen dann Informationen zur Installation und Konfiguration der Demonstrator-Umgebung.

4.1 Szenario

Das Szenario des Demonstators besteht zum einen aus einigen Komponenten des SIMU-Systems und zum anderen aus zwei Client-Maschinen, die im Netzwerk authentifiziert wurden bzw. werden.

Als SIMU-Komponenten kommen dabei das *macmon NAC* und der *NCP VPN-Server* als Authentifikationsstellen zum Einsatz. Als Sensoren werden *ironnmap (nmap, HsH)*, *ironvas (OpenVAS, HsH)* und *DECOmap (Snort, DECOIT)* eingesetzt. Als MAP-Server wird der *irond* der Hochschule Hannover (HsH) verwendet. Der *VisITMeta Dataservice* (HsH) übernimmt die Aufzeichnung der MAP Daten. Die Korrelation übernimmt die GPM-Engine *irongpm* der HsH, die Anzeige und Verwaltung der Vorfälle wird durch die *SIEM-GUI* der DECOIT ermöglicht. Auf der Maschine läuft zusätzlich noch das benötigte *Ticketsystem RT* und die Datenbank für den Austausch von Vorfällen zwischen GPM-Engine und SIEM-GUI. Die Komponenten der HsH laufen dabei jeweils innerhalb einer eigenen virtuellen Maschine.

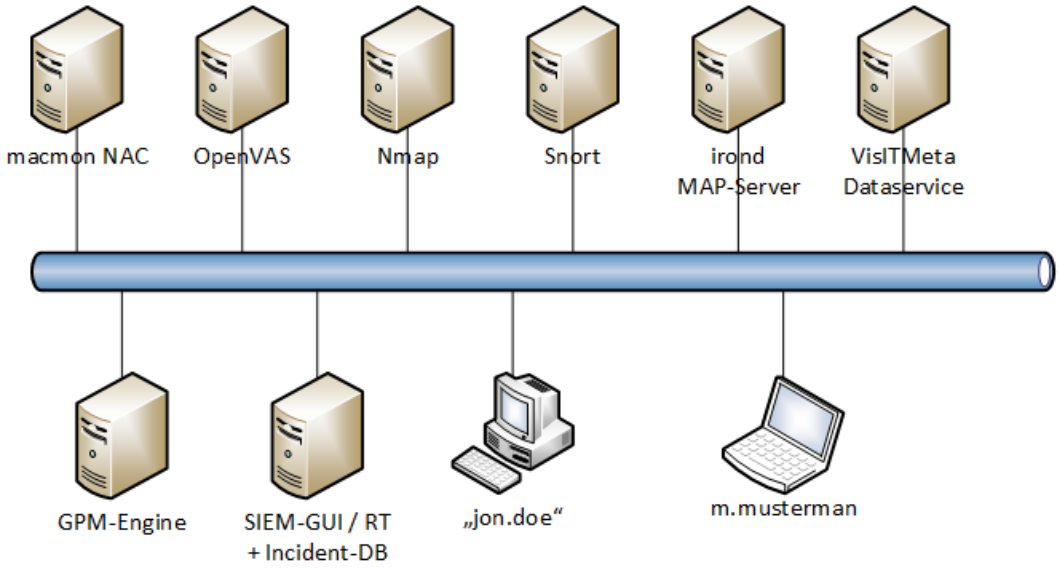

Abbildung 11: Demo Use-Case mit macmon NAC

Das Client-Gerät des Benutzers „jon.doe“ befindet sich zu Beginn des Szenarios bereits im Netzwerk. Es wurde über das macmon NAC authentifiziert und hostet einen Postfix-Service auf Port 25, der über die Schwachstelle CVE-2015-1234 verwundbar

ist. Der Service wird durch einen Scan von ironnmap erkannt, die vorhandene Schwachstelle durch einen OpenVAS-Scan, der von ironvas ausgewertet wird.

Der Benutzer „m.musterman“ meldet sich nach einiger Zeit im Netzwerk an. Dabei kann dies zum einen über das macmon NAC geschehen, was einem neuen Gerät entspricht, das direkt mit dem Firmennetzwerk verbunden ist.

Zum anderen ist es auch möglich, dass dieser Benutzer sich über den NCP VPN-Server mit dem Netzwerk verbindet, dabei könnte es sich zum Beispiel über einen Außendienstmitarbeiter handeln.

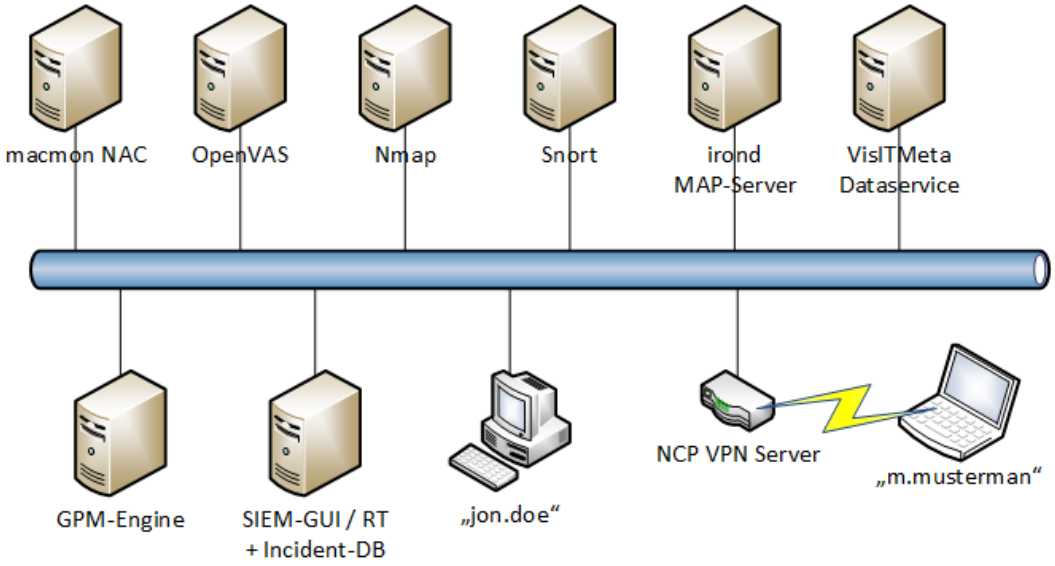

Abbildung 12: Demo Use-Case mit NCP VPN Server

Die Authentifizierung des neuen Benutzers ist in beiden Fällen erfolgreich. Dieser beginnt danach damit, die auf „jon.doe“ gefundene Schwachstelle auszunutzen und einen Angriff auf das Gerät durchzuführen.

Dieser Angriff wird von Snort erkannt und mit der passenden CVE-ID gemeldet, was durch den DECOmap Snort MAP-Client als „attack-detected“ Metadatum zum MAP-Server gesendet wird. Die GPM-Engine enthält eine Regel, die auf die Korrelation zwischen der CVE-ID des Angriffs und der CVE-ID der erkannten Schwachstelle auf dem angegriffenen Gerät achtet. Da dieser Fall eingetreten ist, wird ein Vorfall erzeugt, der in der Austauschdatenbank auf der Maschine der SIEM-GUI hinterlegt wird. Zudem erzeugt die GPM-Engine ein „unexpected-behavior“ Metadatum an der IP-Adresse des Angreifers und sendet dies an den MAP-Server. Durch dieses Metadatum könnte zum Beispiel ein automatisiertes Enforcement ausgelöst werden, wenn dies erwünscht ist. Das Enforcement ist aber nicht Teil dieses Use-Case.

Die SIEM-GUI informiert den Administrator durch eine Benachrichtigung über den neuen Vorfall. Dieser kann nun mit Hilfe der Informationen, die von GPM-Engine dem Vorfall beigefügt wurden, sie Situation bewerten, den Angreifer ausfindig machen und das Problem beheben.

Dabei helfen ihm zum einen die Beschreibung des Vorfalls und ein Lösungsvorschlag, die bei beiden von der GPM-Engine zusammen mit dem Vorfall generiert werden. Weiterhin ist im Vorfall der Teilgraph des MAP-Servers enthalten, der die Regel ausgelöst hat. Durch eine JavaScript Version von VisITMeta, die in der SIEM-GUI integriert ist, kann dieser Teilgraph angezeigt und ausgewertet werden.

Der Ablauf des Demonstrators wurde auch mit dem Demo-Video dokumentiert und beschrieben. Dieses kann unter der folgenden URL-Adresse abgerufen werden: https://www.youtube.com/watch?v=uFED-U0LzmA&feature=youtu.be

Abbildung 13: SIMU Use-Case Demonstration

Die Abbildung 14 verdeutlicht den Datenfluss zwischen den SIMU-Komponenten in diesem Demo-Szenario.

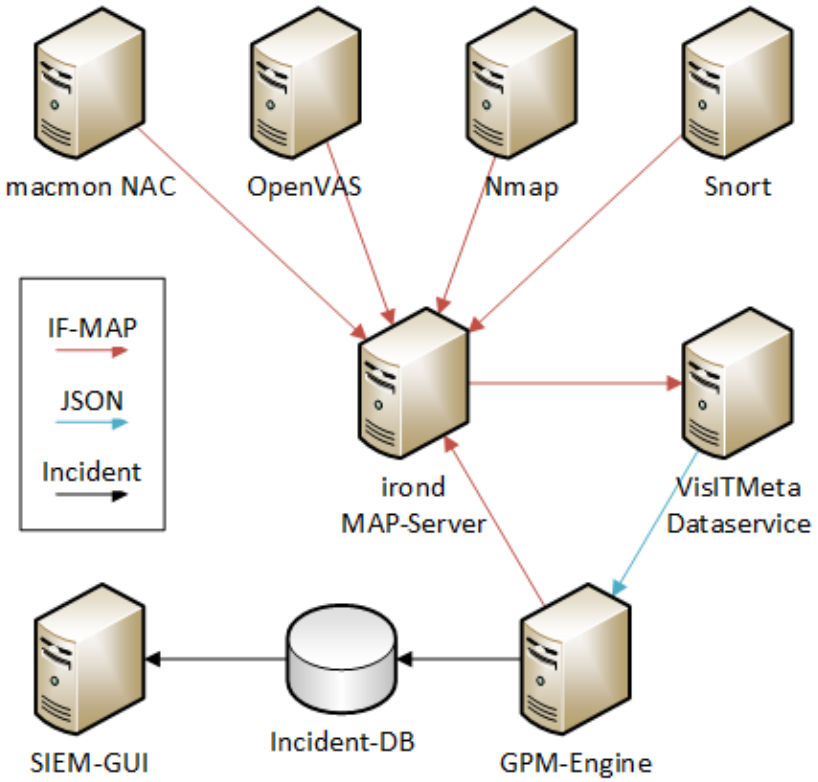

Abbildung 14: Datenfluss zwischen den SIMU-Komponenten im Demo-Szenario

4.2 Demonstrator-Umgebung installieren

Der folgende Abschnitt stellt eine kurze Anleitung dar, wie die Umgebung für den Demonstrator installiert werden kann.

Hinweis: Die aktuell verfügbare Demo-Version des NCP VPN-Systems (Stand: Oktober 2015) enthält noch nicht die Entwicklungen aus dem SIMU-Projekt. Bitte wenden Sie sich an support@ncp-e.de, um alle benötigten Komponenten zu erhalten.

Die Demo-Version des macmon secure NAC enthält ebenfalls nicht alle notwendigen Funktionen, um dieses Demonstrator-Szenario nachzustellen. Bitte setzen Sie sich für weitere Informationen direkt mit macmon secure in Verbindung: http://www.macmon.eu/kontakt/kontaktformular/

Für die Installation des Demonstrators werden einige Rechner benötigt, daher bietet es sich an, hierfür eine virtuelle Umgebung zu nutzen. Es ist natürlich auch möglich, physikalische Maschinen zu verwenden. Im SIMU-Projekt wurde für den Aufbau des Demonstrators eine Proxmox-/KVM-Umgebung eingerichtet und verwendet.

Auf der Website des SIMU-Projekts http://www.simu-project.de lassen sich Links zu allen im Folgenden genannten Produkten finden. Viele sind in Java geschrieben und müssen selbst kompiliert werden, daher werden ein Java JDK in der Version 8[2] und Apache Maven in der Version 3[3] benötigt. Da die meisten Projekte auf Github veröffentlich wurden, bietet sich für den Download das Programm git an. Es ist jedoch auch möglich ZIP-Dateien mit dem Quellcode von Github herunter zu laden.

4.2.1 MAP-Server und VisITMeta Dataservice

Die erste Maschine, die aufgesetzt werden muss, beherbergt den MAP-Server und den VisITMeta-Dataservice. Diese beiden Dienste können ohne Probleme auf der gleichen Maschine betrieben werden. Die Github-Seiten enthalten detaillierte Anweisungen zum Kompilieren und Installieren der Komponenten, daher soll an dieser Stelle auf diese Anleitungen verwiesen werden.

Weiterhin ist es sinnvoll, die Visualisierungskomponente von VisITMeta auf einer Maschine mit graphischer Oberfläche zu installieren, da so die Fehlersuche über den MAP-Graphen stark vereinfacht wird.

4.2.2 Open-Source Sensoren und IF-MAP Clients

Als Open-Source Sensoren kommen in diesem Szenario die Tools Snort[4], Nmap[5] und OpenVAS[6] zum Einsatz. Jedes dieser Programme sollte auf einer eigenen Maschine installiert werden, da es sonst zu Problemen und Kollisionen im MAP-Server kommen kann. Die Installation dieser Tools hängt stark vom verwendeten Basissystem ab, daher sollte hier die Dokumentation der jeweiligen Entwickler herangezogen werden. Im Rahmen des SIMU-Projekts wurden die Maschinen mit einer Server-Installation von Ubuntu 14.04 ohne eigene Anpassungen betrieben.

Die Anbindung der Sensoren erfolgt über kleine IF-MAP-Clients, die in Java geschrieben wurden und die Informationen der Sensoren auslesen, verarbeiten und an den MAP-Server senden. Die Clients müssen jeweils auf der gleichen Maschine wie die Sensoren installiert werden, da sie in der Regel lokale Log-Dateien auslesen.

Für das Snort IDS wird der DECOmap IF-MAP-Client der DECOIT GmbH benötigt. Für Nmap und OpenVAS werden die jeweiligen IF-MAP-Clients der Hochschule Hannover ironnmap bzw. ironvas benötigt. Alle diese Clients sind auf Github verfügbar, Links zu den Projekten finden sich auf der Website des SIMU Projekts.

4.2.3 Graph-Pattern-Matching Engine

Die Graph-Pattern-Matching (GPM) Engine irongpm der Hochschule Hannover ist die korrelierende Komponente des SIMU-Systems. Sie sollte auf einer eigenen Maschine

[2] http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
[3] https://maven.apache.org
[4] https://www.snort.org
[5] https://nmap.org
[6] http://www.openvas.org/index.de.html

installiert werden, kann alternativ aber auch auf der gleichen Maschine wie der MAP-Server betrieben werden.

4.2.4 SIEM-GUI, RT und Vorfalldatenbank

Für die SIEM-GUI müssen einige zusätzliche Dienste installiert werden. Außerdem ist es sinnvoll, auf der gleichen Maschine die Vorfalldatenbank zu installieren.

Als Datenbank für den Austausch zwischen Graph-Pattern-Matching Engine und SIEM-GUI wird eine einfache MySQL-Datenbank verwendet. Diese kann üblicherweise direkt über die Paketquellen der Linux-Distribution installiert werden.

Das Ticketsystem Request Tracker (RT) ist für die SIEM-GUI notwendig, da darüber der Fortschritt bei der Bearbeitung von Vorfällen gespeichert wird. RT kann auf der Website des Herstellers Best Practical[7] heruntergeladen werden. Die im Paket enthaltene README Datei enthält eine Anleitung zur Installation und Einrichtung. Als Datenbank kann die zuvor installierte MySQL Datenbank verwendet werden.

Um die SIEM-GUI starten zu können, wird ein installierter Tomcat 8 Application Server[8] mit Java in der Version 8 benötigt. Weiterhin ist es empfehlenswert, die Manager Web Application einzurichten, da damit das Ausrollen der SIEM-GUI stark vereinfacht wird. Weiterhin wird Node.js in Version 0.10 oder 4.x inklusive des dazugehörigen Paketmanagers NPM benötigt.

Um die in der SIEM-GUI enthaltene JavaScript-Version von VisITMeta kompilieren zu können, muss das RPM-Paket *gulp* global installiert werden.

```
sudo npm install gulp -g
```

Danach kann das SIEM-GUI Repository mit git geklont werden oder der Quellcode an eine beliebige Stelle entpackt werden. Im Verzeichnis des Quellcodes müssen die folgenden Befehle ausgeführt werden, um VisITMeta zu bauen:

```
cd src/main/webapp/visitmeta
npm install
gulp run
```

Der dabei gestartete lokale Webserver kann einfach mit *Ctrl-C* beendet werden, dieser wird nicht benötigt.

4.2.5 Proprietäre Komponenten

Da die Komponenten von *macmon secure* und *NCP Engineering* nicht ohne weiteres in Betrieb genommen werden können, und sich auch durch aktuelle Releases immer wieder Änderungen ergeben werden, wenden Sie sich für Informationen diesbezüglich bitte direkt an die genannten Hersteller der Komponenten:

[7] https://www.bestpractical.com/rt/
[8] http://tomcat.apache.org

a. NCP Dwonload: https://www.ncp-e.com/de/service/download-vpn-client.html,
b. macmom secure: http://www.macmon.eu/downloads/software/.

4.3 Konfiguration der Komponenten

Wie die einzelnen Komponenten konfiguriert werden müssen, um das Demonstrator-Szenario nachstellen zu können, wird in den folgenden Abschnitten erläutert. Für Hinweise zur Konfiguration der proprietären Komponenten, wenden Sie sich bitte wie gehabt an die entsprechenden Hersteller.

4.3.1 MAP-Server irond

An der Konfiguration für den MAP-Server irond muss für den Demonstrator an den voreingestellten Parametern wenig verändert werden. Standardmäßig wird der MAP Server auf den Ports 8443 (Authentifizierung per Basic Authentication) bzw. 8444 (Authentifizierung per Zertifikat) gestartet. Es ist aber darauf zu achten, dass entsprechend der verwendeten Komponenten und Authentifizierungsmethode, die *basicauthusers.properties* alle notwendigen Benutzer-Passwort-Kombinationen beinhaltet.

Für das Zusammenspiel mit der VisITMeta-Visualisierung ist außerdem die Datei *publisher.properties* wichtig. Der MAP Server sammelt dort im Laufe des Betriebs für jeden MAP Client die vergebenen Publisher ID; meldet sich derselbe Client erneut an, wird ein ggf. vorhandener Eintrag wiederverwendet. Darauf basierend kann in der Visualisierung für jede Publisher ID ein eigener Farbwert für die veröffentlichten Metadaten konfiguriert werden. Durch diese Datei ist eine einmal konfigurierte Situation auch auf andere Maschinen übertragbar, ohne die Farbzuweisung neu vornehmen zu müssen. Im Testbed wurden folgende Publisher IDs vergeben:

```
decoit=decoit-677f8d81-4bcf-48a2-b6ff-fda6201cdc7c
dhcp=dhcp-4d02574f-888a-4a09-8b6b-a87ae38ce69b
iptables=iptables-d6bd8aff-0036-4fe1-9a22-f8e5984af112
irongpm=irongpm-f91af8bb-b0a1-48fe-b952-57479efb99b9
ironvas=ironvas-bf31fd64-634d-4d0e-b255-05d6c15d75da
macmon=macmon-51dc8be1-3655-40a7-82b8-3284590dfc28
ncp=ncp-65a22a85-bdb8-4137-bda7-033b66f62da7
nmap=nmap-a6dfdcc8-8495-42b5-b0b9-932e2265016f
pdp=pdp-4f00a270-dd47-480c-bd02-e1574dad7e12
snort=snort-28ef329d-37f0-44e9-95b9-b48fc337a84c
test=test-0eb4f123-605f-4eae-9f35-83c3dea2cdf9
visitmeta=visitmeta-eaf0ad5b-fed3-4b00-a6ee-5c75badf50da
visual=visual-e4acdb4d-08d0-4b08-a15a-ddcd1dd7d054
```

4.3.2 VisITMeta Dataservice und Visualisierung

Der VisITMeta Dataservice und die Komponente zur Visualisierung können grundsätzlich ebenfalls auf der gleichen Maschine wie der MAP Server laufen, aber auch getrennt voneinander installiert werden.

Hierzu müssen die Konfigurationsparameter für die Verbindung zum Dataservice, bzw. zur Subscription am MAP-Server entsprechend gesetzt werden.

4.3.2.1 Dataservice

Für den Dataservice muss vor allem die Verbindung zum MAP-Server konfiguriert werden. Dies geschieht in der Datei *dataservice_connections.yml* (oder kann alternativ auch über die grafische Oberfläche der Visualisierung erfolgen). Folgende Konfiguration für die Subscriptions auf die verschiedenen Komponenten der Hersteller war dafür beispielsweise in dem Testbed vorhanden:

```
testbed:
  userName: visitmeta
  userPassword: visitmeta
  ifmapServerUrl: https://10.148.149.28:8443
  useConnectionAsStartup: true
  subscriptions:
    macmon:
      startIdentifier: IPv4,10.148.149.32
      identifierType: ip-address
      useSubscriptionAsStartup: true
    macmondev:
      startIdentifier: macmon-687713ed-5283-4163-b21d-0b7c72ce6908
      identifierType: device
      useSubscriptionAsStartup: true
    nmap:
      startIdentifier: nmap
      identifierType: device
      useSubscriptionAsStartup: true
    ncpvpn:
      startIdentifier: IPv4,10.148.148.125
      identifierType: ip-address
      useSubscriptionAsStartup: true
```

Der Name der angelegten Connection lautet *testbed*, die nächsten Zeilen beschreiben allgemein die Verbindung zum MAP-Server (username, password und URL). Anschließend werden die einzelnen Subscriptions für die einzelnen Geräte der Infrastruktur festgelegt. Für die Verbindung sowie die Subscriptions wurde konfiguriert, dass diese bei Start der Dataservice-Anwendung automatisch aktiviert werden.

4.3.2.2 Visualisierung

In der Visualisierung muss lediglich die Verbindung zu dem Dataservice der *dataservices.yml* unter dem folgenden Parameter eingestellt werden:

```
dataserviceRestUrl: http://10.148.149.30:8000
```

Die Einstellung der Farben für die entsprechenden Publisher, so wie in dem Demonstrator-Video gezeigt, kann über die GUI erfolgen oder mittels der Datei *visualization_config.yml* angepasst werden. Wie beim MAP Server irond schon erwähnt, werden für die Farbzuweisungen die Publisher IDs aus der *publisher.properties* Datei verwendet. Die im Testbed verwendeten Farben für die Clients sind:

```
  irongpm-f91af8bb-b0a1-48fe-b952-57479efb99b9:
    inside: '0xD1686D'
    outside: '0xD1686D'
```

```
  dhcp-4d02574f-888a-4a09-8b6b-a87ae38ce69b:
    inside: '0xF7C2AB'
    outside: '0xF7C2AB'
  pdp-4f00a270-dd47-480c-bd02-e1574dad7e12:
    inside: '0xF7F4AD'
    outside: '0xF7F4AD'
  test-0eb4f123-605f-4eae-9f35-83c3dea2cdf9:
    inside: '0x96D6CE'
    outside: '0x96D6CE'
  ironvas-bf31fd64-634d-4d0e-b255-05d6c15d75da:
    inside: '0xff3300'
    outside: '0xff3300'
  macmon-51dc8be1-3655-40a7-82b8-3284590dfc28:
    inside: '0x5FAE57'
    outside: '0x5FAE57'
  snort-28ef329d-37f0-44e9-95b9-b48fc337a84c:
    inside: '0x00cc00'
    outside: '0x00cc00'
  ncp-65a22a85-bdb8-4137-bda7-033b66f62da7:
    inside: '0x5FAE57'
    outside: '0x5FAE57'
  iptables-d6bd8aff-0036-4fe1-9a22-f8e5984af112:
    inside: '0x789078'
    outside: '0x789078'
  nmap-a6dfdcc8-8495-42b5-b0b9-932e2265016f:
    inside: '0xffcc00'
    outside: '0xffcc00'
```

4.3.3 Sensor: Snort und DECOmap IF-MAP-Client

Um den Snort-Sensor in Betrieb zu nehmen, müssen sowohl das Snort IDS, als auch der DECOmap IF-MAP-Client entsprechend konfiguriert werden.

4.3.3.1 Snort IDS

Für den Demonstrator reicht eine Basiskonfiguration des Snort IDS aus, da nur grundlegende Funktionen benötigt werden. Dazu ist zunächst in der Konfigurationsdatei */etc/snort/snort.conf* die Option *HOME_NET* so zu konfigurieren, dass sie auf das Netzwerk, in dem alle angelegten Maschinen liegen, zeigt.

Beispiel aus dem Projekt:

```
ipvar HOME_NET 10.148.148.0/23
```

Soll nicht die Snort-Maschine selbst, sondern eine andere Maschine, angegriffen werden, so muss das Netzwerk-Interface in den *Promiscuous Mode* geschaltet werden. Dies ist notwendig, damit Snort den Netzwerkverkehr erkennen kann, der nicht für die eigene Maschine bestimmt ist. Ist die Snort-Maschine selbst das Angriffsziel, so ist dieser Schritt nicht notwendig.

Weiterhin sollte das Regelwerk von Snort vollständig gelöscht werden und danach mit einer einzigen Regel gefüllt werden. Dazu reicht es aus alle Dateien außer der *local.rules* im Verzeichnis */etc/snort/rules/* zu löschen. Danach muss diese verbliebene Datei mit einem Texteditor geöffnet und mit der Demonstrator-Regel gefüllt werden. In

diesem Szenario wird einfach auf Netzwerkverkehr auf Port 25 der Zielmaschine reagiert und eine Meldung mit einer fiktiven CVE-ID generiert. Diese Regel sieht folgendermaßen aus, wobei die IP-Adresse der tatsächlichen Adresse des Ziels entsprechen muss:

```
alert tcp any any -> 10.148.149.110 25 (msg:"SMTP attack detected";
sid:10000002; rev:002; reference:cve,2015-1234;)
```

Die gesamte Regel muss in einer Zeile stehen, d.h. es dürfen keine Umbrüche enthalten sein. Danach muss die Snort-Konfiguration neu geladen werden. Dies erreicht man, indem man den Snort-Prozess per Kommandozeile beendet. Er startet sich dann von selbst neu:

```
sudo kill –SIGHUP SNORT-PROCESS-ID
```

Die Prozess-ID des Snort IDS lässt sich zum Beispiel über folgenden Befehl herausfinden:

```
ps aux | grep snort
```

4.3.3.2 DECOmap IF-MAP-Client

Für den DECOmap IF-MAP-Client müssen ebenfalls einige Einstellungen angepasst werden, damit das Snort-Modul und das Attack-Detected-Metadatum verwendet werden können.

Die Konfiguration befindet sich im Installationsverzeichnis des IF-MAP-Clients im Verzeichnis *config/*. Zunächst muss die Datei *config.properties* angepasst werden. Dazu muss eine Kopie der *config.properties.tpl* angelegt und die folgenden Einstellungen geändert werden.

```
cp config.properties.tpl config.properties
```

Der Text *SNORT-MACHINE-IP-ADDRESS* muss durch die tatsächliche IP-Adresse der Snort-Maschine ersetzt werden, *MAPS-ADDRESS* durch die HTTP-Adresse des MAP-Servers inklusive Port. Die beiden Platzhalter *BASICAUTH-USER* und *BASICAUTH-PASS* sind durch entsprechende Werte für die Anmeldung am MAP-Server zu ersetzen. Diese dürfen von keinem anderen IF-MAP-Client im Netzwerk verwendet werden!

```
application.ipaddress=SNORT-MACHINE-IP-ADDRESS
application.isservice=false
application.component=SNORT_AD_FILE
application.pollingconfig.path=config/snort/file_polling.properties
application.mappingconfig.path=config/snort/mapping.properties
application.regexconfig.path=config/snort/regex.properties
mapserver.url=MAPS-ADDRESS
mapserver.keystore.path=/keystore/snortmap.jks
mapserver.keystore.password=snortmap
```

```
mapserver.truststore.path=/keystore/snortmap.jks
mapserver.truststore.password=snortmap
mapserver.basicauth.enabled=true
mapserver.basicauth.user=BASICAUTH-USER
mapserver.basicauth.password=BASICAUTH-PASS
```

Danach muss noch in der Datei *config/snort/file_polling.properties* die Alert-Logfile von Snort eingetragen werden. Der Pfad kann je nach Installation und Basissystem variieren.

```
filepath=/var/log/snort/alertlog
```

Nach der Konfiguration kann der IF-MAP Client mit folgendem Befehl gestartet werden. Das Logging kann in der Datei *config/logging.properties* angepasst werden, falls mehr oder weniger Ausgaben gewünscht sind.

```
java –jar decomap-0.2.0.0.jar
```

4.3.4 Sensor: Nmap und ironnmap

Um den ironnmap-Client zu verwenden, muss zunächst nmap für das System installiert werden. Anschließend kann ironnmap per Maven und Java gebaut werden. In der Konfiguration per ironnmap.yml. Datei muss die Adresse sowie Zugangsdaten zum MAP Server und der Name eines device-Identifiers angegeben werden, der per Subscription auf das Auftreten von *request-for-investigation* Metadaten beobachtet wird. Im Testbed wurde hierzu der device-Identifier des Macmon-NAC-Gerätes verwendet. Beim Auftreten wird dann ein nmap-Scan für die verbundene IP-Adresse gestartet und die Ergebnisse hinterher als SIMU-Metadatenteilgraph mit service und implementation-Identifiern veröffentlicht.

```
ifmap:
    server:
        url:
            basic: https://10.148.149.28:8443
            cert: https://10.148.149.28:8444
        auth:
            method: cert
            user: ironnmap
            password: ironnmap

    subscriber:
      # the name of the PDP
      subscriptionroot: macmon-687713ed-5283-4163-b21d-0b7c72ce6908
```

4.3.5 Sensor: OpenVAS und ironvas

Für die Verwendung von OpenVAS in Kombination mit dem ironvas MAP Client muss zunächst OpenVAS installiert und konfiguriert werden (siehe dazu offzielle Anleitungen für OpenVAS).

Als Zugangspunkt für ironvas muss eine Scankonfiguration in OpenVAS (z.B. per Management Weboberfläche) angelegt werden, die später unter diesem Namen in der Konfiguration von ironvas (per *configuration.properties* Datei) referenziert werden muss. Im Testbed-Fall ist dies die Scanconfig namens *ironvas-config*.

Ironnmap muss neben der Anbindung an den MAP Server (URL sowie Zugangsdaten) zudem in seinen Funktionen als Publisher und Subscriber konfiguriert werden.

```
# OpenVAS Server ip address
openvas.server.ip = 127.0.0.1

# OpenVAS Manager port to connect to
openvas.server.omp.port = 9390

# OpenVAS Manager credentials
openvas.server.omp.user = admin
openvas.server.omp.password = admin

# IF-MAP authentication method, valid values are 'cert' and 'basic'
ifmap.server.auth.method = cert

# IF-MAP connection information
ifmap.server.url.basic = https://10.148.149.28:8443
ifmap.server.url.cert = https://10.148.149.28:8444
ifmap.server.auth.basic.user = ironvas
ifmap.server.auth.basic.password = ironvas

# the path to the keystore
keystore.path = /ironvas.jks

# the keystore password
keystore.password = ironvas

# the interval between two renewSession commands to the MAPS
ironvas.ifmap.interval = 120
```

Der Subscriber lauscht auf *request-for-investigation* Metadaten an einem definierten Device, im Testbedfall war dies das macmon-Device. Bei jedem Auftreten dieses Metadatums wird über die Verbindung zur IP-Adresse im MAP Graphen diese ausgewertet und in OpenVAS ein neuer Scan mit dieser IP-Adresse als Ziel und der definierten Scanconfig angelegt und ausgeführt.

```
# ---- subscriber specific ---- #

# activate the subscriber?
ironvas.subscriber.enable = true

# the name of the PDP
ironvas.subscriber.pdp = macmon-687713ed-5283-4163-b21d-0b7c72ce6908

# a prefix used for the OpenVAS target and task
ironvas.subscriber.namePrefix = ironvas:

# the name of the OpenVAS configuration which will be used for new tasks
ironvas.subscriber.config = ironvas-config
```

Der Publisher veröffentlicht die gefundenen Schwachstellenberichte wahlweise im ESUKOM-Format (als Feature-Metadatum) oder als IF-MAP Standard-Event. Im Testbed wurde das Event-Format verwendet.

```
# ---- publisher specific ---- #

# activate the publisher?
ironvas.publisher.enable = true

# the interval in seconds in which ironvas will fetch the OpenVAS reports
ironvas.omp.interval = 60

# The class name of the converter. The class must have a no-argument default
# constructor and implement the Converter interface.
ironvas.publish.converter =
de.hshannover.f4.trust.ironvas.converter.EventUpdateConverter
#ironvas.publish.converter =
de.hshannover.f4.trust.ironvas.converter.EsukomFeatureConverter
```

4.3.6 Graph-Pattern-Matching Engine: irongpm

Zur Verwendung der Graph-Pattern-Matching Engine müssen zunächst die Verbindungen zu den anderen Komponenten konfiguriert werden. Dies betrifft auf der einen Seite die IF-MAP Komponenten: Den VisITMeta-Dataservice zur Abfrage von Daten, sowie den MAP-Server irond zur Veröffentlichung neuer Informationen. Außerdem muss die Verbindung zur Datenbank der SIEM-GUI eingestellt werden um neue Vorfälle erzeugen und persistieren zu können.

In der *irongpm.yml* können zunächst die Parameter für die IF-MAP Komponenten angepasst werden. Für den Dataservice:

```
dataservice:
  url: http://10.148.149.30:8000
  connection: testbed
  rawxml: true
```

Und die Verbindung zum MAP-Server:

```
ifmap:
  auth:
    method: basic
    basic:
      url: https://10.148.149.28:8443
      user: irongpm
      password: irongpm
    cert:
      url: https://127.0.0.1:8444
  truststore:
    path: /irongpm.jks
    password: irongpm
  # connection uses a thread-safe SSRC
  threadsafe: true
  initialconnectiontimeout: 120000
```

Die Konfiguration für die Datenbank erfolgt entsprechend der Verwendung des Hibernate Frameworks in der *hibernate.cfg.xml.* Besonders relevant sind hier die Verbindungsparameter:

```
<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect
</property>
<property name="hibernate.connection.url">jdbc:mysql://10.148.149.145/simu
</property>
<property name="hibernate.connection.username">simu_user</property>
<property name="hibernate.connection.password">decoit2015</property>
<property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver
</property>
<mapping class="de.decoit.simu.incidents.entities.IncidentEntity" />
</session-factory>
```

Zu beachten ist auch, dass die IncidentEntity-Klasse der SIEM-GUI für das Mapping vorhanden sein muss!

Damit ist die irongpm-Komponente auch bereits lauffähig. Zu guter Letzt sollten natürlich noch Regeln für das Regelwerk erstellt und eingebunden werden. Hierzu muss der Reflection Mechanismus genutzt werden. Es muss also eine Implementierung des *RuleLoader*-Interface in dem *rules* Ordner abgelegt werden. Die Regeln des Demonstrators werden in der Klasse *SimuDemonstrationRuleLoader* auf GitHub innerhalb des *irongpm*-Projektes unter *irongpm-rules* mit zur Verfügung gestellt.

4.3.7 Request Tracker

Um das Ticketsystem RT sowohl als Ticketsystem als auch als Benutzerverwaltung nutzen zu können, müssen in RT einige Einstellungen vorgenommen werden. Diese sind notwendig, damit die SIEM-GUI alle notwendigen Operationen durchführen kann und nicht von der RT REST API abgewiesen wird.

4.3.7.1 Mail-Server Dienst

RT benötigt auf dem Server, auf dem es installiert ist, einen funktionierenden SMTP-Server. Bei vielen Linux-Systemen ist dazu Sendmail installiert. Dieses sollte jedoch gegen Postfix ausgetauscht werden, da Sendmail Probleme verursachen kann, die beim Zugriff auf die RT REST API zu Internal Server Errors führen können.

Insbesondere tritt dies auf, wenn E-Mails an unbekannte Adressen geschickt werden sollen oder der eigene Hostname nicht korrekt aufgelöst werden kann. Sendmail scheint mehrere Versuche zu unternehmen, diese Operationen auszuführen und schläft zwischen den Versuchen. Wird nun versucht, eine weitere E-Mail zu verschicken, so kommt es zum Internal Server Error. Bei Postfix tritt dieses Problem nicht auf, daher ist dieser SMTP-Server vorzuziehen.

4.3.7.2 Benutzergruppen

Es muss eine Benutzergruppe für alle Benutzer der SIEM-GUI angelegt werden. Als Name wird „SIEM Users“ vorgeschlagen. Entsprechend müssen alle Benutzer, die Zugriff auf die GUI erhalten sollen, Mitglied in dieser Gruppe sein.

Weitere Einstellungen sind für diese Gruppe nicht notwendig.

4.3.7.3 Custom Fields

Um die Funktion der SIEM-GUI sicherzustellen, sind einige Custom Fields (Benutzerdefinierte Felder) zu erstellen und den entsprechenden Objekten zuzuordnen. Notwendig ist dies für Tickets und Benutzer, weshalb diese beiden Bereiche im Folgenden getrennt betrachtet werden. Die Namen für die Custom Fields können von den in diesem Abschnitt angegebenen Namen abweichen, müsse dann jedoch in der *siem-gui.properties* entsprechend angepasst werden.

Custom Fields für Tickets

Zum einen wird ein Custom Field „Risk“ benötigt, in dem die Risikoeinstufung eines Vorfalls abgelegt wird. Folgende Einstellungen müssen vorgenommen werden:

Attribut (englisch / deutsch)	Wert (englisch / deutsch)
Name	Risk
Description / Beschreibung	*beliebig*
Type / Typ	Enter one value / Einen Wert eingeben
Applies to / Gilt für	Tickets / Anfragen
Validation / Validierung	(?#Digits)^[\d.]+$
Enabled / Aktiviert	Ja

Tabelle 1: Custom Field Ticket: Risk, Basics / Grundsätzliches

Im Tab „Applies to/Gilt für“ muss das Custom Field auf alle Tickets angewendet werden, auch bestehende. Im Tab „Group Rights/Gruppenrechte“ muss zusätzlich die Option „View custom fields/Benutzerdefinierte Felder sehen“ für die oben erstellte Gruppe „SIEM Users“ aktiviert sein.

Das zweite Custom Field ist „Incident“, das die ID des Vorfalls der zu dem dieses Ticket gehört, gespeichert wird. Folgende Einstellungen müssen vorgenommen werden:

Attribut (englisch / deutsch)	Wert (englisch / deutsch)
Name	Incident
Description / Beschreibung	*beliebig*
Type / Typ	Enter one value / Einen Wert eingeben
Applies to / Gilt für	Tickets / Anfragen
Validation / Validierung	(?#Digits)^[\d.]+$
Enabled / Aktiviert	Ja

Tabelle 2: Custom Field Ticket:Incident, Basics / Grundsätzliches

Im Tab „Applies to/Gilt für“ muss das Custom Field auf alle Tickets angewendet werden, auch bestehende. Im Tab „Group Rights/Gruppenrechte“ muss zusätzlich die Option „View custom fields/Benutzerdefinierte Felder sehen“ für die oben erstellte Gruppe „SIEM Users“ aktiviert sein.

Custom Fields für Benutzer

Für Benutzer werden ebenfalls zwei Custom Fields benötigt. Zum einen ist dies das Feld „SIEM“, das steuert, ob ein Benutzer Zugriff auf die SIEM-GUI hat. Für dieses Feld müssen die folgenden Einstellungen vorgenommen werden:

Attribut (englisch / deutsch)	**Wert (englisch / deutsch)**
Name	SIEM
Description / Beschreibung	*beliebig*
Type / Typ	Select one value/Einen Wert auswählen
Render Type / Anzeige Typ	Select box
Applies to / Gilt für	Users / Benutzer
Enabled / Aktiviert	Ja

Tabelle 3: Custom Field User:SIEM, Basics / Grundsätzliches

Als Werte müssen für dieses Custom Field hinzugefügt werden:

Sort / Sortieren	**Name**	**Description / Beschreibung**
0	0	No
1	1	Yes

Tabelle 4: Custom Field User:SIEM, Werte

Im Tab „Group Rights/Gruppenrechte“ muss zusätzlich die Option „View custom fields/Benutzerdefinierte Felder sehen“ für die oben erstellte Gruppe „SIEM Users“ aktiviert sein. Als zweites wird das Custom Field „SIEMADMIN“ benötigt, über das gesteuert wird, ob ein Benutzer den Admin-Bereich in der SIEM-GUI einsehen kann. Für dieses Feld müssen die folgenden Einstellungen vorgenommen werden:

Attribut (englisch / deutsch)	**Wert (englisch / deutsch)**
Name	SIEMADMIN
Description / Beschreibung	*beliebig*
Type / Typ	Select one value/Einen Wert auswählen
Render Type / Anzeige Typ	Select box
Applies to / Gilt für	Users / Benutzer
Enabled / Aktiviert	Ja

Tabelle 5: Custom Field User:SIEMADMIN, Basics / Grundsätzliches

Als Werte müssen für dieses Custom Field hinzugefügt werden:

Sort / Sortieren	Name	Description / Beschreibung
0	0	No
1	1	Yes

Tabelle 6: Custom Field User:SIEMADMIN, Werte

Im Tab „Group Rights/Gruppenrechte“ muss zusätzlich die Option „View custom fields/Benutzerdefinierte Felder sehen“ für die oben erstellte Gruppe „SIEM Users“ aktiviert sein.

4.3.7.4 Benutzer

Neben den Benutzern, die als Anwender auf die SIEM-GUI zugreifen sollen, muss noch ein System-Account für die GUI eingerichtet werden. Als Benutzername für diesen Account wird „siemsystem“ vorgeschlagen, als Sprache muss „English“ eingestellt sein. Die Custom Fields „SIEM“ und „SIEMADMIN“ sollten jeweils auf 0 gesetzt sein. So wird verhindert, dass sich jemand mit diesem Account unberechtigt an der SIEM-GUI anmeldet. Weiterhin muss dieser System-Account ebenfalls Mitglied in der oben genannten Benutzergruppe „SIEM Users“ sein.

RT schreibt vor das alle Benutzer, die die RT REST API verwenden sollen, als „privilegiert“ (privileged) markiert sind. Bei einer bestehenden Installation von RT kann dies eventuell Probleme mit den Zugriffsrechten auf bestimmte Bereiche verursachen. Diese Tatsache sollte beachtet werden. Die Einstellung ist aber notwendig, es müsste im Zweifelsfall also die Rechteverwaltung der bestehenden Installation angepasst werden.

Für alle Benutzer, die die Web-Oberfläche der SIEM-GUI nutzen sollen, müssen die Custom Fields „SIEM“ und „SIEMADMIN“ entsprechend ihrer Berechtigungen eingestellt werden. Das Feld „SIEM“ muss auf 1 stehen, da der Login an der GUI sonst fehlschlägt. Für das Feld „SIEMADMIN“ können die Einstellungen je nach Benutzer vorgenommen werden. Auch wenn die GUI einen Adminbereich für die Benutzerrechte bietet, können diese nur aus RT heraus geändert werden. Die RT REST API erlaubt das Ändern von Custom Fields auf Benutzern nicht, daher ist der entsprechende Administrationsbereich ohne Funktion.

4.3.7.5 Queues (Bereiche)

Für die Tickets, die vom SIEM-System generiert werden, muss eine eigene Queue angelegt werden. Diese sollte den Namen „SIEM“ tragen und als Subject Tag/Betreffskennzeichnung den String „[SIEM]“ verwenden. Abweichende Werte sind möglich, der Name muss in diesem Fall jedoch in der siem-gui.properties angepasst werden.

Im Tab „Group Rights/Gruppenrechte“ müssen die folgenden Berechtigungen für die oben erstellte Gruppe „SIEM Users“ eingestellt werden.

a. General rights / Allgemeine Rechte: alle
b. Rights for Administrators/Rechte für Administratoren: keine

Berechtigung (englisch)	Berechtigung (deutsch)
Modify custom field values	Werte für benutzerdefiniertes Feld anpassen
Modify ticket owner on owned tickets	Modify ticket owner on owned tickets
Modify tickets	Anfragen ändern
Own tickets	Anfragen besitzen
Steal tickets	Anfragen stehlen
Take tickets	Anfragen übernehmen
View exact outgoing email messages and their recipients	Rohform der ausgehenden E-Mails mit ihren Empfängern anzeigen
View ticket private commentary	Vertrauliche Anfrage-Kommentare sehen

Tabelle 7: Benutzergruppe SIEM, Rights for Staff / Rechte für Bearbeiter

Alle anderen Gruppen, auch die System-Gruppen und Rollengruppen, sollten aus Sicherheitsgründen keinerlei Berechtigungen auf dieser Queue haben. Dies gilt ebenso für das Tab „User Rights/Benutzerrechte“, in dem keinerlei Berechtigungen vergeben werden sollten.

4.3.7.6 Globale Einstellungen

In den globalen Einstellungen zu Gruppenrechten, zu finden unter „Admin → Global → Group Rights“ bzw. „Admin → Global → Gruppenrechte“, kann für die oben erstellte Gruppe „SIEM Users“ noch die folgende Berechtigung gesetzt werden (optional):

Berechtigung (englisch)	Berechtigung (deutsch)
Create, modify and delete users	Benutzer erstellen, ändern und löschen

Tabelle 8: Global, Rights for Administrators / Rechte für Administratoren

Diese Einstellung ist ein Sicherheitsrisiko, allerdings lassen sich ohne sie keine Informationen zu fremden Benutzern über die RT REST API abrufen und damit kann der Adminbereich für Benutzer nicht verwendet werden. Da dieser allerdings bei RT ohnehin ohne Funktion ist, kann auf diese Einstellung verzichtet werden. Die übrigen Funktionen der SIEM-GUI sind davon nicht betroffen.

4.3.8 Vorfalldatenbank

Für die Vorfalldatenbank muss im installierten DBMS eine Datenbank angelegt werden. Dazu wird ein Benutzer benötigt, über den die GPM-Engine und die SIEM-GUI auf die Datenbank zugreifen. Dieser benötigt natürlich Zugriff von den entsprechenden Hosts aus und Berechtigungen zum Erstellen, Manipulieren und Auslesen von Tabellen und Datensätzen auf der zuvor erstellten Datenbank. Im Folgenden wird davon ausgegangen, dass die Datenbank den Namen *simu* hat und der Benutzer *simu_user* heißt.

4.3.9 SIEM-GUI

Für die Konfiguration der SIEM-GUI müssen im Wesentlichen zwei Konfigurationsdateien bearbeitet werden. Diese enthalten die Einstellungen für die Verbindung zu RT und zur Vorfalldatenbank. Es ist empfehlenswert die Einstellungen vor dem Kompilieren zu ändern, da so die korrekten Dateien in das WAR-File gepackt werden. Es ist aber auch möglich, die Dateien im WAR-File anzupassen. Dies muss jedoch vor dem Deployment auf dem Apache Tomcat geschehen.

4.3.9.1 Allgemeine Konfiguration

Die Datei *siem-gui.properties* enthält die allgemeine Konfiguration der SIEM-GUI, sowie die Verbindungsinformationen für RT. Wenn RT nach der obigen Anleitung eingerichtet wurde, können die Einstellungen wie folgt übernommen werden:

```
### RT integration
# Base URI of RT
rt.baseuri=http://127.0.0.1/

# Username and password to use for system access
rt.integration.systemusername=siemsystem
rt.integration.systempassword=YOUR_SIEMSYS_PASSWORD

# Custom field name for SIEM access property
rt.integration.cf-access=SIEM
rt.integration.cf-admin=SIEMADMIN

# Custom field names for incident ID and risk on tickets
rt.integration.cf-incident=Incident
rt.integration.cf-risk=Risk

# Ticket queue name for SIEM tickets
rt.integration.siem-queue=SIEM
```

Entsprechende Anpassungen bei Abweichungen von der empfohlenen Konfiguration sind zu berücksichtigen. Außerdem muss das gewählte Passwort für den Systembenutzer eingetragen werden.

4.3.9.2 Verbindung zur Vorfalldatenbank

Die Verbindungsdaten zur für die Vorfalldatenbank sind in der Datei *correlation-db-connector.properties* enthalten. Auch hier muss das Passwort für den Datenbankbenutzer eingetragen und eventuelle Abweichungen von der empfohlenen Konfiguration beachtet werden.

```
### JDBC connection for correlation database
correlation.jdbc.driver = com.mysql.jdbc.Driver
correlation.jdbc.url = jdbc:mysql://127.0.0.1:3306/simu
correlation.jdbc.user = simu_user
correlation.jdbc.pass = YOUR_SIEMUSER_PASSWORD

correlation.vendoradapter.generateddl = false
correlation.vendoradapter.showsql = false
correlation.vendoradapter.database = mysql

correlation.hibernate.globally_quoted_identifiers=false
correlation.hibernate.hbm2ddl.auto=update
```

4.4 Demonstrator-Testergebnisse

4.4.1 MAP-Server irond

Irond implementiert die IF-MAP Spezifikation in der aktuellen Version 2.2 und ist damit ideal geeignet, um als MAP-Server in der SIMU-Umgebung eingesetzt zu werden. Hierbei werden sämtliche neue Features, wie beispielsweise MAP-Content-Authorization, unterstützt. Diese werden zwar aktuell nicht im Projekt genutzt, aber somit bietet *irond* auch die Möglichkeit, bei einer Weiterentwicklung der SIMU-Komponenten eingesetzt zu werden. Anpassungen im Hinblick auf die CBOR/CDDL-Entwicklung fanden bisher am MAP-Server noch nicht statt.

Im Hinblick auf Performance und Skalierbarkeit kann *irond* ebenfalls bedenkenlos eingesetzt werden, da die Menge der Daten, die in einem KMU erzeugt werden, problemlos bewältigt wird. Dies wurde bereits im Einsatz von *irond* in den Vorgängerprojekten gezeigt und an der Verarbeitungsweise hat sich durch das SIMU-Projekt nichts verändert.

Im Testbed unterstützt der MAP-Server sowohl die einfache Benutzer-Authentifizierung als auch Zertifikat-basierte Authentifizierung.

4.4.2 GPM-Engine irongpm

Die Umsetzung der Ideen zur GPM befindet sich noch in einem höchst experimentellen Zustand. Die grundsätzliche und korrekte Funktionsweise konnte allerdings gezeigt und durch Tests abgedeckt werden. Einzig die Funktionalität für ein Muster mit mehreren Metadaten direkt an einem Identifier wirft noch größere Probleme in der Auswertelogik auf. Mehrere Kanten werden hingegen korrekt behandelt.

Für den SIMU-Use-Case im Testbed ist mit der definierten Regel ein korrektes Erkennen eines Angriffes auf eine vorher gefundene Schwachstelle möglich.

Ebenso funktioniert die Anbindung an den VisITMeta-Dataservice zum Abrufen der Daten sowie das Veröffentlichen der Daten per IF-MAP und parallel in die Incident-Datenbank, die von der SIMU-GUI zur Verwaltung und Anzeige der erkannten Vorfälle verwendet wird.

Die Performance und Skalierbarkeit der GPM wurde durch eine parallele Auswertung nur der potentiell betroffenen Regeln zumindest gegenüber der ersten Implementierung des Algorithmus verbessert, ist aber bei weitem noch nicht perfekt. Weitere Maßnahmen zur Verbesserung der Performance, wie beispielsweise Pruning, wurden in der Umsetzung nicht weiter verfolgt.

Die Regelsprache und Einbindung in die GPM-Engine ist prototypisch; Regeln müssen als Implementierung einer Schnittstellenklasse umgesetzt werden, können dafür aber extern entwickelt und zum Programmstart als jar-Archiv eingelesen werden.

Die starke Bindung zu den SIMU-Komponenten, insbesondere zu der SIEM-GUI, wodurch die Abhängigkeiten nicht mehr auf der Ebene der eingelesenen Regeln und Aktionen, sondern global gelten, ist für den Usecase im Testbed kein Problem, müsste aber für eine zukünftige Nutzung ohne SIMU-Bezug aufgebrochen werden - momentan lässt sich die GPM nicht losgelöst von SIMU einsetzen.

Die für den Usecase definierte Regel prüft den Graphen auf attack-detected-Metadaten, welche die gleiche CVE-ID wie eine zuvor erkannte Schwachstelle besitzen. In diesem Fall wird der konkrete Teilgraph, der das Muster der Regel komplett erfüllt hat, als JSON-String zusammen mit weiteren Informationen wie einem Namen per Hibernate in die Incident-Datenbank eingetragen, von wo aus die SIMU-GUI diese Informationen ausliest.

Die Regel definiert dabei ein Muster mit den folgenden Identifiern:

- Service,
- Implementation dieses Services,
- IP-Adresse des Services,
- das Gerät (Device) des Services,
- die IP-Adresse des Angreifers,
- die Identität des Angreifers,
- die MAC-Adresse des Angreifers,
- der Access-Request-Knoten des Angreifers.

Im Testbed verwendet irongpm den VisITMeta-Dataservice als Datenquelle, die Datenbank der SIMU-GUI als Ablageort für erkannte Vorfälle sowie eine Verbindung zum MAP Server für das zusätzliche Veröffentlichen von Metadaten bei Erfüllung einer Regel.

Im Gegensatz zu den ursprünglichen Projektzielen, die eine Anomalie-Erkennung auf Graphebene vorsahen, wurde mit der GPM eine reine signaturbasierte Erkennung umgesetzt. Das bedeutet auch, dass keinerlei Auswerteverfahren eine zeitliche Veränderung des Graphen berücksichtigen. Ein Graph passend zu einem Muster wird somit nur erkannt, wenn alle benötigten Elemente zu einem Zeitpunkt im Graph

vorhanden sind. Außerdem hat die Neuentwicklung dazu geführt, dass keine eigene Regelsprache mehr konzipiert werden konnte. Dadurch konnten letztlich auch darauf aufbauende Fragestellungen nicht tiefer betrachtet werden. Hierzu zählt, im Hinblick auf den Projektantrag, die visuelle Unterstützung einer intuitiven Regelerstellung und -Auswertung sowie das Vorhaben, den Aufwand für Konfiguration und Betrieb zu minimieren. Diese Ziele konnten jeweils nur partiell, bzw. konzeptionell erfüllt werden, beispielsweise durch den Einsatz frei verfügbarer und gängiger Tools zur Datensammlung oder die Integration der VisITMeta-Software zur Analyse eines Vorfalls. Allerdings konnten keine Erweiterungen hinsichtlich der Regelsprache umgesetzt werden, also keine Unterstützung in der Regelerstellung oder eine Erfassung des Normalzustandes.

4.4.3 Sensor: DECOmap Snort

Das Snort-Modul des DECOmap dient beim Demonstrator dazu den Angriff von „m.musterman“ auf „jon.doe“ zu erkennen. Da es schwierig ist, den Netzwerkverkehr so zu simulieren, dass Snort genau eine spezifische CVE-Meldung erzeugt, wurde das Regelwerk auf eine einfache Regel reduziert, die Datenverkehr auf Port 25 erkennt und diese mit der im Szenario festgelegten CVE-2015-1234 meldet. Diese CVE ist genau die, die von OpenVAS auf dem Gerät des Opfers gefunden wurde. Nur so kann die Regel der GPM-Engine ausgelöst werden und ein Vorfall erzeugt werden.

Wie gut die Erkennung von echten CVE-Angriffen in einem überwachten Netzwerk ist, hängt dabei ausschließlich von der Implementation und Konfiguration des Snort IDS ab. DECOmap liest lediglich die generierten Meldungen aus. Daher lassen sich über die Zuverlässigkeit des Sensors nur begrenzt Aussagen treffen. Wie die Dummy-Regel für den Demonstrator zeigt, kann eine fehlerhafte Konfiguration zu Unmengen von False-Positives führen, die das System belasten und im Zweifelsfall ad absurdum führen.

Ein Problem beim verwendeten Modul ist, dass ausschließlich CVE-Meldungen aufgenommen und im MAP-Server als SIMU-spezifisches „attack-detected“ Metadatum abgelegt werden. Alle weiteren Meldungen werden ignoriert. Dadurch wird die Last auf dem MAP-Server zwar gegenüber der ursprünglichen Implementation mit „event“ Metadaten massiv verringert, es gehen aber auch sehr viele Informationen verloren. An dieser Stelle müsste weiter geforscht werden, um eine zufriedenstellende Lösung zu finden.

4.4.4 Sensor: nmap

Ironnmap, als nmap-Wrapper, läuft als Dienst auf einer Maschine im Testbed. Das heißt, ironnmap läuft kontinuierlich und nutzt eine IF-MAP Subscription, um auf neue Informationen zu reagieren.

Für den SIMU-Usecase wird das macmon-Device abonniert und dort auf request-for-investigation-Metadaten zwischen dem Device und einer IP- oder MAC-Adresse gewartet. Diese werden von macmon für jedes neu entdeckte Endgerät veröffentlicht.

Daraufhin startet ironnmap einen Scan gemäß der vorgegebenen Konfiguration für das Endgerät. Die Ergebnisse werden ausgewertet und entsprechend des

Metadatenschemas veröffentlicht. Dazu gehören Service-, Implementation- und IP-Address-Identifier mit service-ip, service-implementation, hop-count sowie discovered-by Metadaten.

Durch nmap bedingt werden die Informationen über das Betriebssystem eines Gerätes oder die dort laufenden Dienste mit einer bestimmten Wahrscheinlichkeit versehen. Gerade bei der Betriebssystemerkennung treten dabei oftmals stark unterschiedliche Erkenntnisse auf. So werden Apple Computer häufig sowohl als solche mit OSX-Betriebssystemen, aber auch als iOS-Geräte erkannt. Vermutlich liegt das am in beiden Betriebssystemen verwendeten Netzwerkstack.

Für die Implementierung von ironnmap wurde daher entschieden, das Ergebnis mit der höchsten Wahrscheinlichkeit zu verwenden. Sind mehrere Ergebnisse mit gleicher Wahrscheinlichkeit vorhanden, wird das erste aus der Liste von nmap verwendet.

Diese Informationen sind also mit Vorsicht zu verwenden.

4.4.5 Sensor: OpenVAS

Ironvas, als OpenVAS-Wrapper, läuft als IF-MAP-Client auf einer weiteren virtuellen Maschine im Testbed und ist genauso wie ironnmap als Dienst konfiguriert. Mithilfe der gleichen Subscription auf das macmon-Device und request-for-investigation-Metadaten werden hier statt nmap-Portscans die Ergebnisse des Schwachstellenscanners OpenVAS veröffentlicht. Dazu gehören sowohl die Targets (Ziele anhand ihrer IP-Adresse) als auch direkt dazu passende Tasks (Aufgaben, die auf einem Target und einer Scan-Config basieren; diese wurde vorher angelegt und deckt die allermeisten vorhandenen Tests von OpenVAS ab). So wird beim Bekanntwerden eines neuen Geräts ein OpenVAS-Scan für dieses Gerät gestartet und die Informationen in IF-MAP veröffentlicht.

Welche Teile eines Reports am Ende als Schwachstellen in IF-MAP veröffentlicht werden, kann über Filter in Form einer von extern eingelesenen Datei mit JavaScript-Code gesteuert werden. Hierzu kann auf alle Informationen einer gefundenen Schwachstelle zugegriffen und somit z.B. der Threatlevel oder der Name verwendet werden.

Grundsätzlich unterstützt ironvas die Verarbeitung von erneute Reports, also weitere Scans auf einem vorhandenen Gerät, und daraufhin ggf. veränderte Schwachstellen zu veröffentlichen und nicht weiter gültige wieder zu entfernen.

Bei der Anbindung im Testbed ergaben sich allerdings Probleme, was das Mapping der gefundenen OpenVAS-Informationen auf IF-MAP Daten bzw. das SIMU-Schema angeht.

Aus dem OpenVAS-Report können Informationen wie der Typ CVE und die dazugehörige CVE-ID ausgelesen werden, was zum Erstellen des Vulnerability-Identifiers genügt. Um die Verbindung zu einem erkannten Service und der dazugehörenden Implementation per „implementation-vulnerability“-Linkmetadatum zu erstellen, sind allerdings nicht ausreichend Informationen vorhanden.

Die Information über den Namen der Implementierung eines Dienstes und die Version, die von einer bestimmten CVE betroffen ist, wird in einem OpenVAS-Report nur in einem Freitext beschrieben. Beispiele wären: “Affected Software/OS: Wireshark version 0.9.6 to 1.0.6 on Linux” oder “Versions prior to OpenSSH 5.2 are vulnerable. Various versions of SSH Tectia are also affected.“

Hier kann der Dienstname Wireshark bzw. OpenSSH nicht zuverlässig extrahiert werden. Ebenso ist es problematisch, dass an dieser Stelle auch Versionsbereiche angegeben werden können, d.h. der MAP-Graph müsste nach allen - nicht bekannten - Versionen (im Beispiel zwischen 0.9.6 und 1.0.6) der Implementation-Identifier durchsucht werden. Beim zweiten Beispiel mit der Angabe „Versions prior to 5.2“ gestaltet sich dieses noch schwerer.

Für die Testumgebung wurde daher entschieden, das Standard-Verhalten von ironvas zu verwenden, welches den Report als event-Metadatum an die IP-Adresse des gescannten Gerätes veröffentlicht. In Fall von SIMU ist dieses dann die IP-Adresse des Services.

Die GPM kann mit einer angepassten Regel, die statt dem Vulnerability-Identifier nun diese event-Metadaten auf ihre CVE überprüft, weiterhin die Verbindung zu einem „attack-detected“-Metadatum herstellen und so einen Angriff auf eine erkannte Schwachstelle eines Gerätes erkennen.

Die Events werden von ironvas hier im Widerspruch zur Spezifikation per „publish update“ und nicht per „publish notify“ veröffentlicht. Dies liegt darin begründet, dass die GPM in der kurzen Zeit nicht mehr an die speziellen REST-Aufrufe des VisITMeta-Dataservice zum Abrufen von „notify“-Daten angepasst werden konnte. Ansonsten wäre dies grundsätzlich auch möglich.

Ebenso ist an dieser Stelle aber auch der Widerspruch zwischen einerseits als flüchtig definierten „event“-Metadaten und der eher dauerhaften Gültigkeit einer Schwachstelle zu sehen. Dies war nicht zuletzt einer der Gründe für die „implementation-vulnerability“-Idee im SIMU-Schema.

Prototypisch für die Testumgebung und zur Demonstration des Gesamtprojektes wurde sich daher für die Verwendung von „publish update“ und „event“-Metadaten entschieden.

Für die Zukunft, auch in der Zeit nach dem SIMU-Projekt, soll auf Seiten der Hochschule Hannover im Rahmen allgemeiner Entwicklungsarbeiten oder durch studentische Arbeiten (Hilfskräfte/Abschlussarbeiten) eine zusätzliche Mapping-Komponente entwickelt werden, welche gefundene Implementierungen, bspw. durch ironnmap, anhand eines Lookups in einer öffentlich zugänglichen Datenbank (wie der National Vulnerability Database – NVD) zu einer oder mehreren Implementierungen zuordnen kann. Anschließend kann dann der Implementation-Vulnerability-Subgraph erstellt werden und die für SIMU angedachte Funktionsweise erfüllt werden.

Für die weiteren Komponenten in SIMU und die Verarbeitung durch die GPM, sowie die Bearbeitung und Analyse mithilfe der SIMU-GUI ändert sich durch die abgewandelte Funktionalität des OpenVAS-Clients und den veröffentlichten Informationen nichts.

4.4.6 Sensor: macmon NAC

Der macmon-NAC Dienst wird im Testbed auf einer eigenen virtuellen Maschine ausgeführt. Auf der VM ist als Betriebssystem ein Debian 8.x mit aktuellem Java (openjdk-8-jre Version 8 Update 66) installiert. Der NAC-Service wird mit Hilfe eines Debian-Paketes installiert und läuft als Dienst ‚*nac-engined*' nach der Installation im Hintergrund.

Der NAC-Service im Testbed läuft hierbei ausschließlich als Simulator, da kein echtes physikalisches Netzwerk zum Scannen vorhanden ist. Um Daten zu publizieren/simulieren, wird die in AP4 beschriebene REST-API verwendet. Mit Hilfe der API können Metadaten zu aktiven und autorisierten Endgeräte im internen Netzwerk veröffentlich werden. Zusätzlich zu den Metadaten zur Autorisierung können Metadaten zu beliebigen erkannten Port-/Service-Informationen veröffentlicht werden. Zur Erkennung neuer Endgeräte durch andere Sensoren wie ironnmap wird die IP-/MAC-Adresse des Endgerätes außerdem mit einem ‚*request-for-investigation*' publiziert.

Im Demonstrator wird der macmon-MAPC zur Veröffentlichung der Metadaten zu den Endgeräten des Opfers und des Angreifers verwendet. Die virtuellen Maschinen des Angreifers und des Opfers werden dabei so simuliert, als wenn die beiden Endgeräte von macmon an einem physikalischen Netzwerkverteiler erkannt werden würden. Beim Opfer wird zusätzlich ein Service in einer Version publiziert, welche eine bekannte CVE-Schwachstelle hat. Der ausgeführte Angriff auf diesen verwundbaren Service wird letztlich von der GPM erkannt und für das durch macmon publizierte Endgerät wird ein ‚*unexpected-behavior*' Metadatum publiziert.

Das NAC-System registriert zusätzlich zu den publizierten Sensor-Daten pro Endgerät einen Subscribe auf das ‚*unexpected-behavior*' Metadatum. Im Falle eines echten physikalischen Netzwerks würde das Endgerät des Angreifers von macmon aus dem Netzwerk ausgesperrt werden können. Im Testbed mit virtuellen Maschinen ist ein reales Enforcement nicht möglich und als Reaktion wird das erkannte unerwartete Verhalten ausschließlich geloggt.

4.4.7 Sensor: NCP VPN

Der NCP Secure Client / NCP Secure Server ist ein VPN-System und hat im Demonstrator die Aufgaben, den Netzzugang zu ermöglichen und zu kontrollieren. In Abbildung 15 sind die Aufgaben und der Informationsfluss schematisch dargestellt.

Das NCP VPN-System unterstützt das Gesamtsystem außerdem als „Sensor", d. h. vom VPN-Client können Informationen über das Endgerät gesammelt und an die Zentrale übermittelt werden (1). Dort werden die Informationen zur weiteren Verarbeitung in den MAP-Server eingetragen (2) (siehe AP4). Am MAP-Server werden diese mit anderen Daten (3) verknüpft und durch die GPM-Detection-Engine auf mögliche Angriffsszenarios oder Schwachstellen hin untersucht (4). Die Detection-Engine kann dann ein IF-MAP-Event an den NCP-Server senden (5), so dass dieser den Netzzugang beschränkt oder weiter öffnet.

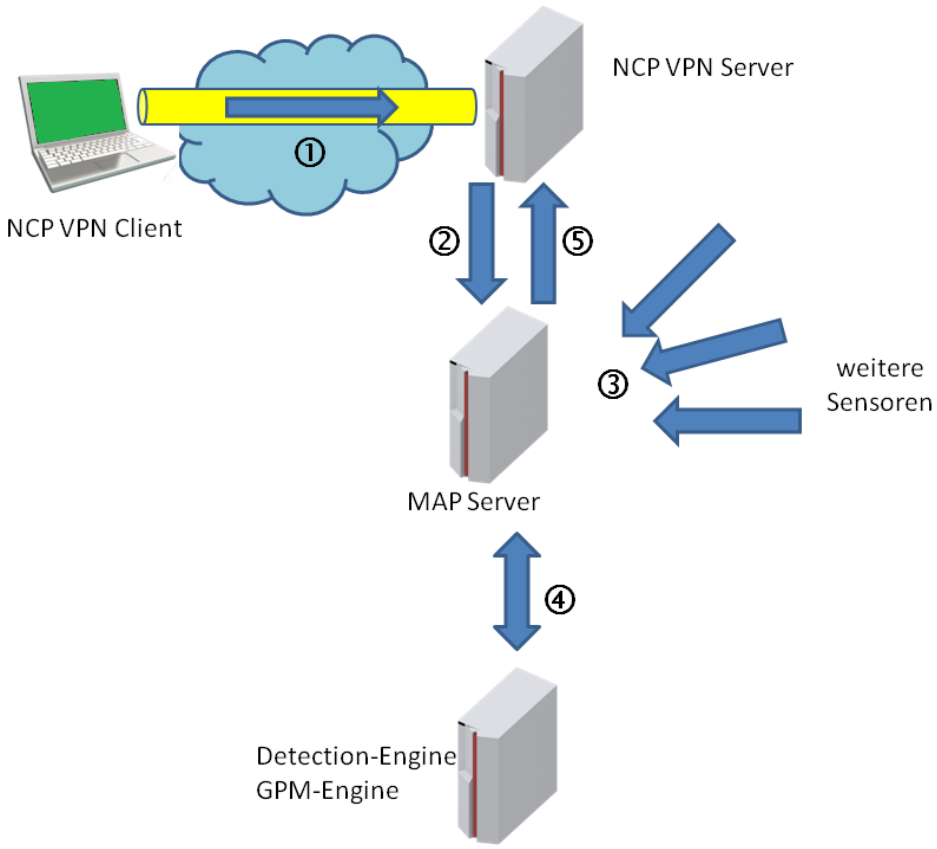

Abbildung 15: Informationsfluss zu und von den NCP-Komponenten

Die Daten, welche der Client sammelt, sind in der Praxis vor allem Informationen über am Client-System installierte Software und deren Version und über Betriebssystem-Hotfixes. Wie in AP4 dargelegt, werden die Daten über die WMI-Schnittstelle vom Windows-System abgefragt. Wie sich im Laufe des Projekts festgestellt hat, hat dieser Weg einige Nachteile:

- WMI-Abfragen sind relativ langsam.
- Es können nur Informationen zu Programmen abgefragt werden, die mit Hilfe des Microsoft Installers (MSI) installiert wurde.
- Die Liste der ausgegebenen Hotfixes ist nicht vollständig.

Um diese Schwächen zu umgehen, ist zukünftig die Verwendung alternativer Schnittstellen angedacht:

- Zur Abfrage der installierten Programme kann ein Registry-Schlüssel genutzt werden, der die gleiche Information enthält, die in der Systemsteuerung zur Anzeige der de-installierbaren Software verwendet wird.

Zur Abfrage der installierten Hotfixes kann die WSUS-API („Windows Software Update System“) genutzt werden.

4.4.8 Visualisierung: VisITMeta

VisITMeta erfüllt entsprechend seiner Architektur zwei Rollen in der SIMU-Umgebung. Zum einen wird der Dataservice verwendet, um auf die Zustände des MAP-Graphen zuzugreifen. Zum anderen können die beiden Visualisierungskomponenten (JavaScript und Java-GUI) genutzt werden, um den Netzzustand nachzuvollziehen.

Der Dataservice bietet dabei die Möglichkeit, Informationen über den MAP-Server über mehrere parallele Subscriptions zu erfassen. Im Testbed wird dies verwendet, um zu Beginn das macmon-Device, den NCP VPN-Server sowie den NMAP-Scanner über jeweils eine eigene Subscription zu abonnieren. So können von Anfang an alle Informationen gesammelt werden. Wenn im weiteren Verlauf des Usecases die Informationen verbunden werden und so bei weiteren Updates jeweils die gleichen Daten über mehrere Subscriptions erfasst werden, wird dies vom Dataservice weiterhin korrekt verarbeitet.

Der Dataservice ist zudem so konfiguriert, das auf seine REST-Schnittstelle auch von anderen IP-Adressen zugegriffen werden kann. Damit kann zum einen die VisITMeta-Visualisierungskomponente auf Java-Basis verwendet werden, um den Zustand des MAP-Graphen zu analysieren. Zum anderen greift auf diese Weise auch die GPM auf die Informationen des Dataservice zu, um darauf ihre Auswertungen durchzuführen. Ebenso kann die JavaScript-Visualisierung, die in die SIMU-GUI integriert ist, auf diesem Wege den Stand des Graphen abfragen.

4.4.9 Visualisierung: SIEM-GUI

Durch den dezentralen Aufbau des Systems, das heißt die Verwendung von bestehenden externen Diensten, ließ sich eine Menge Implementationsaufwand einsparen. Daher ist dieser Ansatz grundsätzlich zu begrüßen. Bestehende Dienste, wie zum Beispiel ein Ticketsystem zu nutzen, bedeutet auch, dass auf bereits breitflächig eingesetzte und getestete Logik zurückgegriffen werden kann. Das Fehlerpotential ist dadurch in diesen Bereichen geringer als bei einer eigenen Implementation. Zudem ist die verwendete Software in der Regel auf ihren Zweck hin optimiert und daher leistungsfähiger als eine in die SIEM-GUI eingebettete Logik, die als Teil der SIEM-GUI nicht in vollem Maße auf diese Aufgabe optimiert werden kann.

Dabei das Ticketsystem zur Benutzer-Authentifikation zu verwenden, ist ein weiterer Punkt, an dem Aufwand eingespart werden kann. Zwar ist die Integration der REST API (oder einer vergleichbaren Schnittstelle) eines Ticketsystems in Spring Security vergleichsweise aufwändig und erfordert einiges an Einarbeitung in die internen Abläufe von Spring Security. Allerdings lässt sich dadurch eine fehleranfällige Synchronisation zwischen SIEM-GUI-Benutzern und Ticketsystem-Benutzern vermeiden. Wären die Systeme getrennt, so müsste immer sichergestellt sein, dass jedem SIEM-GUI-Benutzer A eine Ticketsystem-Benutzer A zugewiesen ist. Sonst könnte das Ticketsystem nicht verwendet werden. Dieses Vorgehen ist jedoch auf anfällig für Fehler bei der Verknüpfung der Konten und es ist eine redundante Authentifikation notwendig, da eine Session der SIEM-GUI nicht für das Ticketsystem gilt.

Weiterhin als positiv zu bewerten sind die direkten und leichtgewichtigen Kommunikationswege zwischen SIEM-GUI und der GPM-Engine. Da zwischen Engine und SIEM-GUI eine Datenbank geschaltet ist, können beide Komponenten unabhängig voneinander arbeiten. Die SIEM-GUI kann mit Ergebnissen arbeiten, die die GPM-Engine in der Vergangenheit erkannt hat. Umgekehrt ist die GPM-Engine auch nicht auf eine laufende SIEM-GUI Instanz als Empfänger angewiesen. Dennoch ist dieser Weg durch den direkten Datenbankzugriff kurz gehalten und ohne den Overhead einer zusätzlichen API realisiert worden.

Allerdings gibt es auch negative Aspekte der Architektur, die die Implementation teilweise schwer oder in bestimmten Bereichen sogar unmöglich gemacht haben.

Insbesondere ist hier die Wahl von Request Tracker (RT) als Ticketsystem zu nennen. Zwar ist es als reines Ticketsystem im Vergleich zu größeren Systemen wie Redmine eher leichtgewichtig. Allerdings bringt es einige Unzulänglichkeiten mit sich, die die Verwendung durch externe Programme sehr erschweren.

Zum einen ist die REST API von RT sehr unvollständig, das heißt, es können nicht alle Typen von Objekten (Tickets, Queues etc.) innerhalb von RT ausgelesen werden. Als Datenformat für die Rückgabe wird kein strukturiertes Format wie JSON, XML oder YAML verwendet, sondern eine reine Plaintext-Ausgabe mit Key:Value Zuweisungen. Diese ähnelt dem Format eines HTTP-Headers. Bei Listen werden die einzelnen Elemente durch Trennsymbole voneinander abgegrenzt. Die Verwendung eines strukturierten Datenformats hätte gerade diesen Punkt massiv vereinfacht, insbesondere bei der Auswertung.

Weiterhin ist das Format, das die einzelnen Endpoints der API verwenden, nicht konsistent. Je nach Endpoint unterscheiden sich einzelne Aspekte. Insbesondere bei der Darstellung von Custom Fields gibt es zwei verschiedene Formate, die scheinbar völlig willkürlich eingesetzt werden. Außerdem ist die Einrückung von Multiline-Texten je nach Feld unterschiedlich, was zwar für Menschen schön zu lesen ist, die programmatische Auswertung allerdings unnötig kompliziert macht. Die mangelhafte Dokumentation der REST API seitens RT macht die Arbeit mit diesem Format nicht einfacher. Eigene Nachforschungen über den Funktionsumfang und die verwendeten Formate im Source-Code von RT scheitert an der extrem komplizierten Perl-Programmstruktur mit hunderten winziger Module.

Auch die Rückgaben im Fehlerfall sind bei der REST API alles andere als konsistent. Die meisten Endpoints liefern Fehlermeldungen in der Form aus, dass der Statuscode der Anfrage angegeben wird und darunter eine Nachricht, was fehlgeschlagen ist.

```
RT/4.2.3 200 Ok


# Ticket 1024 does not exist.
```

Allerdings gibt es einige Operationen, bei denen zwar ein Statuscode angegeben wird, der nicht 200 ist, jedoch darunter eine normale Ausgabe wie bei einem erfolgreichen Aufruf ausgegeben wird. Exemplarisch dafür ist der Endpoint zum Bearbeiten eines Benutzers. Dieser scheint nicht mit Custom Fields, egal in welchem der zwei Formate,

umgehen zu können. Dies wird auch mit einem Syntax Error und dem entsprechenden Fehlercode quittiert, allerdings fehlt die Fehlernachricht und stattdessen wird, wie bei einer erfolgreichen Operation, das Benutzerobjekt ausgegeben. Das macht insbesondere das Error-Handling sehr kompliziert und muss für jede einzelne Operation gesondert entwickelt werden.

Außerdem fehlen einige wichtige Funktionen. So können, wie bereits erwähnt, Custom Fields auf Benutzern nicht bearbeitet werden. Der Versuch, dies zu tun, wird mit einem Syntax Error quittiert. Ob dies bei anderen Objekten möglich ist, wurde nicht getestet, das Erstellen von Tickets mit Custom Fields funktioniert allerdings tadellos. Zudem kann über die REST API nicht das Gruppen- und Rechtesystem von RT zugegriffen werden, weder lesend noch schreibend. Dadurch lässt sich dieses nicht für eigene Zwecke nutzen und müsste redundant in der SIEM-GUI nachgebildet werden. Das wäre allerdings sehr anfällig für Fehler und schwer zu pflegen, da die Synchronisation manuell durchgeführt werden müsste.

Das CLI Tool von RT ist identisch mit der REST API und bietet daher auch keine weiteren Möglichkeiten. Im Prinzip tut das CLI Tool nichts anderes als selbst die REST API aufzurufen.

4.4.10 Weitere Komponenten

Es gibt eine Vielzahl von weiteren Komponenten, die in dem SIMU-Projekt entwickelt worden sind, aber im Demonstrator nicht zum Einsatz kommen. Daher werden diese Tests hier auch an dieser Stelle nicht mehr erwähnt. Dies betrifft weitere DECOmap-Module der DECOIT GmbH (iptables, LDAP, RADIUS, Nagios/Icinga, OpenVPN, Android, File Integrity Monitor), das IO-Tool und das CBOR-Protokoll von Fraunhofer SIT sowie weitere Log-Kollektoren von der Hochschule Hannover (irongenlog und ironsyslog).

5 Zusammenfassung

Im beschriebenen Projekt wurde eine neue SIEM-Architektur entwickelt, die auf Basis des IF-MAP-Protokolls der Trusted Computing Group (TCG) in der Lage ist, verschiedene Log-Informationen und Events diverser Sicherheitskomponenten zu konsolidieren und auszuwerten. Dazu wurde ein Metadaten-Schema für ein homogenes Ereignistransportprotokoll ausgearbeitet. Durch die Visualisierung der Kommunikationsbeziehungen über VisITMeta lassen sich unerwünschte Zustände herausfiltern und getrennt von den anderen Daten analysieren. Zudem ist die Graph-Pattern-Matching (GPM) Engine in der Lage, Angriffsvektoren auszumachen und in nahezu Echtzeit darauf zu reagieren. Hier wird zum einen die Weiterleitung erkannter Vorfälle gemäß den definierten Sicherheitsprozessen unterstützt, die bspw. die Verantwortlichkeiten zur Reaktion dem Administrator überlässt. Es besteht aber auch die Möglichkeit einer automatisierten Ereignisbehandlung durch das Erzeugen weiterer spezifischer Metadaten.

Die wesentlichen Vorteile der GPM-Engine bestehen zum einen darin, dass alle im Graphen enthaltenen Informationen der diversen Kollektoren und Flow Controller in die Erkennung von Sicherheitsvorfällen einbezogen werden können, da diese in einem einheitlichen Datenformat in den Graphen geladen werden. Dies ermöglicht eine deutlich bessere und komplexere Detektion von Sicherheitsvorfällen. Zum anderen können neben den aktuellen Metadaten auch beliebige historische Zustände, die vom Data-Service bereitgestellt werden, in die Erkennung der Sicherheitsvorfälle mit einbezogen werden. Zusätzlich besteht durch die flexible Definition der Erkennung von Vorfällen auf Basis des Konzepts der Listener eine sehr flexible und leicht erweiterbare Möglichkeit der Definition relevanter Ereignisse, die zudem durch die Benachrichtigungen nahezu in Echtzeit erkannt werden können.

Durch die übersichtliche Oberfläche der SIMU-GUI können Vorfälle, die von der GPM-Engine erkannt werden, einheitlich dargestellt werden. Bei neuen Vorfällen wird das Ticketsystem kontaktiert, so dass auch eine Bearbeitung der Vorfälle schnell möglich und gut dokumentiert ist. Durch die Konsolidierung weiterer sicherheitsrelevanter Daten in der GUI kann eine Art Gesamt-IT-Bedrohungsstufe eines Unternehmens anhand der Risikobewertung ermittelt und mit entsprechenden Farbbalken dargestellt werden. Eine strukturierte und priorisierte Bearbeitung von Sicherheitsvorfällen wird so ermöglicht.

Der im SIMU-Projekt entwickelte Demonstrator zeigt beispielhaft, dass es möglich ist, ein SIEM-artiges System auf IF-MAP-Basis aufzubauen. Durch den Einsatz von Erweiterungen des IF-MAP-Standards und Werkzeugen wie VisITMeta ist es theoretisch zusätzlich machbar, auch zeitliche Verläufe zu analysieren und so eine zusätzliche Ebene an Informationen für die Erkennung von Vorfällen zur Verfügung zu stellen.

Durch die Speicherung der gesammelten Informationen in einem Graphen, ist die Regelerstellung und spätere Auswertung der Regeltreffer intuitiver, als bei einer Datensatz-basierten Speicherung, wie sie zum Beispiel bei relationalen Datenbanken zum Einsatz kommt.

Eine große Schwachstelle des entwickelten Demonstrators ist jedoch, dass aus Performance-Gründen viele Meldungen, z.B. vom Snort-Sensor, ignoriert werden müssen. An dieser Stelle sollte in zukünftigen Projekten weiter geforscht werden, um eine zuverlässige und leistungsfähige Verarbeitung von Ereignisdaten im Big-Data-Umfeld gewährleisten zu können. Auch Herstellerlösungen kranken an diesem Problem, wie Tests innerhalb des Projektes gezeigt haben. Denn erst die Verknüpfung der strukturellen Informationen, die im MAP-Server gespeichert werden, mit den detaillierten Ereignisdaten von Sensoren, wie z.B. Snort, erlauben eine präzise Ermittlung von Angriffen auf das überwachte Netzwerk.

Das SIMU-Projekt ist ursprünglich angetreten, um SIEM-Systeme auch in kleineren Umgebungen zu ermöglichen und einsetzen zu können. Gerade Klein- und Mittelständische Unternehmen (KMU) sollten davon profitieren, da sie keine eigenen IT-Sicherheitsexperten vorhalten können. Das SIMU-Projekt hatte sich daher vorgenommen, die folgenden Leistungsmerkmale zu unterstützen:

a. **Leichte Integrierbarkeit in IT-Infrastrukturen von KMU:** Durch die Verwendung von weit verbreiteten Standards bei Kommunikationsprotokollen und Datenformaten, die in typischen Netzkomponenten bereits implementiert sind, sollte der Aufwand für Installation, Konfiguration und Wartung von zusätzlichen SIMU-Komponenten minimiert werden.
b. **Einfache Nachvollziehbarkeit von relevanten Ereignissen und Vorgängen im Netz:** Relevante Ereignisse und Vorgänge im Netz sollten leicht verständlich visualisiert werden. Dadurch wird das Verständnis und die Nachvollziehbarkeit von Vorgängen im Netz erleichtert und damit die Sicherheit gestärkt.
c. **Geringer Aufwand für Konfiguration, Betrieb und Wartung:** Das SIMU-System sollte mit standardisierten Vorkonfigurationen arbeiten und die Möglichkeit bieten, aus der leicht verständlichen Visualisierung des Netzes teilautomatisch Richtlinien und Konfigurationen abzuleiten. Dadurch wird der Aufwand für Konfiguration, Betrieb und Wartung gegenüber klassischen SIEM-Systemen stark reduziert.

Diese Ziele sind allerdings nur zum Teil erreicht worden. So kann der Demonstrator zwar relativ leicht in bestehende IT-Infrastrukturen integriert werden, da er auf offenen Standards basiert. Aber der Wartungsaufwand konnte nicht wirklich verringert werden, da diverse Komponenten in der SIMU-Architektur verankert sind. Ein einheitlicher Prototyp, der „Out-of-the-Box“ installiert wird, konnte daher leider nicht entwickelt werden. Durch die SIEM-GUI ist aber die Nachvollziehbarkeit von Ereignissen übersichtlich umgesetzt worden; die aber zur Analyse der VisITMeta-Darstellung im Graphen wiederum Expertenwissen notwendig macht. Auch die intelligente Anomalie-Erkennung ist nicht komplett, wie ursprünglich angedacht, implementiert worden, um ebenfalls den IT-Administrator zu entlasten und gleichzeitig den Normalzustand eindeutig kennzeichnen zu können.

Trotzdem kann ein sehr positives Fazit zu den Projektergebnissen gezogen werden, da sich alle entwickelten Komponenten unabhängig voneinander weiter nutzen und entwickeln lassen. Zudem wurden die Arbeiten entsprechend dokumentiert und stehen größtenteils unter Github als Open Source Software frei zur Verfügung.

6 Anhang

6.1 CBOR-Draft draft-greevenbosch-appsawg-cbor-cddl-07

Network Working Group — C. Vigano
Internet-Draft — Universitaet Bremen
Intended status: Informational — H. Birkholz
Expires: April 20, 2016 — Fraunhofer SIT
October 18, 2015

CBOR data definition language (CDDL): a notational convention to express CBOR data structures
draft-greevenbosch-appsawg-cbor-cddl-07

Abstract

This document proposes a notational convention to express CBOR data structures (RFC 7049). Its main goal is to provide an easy and unambiguous way to express structures for protocol messages and data formats that use CBOR.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 20, 2016.

6.2 Abbildungsverzeichnis

6.3 Tabellenverzeichnis

6.4 Literaturverweise

[AHH14] Ahlers, Heine, Hellmann, Kleiner, Renners, Rossow, Steuerwald: *Replicable security monitoring: Visualizing time-variant graphs of network metadata.* Joint Proceedings of the Fourth International Workshop on Euler Diagrams (ED 2014) and the First International Workshop on Graph Visualization in Practice (GViP 2014) co-located with Diagrams 2014, Hrsg. Jim Burton, Gem Stapleton u. Karsten Klein, Melbourne (Australien) 2014

[BIRK12] Birkholz, Sieverdingbeck, Sohr, Bormann: *IO: An interconnected asset ontology in support of risk management processes.* IEEE Seventh International Conference on Availability, Reliability and Security, Page 534-541, 2012

[BIVI15] Birkholz, H., Vigano, C.: *CBOR data definition language (CDDL): a notational convention to express CBOR data structures*. IETF Internet-Draft, Intended status: Informational, Expires: April 20, 2016, October 2015

[CHAN09] Varun Chandola, Arindam Banerjee, Vipin Kumar: *Anomaly detection: A survey*. ACM Computing Surveys, 41(3), 2009

[DDB11] Detken, Dunekacke, Bente: *Konsolidierung von Metadaten zur Erhöhung der Unternehmenssicherheit*. D.A.CH Security 2011: Bestandsaufnahme, Konzepte, Anwendungen und Perspektiven, Herausgeber: Peter Schartner, syssec Verlag, ISBN 978-3-00-027488-6, Oldenburg 2011

[DHRR15] Detken, Heine, Rix, Renners: *Event-Korrelation in SIEM-Systemen auf Basis von IF-MAP.* D.A.CH Security 2015: Bestandsaufnahme, Konzepte, Anwendungen und Perspektiven, ISBN 978-3-00-049965-4, Hrsg. Peter Schartner, Kerstin Lemke-Rust, Markus Ullmann, syssec-Verlag, Bonn 2015

[DRS14] Detken, Rossow, Steuerwald: *SIEM-Ansätze zur Erhöhung der IT-Sicherheit auf Basis von IF-MAP*. D.A.CH Security 2014: Bestandsaufnahme, Konzepte, Anwendungen und Perspektiven, ISBN 978-3-00-046463-8, Hrsg. Peter Schartner u. Peter Lipp, syssec-Verlag, Graz (Österreich) 2014

[DSBW12] Detken, Scheuermann, Bente, Westerkamp: *Automatisches Erkennen mobiler Angriffe auf die IT-Infrastruktur*. D.A.CH Security 2012: Bestandsaufnahme, Konzepte, Anwendungen und Perspektiven, Herausgeber: Peter Schartner und Jürgen Taeger, syssec Verlag, ISBN 978-3-00-039221-4, Konstanz 2012

[DSH15] Detken, Scheuermann, Hellmann: *Using Extensible Metadata Definitions to Create a Vendor-Independent SIEM System*. The Sixth International Conference on Swarm Intelligence - The Second BRICS Congress on Computational Intelligence (ICSI-CCI) 2015, Advanced

in Swarm and Computational Intelligence, Proceedings Part II, Editors: Y. Tan, Y. Shi, F. Buarque, A. Gelbukh, S. Das, A. Engelbrecht, ISBN 978-3-319-20471-0, publishing house Springer, 25.-28. June, Beijing International Convention Center (BICC), Beijing (China) 2015

[TCG14] Trusted Computing Group: *TNC IF-MAP Binding for SOAP*. Specification Version 2.2, Revision 9, March 2014

[TCG12b] Trusted Computing Group: *TNC IF-MAP Metadata for Network Security*. Specification Version 1.1, Revision 8, May 2012

6.5 Glossar

AP	Arbeitspaket
API	Application Programming Interface
AR	Access Requestor
BMBF	Bundesministerium für Bildung und Forschung
CBOR	Concise Binary Object Representation
CDDL	Common Development and Distribution License
CLI	Command Line Interface
CPU	Central Processing Unit
CVE	Common Vulnerabilities and Exposures
DHCP	Dynamic Host Configuration Protocol
ENUM	Telephone Number Mapping
GPM	Graph Pattern Matching
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
ICS	Internet Connection Sharing

ID	Identification
IDMEF	Intrusion Detection Message Exchange Format
IDS	Intrusion-Detection-Systemen
IF-MAP	Interface for Metadata Access Point
IP	Internet Protocol
ITK-Unternehmen	Informations- und Telekommunikationstechnikunternehmen
JDK	Java Development Kit
JSON	JavaScript Object Notation
KMU	kleine und mittelständische Unternehmen
KVM	Kernel-based Virtual Machine
LDAP	Lightweight Directory Access Protocol
MAP	Metadata Access Point
MAPC	Metadata Access Point Client
MSI	Microsoft Installers
NAC	Network Access Control
NBAD	Network Behaviour Anomaly Detection
NVD	National Vulnerability Database
PDP	Policy Decision Point
PEP	Policy Enforcement Point
REST	Representational State Transfer

RFC	Request for Comments
RT	Request Tracker
SEM	Security Event Management
SIEM	Security Information and Event Management
SIM	Security Information Management
SMTP	Simple Mail Network Protocol
SOAP	Simple Object Access Protocol
SSL	Secure Sockets Layer
TCG	Trusted Computing Group
TNC	Trusted Network Connect
VPN	Virtual Private Networks
WLAN-AP	Wireless Local Area Network Access Point
WMI	Windows Management Instrumentation
WSUS-API	Windows Software Update System Application Programming Interface
XML	Extensible Markup Language
XSD	XML Schema Definition