HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

Bachelorarbeit

Herr
Denis Gaist

Prototypischer Aufbau einer sicheren ICS Umgebung für klein- und mittelständische Unternehmen

Mittweida, 2018

Fakultät Angewandte Computer- und Biowissenschaften

Bachelorarbeit

Prototypischer Aufbau einer sicheren ICS Umgebung für klein- und mittelständische Unternehmen

Autor:
Herr Denis Gaist

Studiengang:
IT-Sicherheit

Seminargruppe:
IF15wI-B

Erstprüfer:
Prof. Dr. rer. pol. Dirk Pawlaszczyk

Zweitprüfer:
M.Sc. Philipp Engler

Einreichung:
Mittweida, 08.Oktober.2018

Verteidigung/Bewertung:
Mittweida, 2018

Faculty Angewandte Computer- und Biowissenschaften

Bachelor Thesis

Prototypical design of a secure ICS environment for small and medium-sized enterprises

author:
Mr. Denis Gaist

course of studies:
IT-Security

seminar group:
IF15wI-B

first examiner:
Prof. Dr. rer. pol. Dirk Pawlaszczyk

second examiner:
M.Sc. Philipp Engler

submission:
Mittweida, 08.october.2018

defence/ evaluation:
Mittweida, 2018

Bibliografische Beschreibung:

Gaist, Denis:
Prototypischer Aufbau einer sicheren ICS Umgebung für klein- und mittelständische Unternehmen. - 2018. - 7, 19, 4 S.
Mittweida, Hochschule Mittweida, Fakultät Angewandte Computer- und Biowissenschaften, Bachelorarbeit, 2018

Referat:

Die Vorliegende Arbeit befasst sich mit einem Prototypischen Aufbau einer ICS Umgebung für klein und mittelständische Unternehmen. Ziel ist es, einen Server einzurichten, von dem aus die Clients Zentral konfiguriert und überwacht werden können. Soweit es möglich ist, soll für das Projekt nur Open Source Software eingesetzt werden, um für die Unternehmen keine hohen Kosten zu verursachen.

Inhalt

Abbildungsverzeichnis

Abkürzungsverzeichnis

ICS	Industrial Control System (Industrielle Steuerungssysteme)
NTP	Network Time Protocol
SSH	Secure Shell

1 Übersicht

In diesem Kapitel soll das Problem, mit dem sich die Bachelorarbeit beschäftigt, und das zu erarbeitende Ziel vorgestellt werden. Zudem soll auf die einzelnen Kapitel der Arbeit eingegangen werden.

Dieses Bachelor Projekt baut auf dem Praktikumsbeleg zum selbigen Thema auf.

1.1 Problemstellung

Als die ersten Fertigungsmaschinen und deren Steuerungen entwickelt wurden, war es nicht vorgesehen diese in das bestehende Firmennetz einzubinden oder gar mit Internetzugang auszustatten. Bei der Entwicklung der Protokolle mit denen die ICS (Industrial Control System) kommunizieren war das Thema Sicherheit dementsprechend weniger relevant, da diese Systeme von außen nicht erreichbar waren. [1]

Dadurch, dass die Unternehmen aber auf immer mehr Daten von den verschiedenen Fertigungsprozessen an verschiedenen Standorten zugreifen möchten und auch immer mehr automatisiert werden soll, werden auch immer mehr Steuerungssysteme mit in das Firmennetz integriert. Dabei wird aber oft nicht bedacht, dass diese Systeme ohne entsprechende Sicherheitsmaßnahmen schnell zur Zielscheibe für Angreifer werden können.

Neben einigen Angriffen, die bekannte Sicherheitslücken oder falsche Konfigurationen ausnutzen, zu denen es auch Fallbeispiele vom BSI [2] gibt, gab es auch schon sehr professionelle Angriffe mit speziell dafür entwickelter Schadsoftware. So wurde zum Beispiel 2010 der Computerwurm Stuxnet [3] entdeckt, welcher gezielt für Steuerungssysteme der Firma Siemens geschrieben wurde.

1.2 Zielsetzung

In meinem Praktikumsbeleg wurde schon auf verschiedene Ansätze eingegangen, welche das ICS-Netz sicherer machen sollen und wie dieses Verwaltet werden könnte.

Aufbauend auf diesen Ansätzen soll in diesem Bachelor Projekt eine Testumgebung mit einem Debian Server und einem Debian Client, welche mit Puppet und Foreman verwaltet werden sollen, Installiert werden. Ferner soll der Server über Webmin erreichbar sein und über Puppet sollen Verschiedene Module verwaltet werden.

1.3 Kapitelübersicht

Nach dieser Übersicht folgt ein Kapitel mit Hinweisen zum Betriebssystem und was bei dessen Einrichtung zu beachten ist. Anschließend soll das Konfigurationsmanagementtool Puppet und die dafür entwickelte Weboberfläche Foreman vorgestellt und installiert werden.

Darauf folgt ein Kapitel über das Programm Webmin, welches zur Fernwartung von Servern entwickelt wurde.

Im fünften Kapitel soll es darum gehen, was Puppet Module genau sind und wie man diese installiert, bzw. selbst erstellt. Darauf aufbauen soll im sechsten Kapitel erläutert werden, wie man diese Module in Foreman integrieren und benutzen kann.

Im letzten Kapitel sollen Überlegungen angesprochen werden, welche in einem Unternehmen zusätzlich zu berücksichtigen sind.

2 Betriebssystem

Bei der Wahl des Betriebssystems für ICS spielen natürlich andere Faktoren eine Rolle als bei den Büro Computern der Firma. Das System darf nicht Fehleranfällig sein und sollte sehr lange betrieben werden können, ohne sich aufzuhängen oder Neustarts für z.B. Updates zu benötigen. [4]

2.1 Debian

Debian eignet sich deshalb gut, weil es vor der Veröffentlichung der *stable* Version ausgiebig getestet wird und nach der Veröffentlichung nur noch Sicherheitsupdates eingespielt werden, die in der Regel ohne Neustart wirksam werden [5]. Zudem ist das Betriebssystem kostenlos und bietet mit dem „Debian Long Term Support“ einen langen Supportzeitraum von mindestens 5 Jahren [6].

2.1.1 Installation und Konfiguration

Bei der Installation des Betriebssystems müssen ein paar Dinge beachtet werden, welche im Folgenden aufgelistet werden. Alle anderen Einstellungen können beliebig getroffen werden.

2.1.1.1 Servername

Um die Arbeit mit Puppet zu erleichtern suchen Clients nach der Installation im Netzwerk nach dem Hostname „puppet“. Wird dieser gefunden, verschickt der Client automatisch eine Anmeldung zu der dazugehörigen IP-Adresse.

Natürlich kann man in den Konfigurationen des Clients auch manuell die IP-Adresse des Puppet-Masters eintragen und dem Server somit anders benennen, möchte man sich die Arbeit aber leichter machen, gibt man dem Server einfach den Hostnamen „puppet“.

Die Hostnamen der Clients ist nicht relevant, es bietet sich allerdings an, diese nach einem entsprechenden Muster zu benennen wie z.B. Client_<Maschinentyp>_<Nummer>.

2.1.1.2 Sprache

Bei der Installation von Foreman wird neben Puppet auch direkt ein Datenbanksystem mit Installiert. Dieses arbeitet mit dem Sprachstandard en_US UTF8.

Installiert man also ein Betriebssystem, mit einer anderen Sprache, so muss man nach der Installation entweder mit dem Befehl „sudo dpkg-reconfigure locales“ die Sprache entsprechend hinzufügen oder direkt die entsprechende Datei unter „/etc/default/locale“ anpassen.

Da Foreman nur auf dem Server installiert wird und auch nur dort Mit der Datenbank arbeitet, ist die gewählte Sprache auf den Clients nicht relevant. In der Testumgebung wurde auf dem Server ein entsprechendes englisches Betriebssystem genutzt, während auf dem Client ein Deutsches Betriebssystem lief.

2.1.1.3 Namensauflösung

Damit die verschiedenen Clients den Puppet-Master im Netzwerk auch finden können, müssen diese entsprechend angepasst werden. Hierzu wurde auf allen Clients in der Datei „/etc/hosts“ ein entsprechender Eintrag mit der IP-Adresse des Puppet-Masters und dem Hostnamen „puppet“ hinzugefügt.

Zusätzlich wurde auch auf dem Server für jeden Client ein Eintrag mit entsprechendem Namen und IP-Adresse hinzugefügt.

Statt alle Namenszuordnungen mit der Hand zu pflegen, ist es natürlich auch möglich das mittels eines Domain Name Servers zu managen.

2.1.1.4 SourceList

Während des Projektes ist es beim Installieren von verschiedenen Puppet-Modulen zu einem Fehler gekommen, welcher darauf zurückzuführen war, dass in der Datei „/etc/apt/sources.list“ zwei Einträge waren, welche mit „deb cdrom…“ anfingen.

Nachdem man diese Einträge auskommentierte, wurden alle Installationen fehlerfrei durchgeführt.

2.1.1.5 Super User Rechte

Bei dem für das Bachelor Projekt genutzte Debian9 ist der Befehl „sudo“ in der Standardinstallation nicht enthalten. Um nicht alle Befehle nun mit „su“ als Superuser ausführen zu müssen, sollte dieses noch mit einem einfachen „apt-get install sudo“ nachinstalliert und der entsprechende Nutzer als „sudoer“ eingetragen werden

3 Puppet und Foreman

Da bei der Installation von Foreman auch ein Puppet-Master mit installiert wird [7], sollen die beiden Programme in diesem Kapitel zusammen behandelt werden.

3.1 Funktionsweise

3.1.1 Puppet

Puppet ist ein Administrationsprogramm um mehrere Computer über das Netzwerk automatisiert zu konfigurieren. Es können z.B. Software installiert, Programme ausgeführt oder Daten synchronisiert werden. Puppet funktioniert prinzipiell plattformübergreifend, ist aber auf manchen Betriebssystemen, darunter auch Windows, nur eingeschränkt [8] einsetzbar.

Puppet arbeitet nach dem Client-Server-Prinzip. Der Administrator kann auf dem Server deklarativ über sogenannte Manifeste den Soll-Zustand der Clients festlegen. Über eine REST-API fragen Clients in einem bestimmten Intervall ihre Konfiguration ab und vergleichen den vom Server übermittelten Soll-Zustand mit dem eigenen Ist-Zustand und versuchen anschließend alle Abweichungen anzupassen.

Wird ein Client neu installiert und findet einen Server im Netzwerk, so erstellt der Client als erstes ein Zertifikat, welches dann an den Server übermittelt wird. Dort muss das Zertifikat von einem Administrator bestätigt und somit signiert werden, um auf Anfrage des Clients wieder zurückgeschickt zu werden. Erst wenn das geschehen ist, kann der Client weitere Informationen vom Server bekommen und wird automatisch in regelmäßigen Abständen seinen eigenen Soll-Zustand abfragen.

Die Manifeste, in denen deklariert wird, wie das Zielsystem konfiguriert werden soll, werden in einer eigens dafür entwickelten anwendungsspezifischen Sprache geschrieben. Zusätzlich kann man in den Manifesten über das Integrierte Programm „facter“ Systemeigenschaften ermitteln, was dem Administrator eine Fallunterscheidung erlaubt und somit dasselbe Manifest z.B. für mehrere Betriebssysteme anwendbar macht.

3.1.2 Foreman

Foreman ist ursprünglich für die Open Source Version von Puppet entwickelt worden, da man in dieser Version keine Möglichkeit hat die Clients über eine Oberfläche zu managen. Inzwischen wurde Foreman zwar auch für Zahlreiche andere Konfigurationsmanagement-

tools angepasst, die Integration von Puppet ist allerdings noch am weitesten fortgeschritten.

Foreman bietet dem Administrator eine Weboberfläche, von der aus man einen Überblick über alle Clients im Netzwerk bekommt. Zusätzlich sieht man auch, welche Clients ihre vorgegebenen Konfigurationen erfolgreich umgesetzt haben, welche schon länger nicht mehr mit dem Server verbunden waren und welche bei dem Versuch die Konfiguration umzusetzen auf Fehler gestoßen sind.

Natürlich kann man den Clients über die Weboberfläche auch neue Manifeste zuweisen oder nicht mehr benötigte Manifeste abwählen oder die Zertifikate von neuen Clients dort akzeptieren.

3.2 Installation

Im Folgenden soll die Installation für den Server und Client beschrieben werden.

3.2.1 Server

Auf dem Server wird Foreman installiert, was den Puppet-Master automatisch mit installiert [9]. Hierzu muss man zu aller erst dafür sorgen, dass der Server Zugriff zu den Repositories hat. Dies geschieht mit den Terminalbefehlen:

Puppet:

```
puppet:~$ sudo apt-get -y install ca-certificates
puppet:~$ wget https://apt.puppetlabs.com/puppet5-release-stretch.deb
puppet:~$ sudo dpkg -i puppet5-release-stretch.deb
```

Foreman:

```
puppet:~$ echo "deb http://deb.theforeman.org/ stretch 1.16" | sudo tee /etc/apt/sources.list.d/foreman.list
puppet:~$ echo "deb http://deb.theforeman.org/ plugins 1.16" | sudo tee -a /etc/apt/sources.list.d/foreman.list
puppet:~$ wget -q https://deb.theforeman.org/pubkey.gpg -O- | sudo apt-key add –
OK
```

Danach kann man mit

```
puppet:~$ sudo apt-get update && sudo apt-get -y install foreman-installer
```

die Foreman Installationsdatei runterladen und anschließend mit den entsprechenden Parametern für Admin Username und Password ausführen.

```
puppet:~$ sudo foreman-installer --foreman-admin-username admin --foreman-admin-password "toor"
```

Sobald die Installation abgeschlossen ist, sollte es möglich sein auf die Weboberfläche über https://<Puppet-Master-IP> oder https://<Puppet-Master-Hostname> zuzugreifen und sich dort mit den zuvor festgelegten Parametern für Username und Password anzumelden.

3.2.2 Client

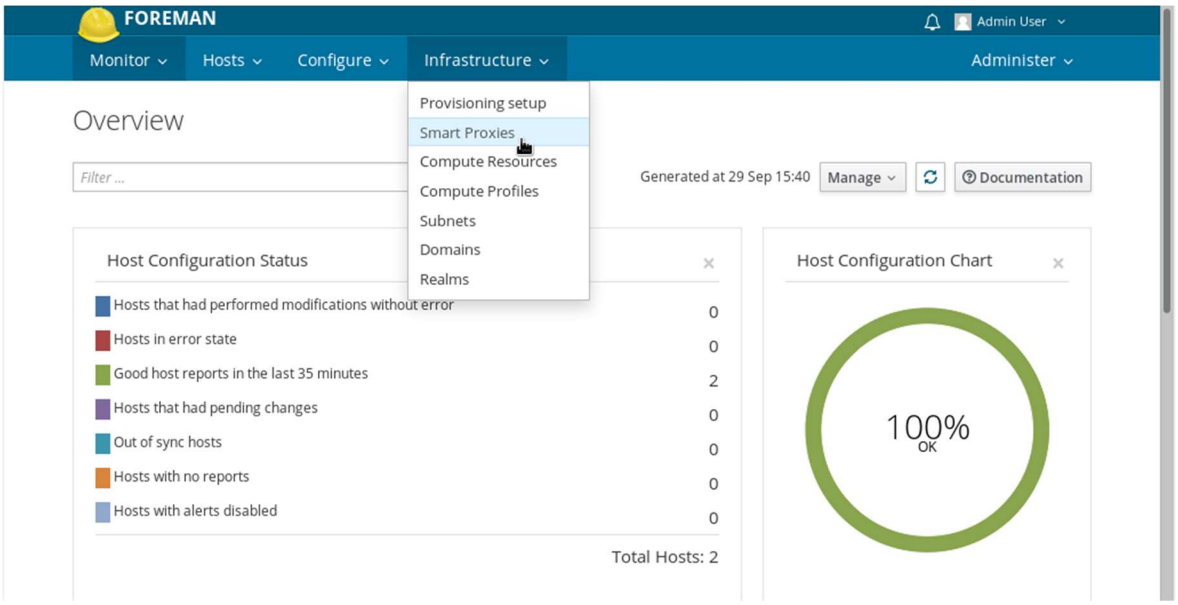
Auf dem Client muss nun nur noch der Puppet-Agent installiert werden. Hierfür muss man genau wie bei der Serverinstallation das Puppet-Repository einbinden.

Danach muss nur noch ein „sudo apt-get install puppet“ ausgeführt werden um den Agenten zu installieren. Um zu testen, ob der Client den Server auch erreicht, wird nun folgender Befehl ausgeführt:

client:~$ puppet agent -td –server=puppet

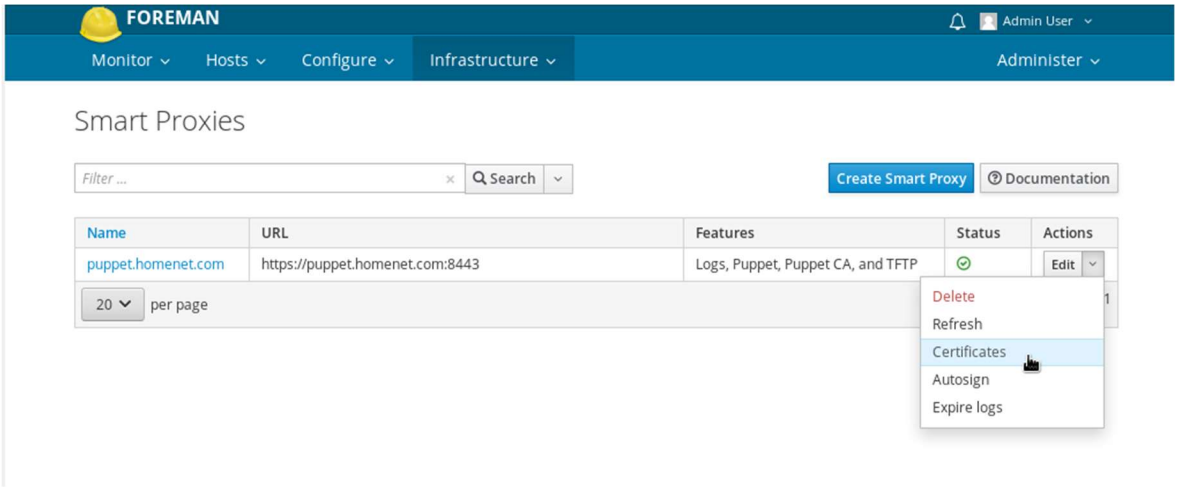
Dieser bewirkt, dass auf dem Client das Zertifikat generiert und an den Server gesendet wird.

Auf dem Server logt man sich nun in der Foreman Weboberfläche ein um über den „Infastructure“-Tab zu den „Smart Proxies“ zu navigieren.



1 Zu den "Smart Proxies" navigieren

Anschließend sieht man eine Auflistung mit genau einem Server. Über die Auswahl des Buttons „Edit“ kann man nun alle Zertifikate sehen, die an den Server gesendet wurden.



2 Zertifikate editieren

Auf dem neuen System sind nun zwei Zertifikate. Das erste vom Server selbst und das zweite, welches eben vom Client gesendet wurde. Letzteres kann nun signiert werden, um somit den Client zu Authentifizieren.

3.3 Nutzung

Über die Oberfläche in Foreman kann man sich nun auf einen Blick neben allen wichtigen Informationen auch alle Hosts anzeigen lassen. Außerdem kann man die Hosts, Gruppen und deren zugeordneten Module verwalten.

Für dem Puppet-Master können nun entweder eigene Manifeste geschrieben werden, oder bereits Fertige Module von der Community importiert werden, um diese dann einfach auf die gewünschten Clients verteilen zu können.

Eine ausführlichere Erläuterung zu den Puppet Modulen folgt in Kapitel fünf und zur Clientverwaltung über Foreman in Kapitel sechs.

4 Webmin

In einem Unternehmen arbeiten oftmals mehrere Administratoren, welche für verschiedene Aufgabenfelder zuständig sind. Um es diesen zu ermöglichen ihre Aufgaben am Server auch von außerhalb zu erfüllen und ihnen gleichzeitig nur auf die Bereiche Zugriff zu gestatten, die sie benötigen, bietet sich das Open Source Programm Webmin als Lösung an.

4.1 Installation

Webmin lässt sich sehr schnell und einfach über das Terminal installieren. Hierzu müssen nur die folgenden beiden Befehle ausgeführt werden.

puppet:~$ sudo wget
http://prdownloads.sourceforge.net/webadmin/webmin_1.890_all.deb

puppet:~$ sudo dpkg --install webmin_1.890_all.deb

Anschließend kann man über den Webbrowser auf den Server zugreifen, in dem man als Adresse https://<Puppet-Master-IP-Adresse>:10000 eingibt und sich mit den Anmeldedaten des Root-Accounts dort anmeldet.

4.2 Nutzung

Meldet man sich nun über Webmin an, fällt einem recht schnell auf, dass man mit dem Root Account sehr viele Informationen über den Server erhält und man auch eine Vielzahl von Einstellungen vornehmen kann. Wenn jeder Administrator über solche Berechtigungen verfügen würde, wäre das eine enorme Sicherheitslücke. Deshalb sollte man als ersten Schritt für jeden Administrator einen Webmin Nutzer anlegen. Diesen Nutzern kann man neben umfangreichen Einschränkungen, auf welche Bereiche des Servers sie zugreifen können auch soweit einschränken, dass sie sich nur an bestimmten Tagen, zu bestimmten Uhrzeiten einloggen können oder auch bestimmte IP-Adressen Black oder White listen. Wobei man hier nicht nur mit einfachen IP-Adressen sondern auch mit Hostnamen, Netzwerkmasken, und Wildcards, wie z.B. *.hs-mittweida.de, arbeiten kann [10].

Hat man nun einen Webmin Account kann man über die Weboberfläche, mit den Entsprechenden Berechtigungen, vom Starten und Stoppen von Prozessen, über Installieren neuer Programme, bis hin zum Konfigurieren der Softwarefirewall alles einstellen.

5 Puppet Module

Puppet arbeitet, indem es die in den Manifesten festgelegte Konfiguration versucht anzuwenden. Damit man aber durch Puppet installierte Programme direkt richtig konfigurieren kann, benötigt man Module. Module sind Ordner, in denen sowohl das Manifest als auch andere Ordner und Dateien abgelegt sind, die zum Konfigurieren benötigt werden, bzw. Dokumente die Puppet als Quelldatei verwenden soll [11].

So zum Beispiel besteht das Modul für das Programm Snort, welches durch dieses Bachelor Projekt entstanden ist, aus dem Manifest, zwei Textdokumenten, welche die Installationsanweisungen und Systemkonfigurationen enthalten und einem Ordner, in dem alle Regeln abgelegt sind, die Snort anwenden soll.

Der große Vorteil, der sich daraus ergibt ist, dass man alle Konfigurationen der Installierten Programme an einem Punkt Zentral bearbeiten kann. Um beim Beispiel Snort zu bleiben, kann man in dem Modul auf dem Server Regeln abändern, löschen oder hinzufügen und jeder Client, dem das Modul zugewiesen wurde, wird feststellen, dass die Quelldatei auf dem Server und die eigenen Regeln nicht mehr identisch sind und die neuen Regeln vom Server übernehmen.

5.1 Puppet Forge

Einer der großen Vorteile von Puppet, ist die große Community, welche neben Fragen in Foren beantworten auch sehr viele Module für Puppet schreibt, und diese in der sogenannten Puppet Forge zur Verfügung stellt, damit andere diese runterladen und nutzen oder auch erweitern können [12].

Für die meisten bekannten Anwendungen existiert somit schon ein Modul, welches man sich runterladen und auf die eigenen Spezifikationen anpassen kann. Dadurch erspart man sich einiges an Zeit, die für das Schreiben der Manifeste benötigt würde. Zudem wird bei jedem Modul auf mögliche Abhängigkeiten der Programme hingewiesen um Fehler, die beim Installieren auf dem Client vorkommen könnten, vorzubeugen.

5.1.1 Module aus der Forge

Wie bereits erwähnt, existieren für die meisten Programme bereits Module, die nur noch über einen einfachen Terminalbefehl auf dem Puppet-Master integriert werden müssen. Der Befehl beginnt mit „puppet module install“ gefolgt vom Namen des Modules und der gewünschten Versionsnummer nach einem „--".

Um also z.B. den NTP Service von Puppet managen zu lassen führt man zuerst den Befehl „puppet module install puppetlabs-ntp --version 7.3.0“ aus und muss dann nur noch den Entsprechenden Server in dem NTP Manifest eintragen.

Für das Bachelor Projekt wurden deshalb die hier aufgelisteten Module aus der Puppet Forge integriert:

- NTP – Network Time Protocol:

  Das Network Time Protocol dient dazu, die Uhren innerhalb eines Netzwerkes Synchron zu halten. Zusätzlich bietet es sich an einen eigenen NTP-Server als Quelle anzugeben, um im ganzen Netzwerk eine gleiche, und von dritten unabhängige, Uhrzeit zu haben. Dass die Uhrzeiten möglichst einheitlich sind, kann für Dienste oder auch Prozessabläufe innerhalb eines Unternehmens wichtig sein.

- SSH – Secure Shell:

  Secure Shell bezeichnet ein Programm, welches ein Netzwerkprotokoll mit demselben Namen nutzt, um sich über eine verschlüsselte Netzwerkverbindung mit einem anderen Gerät zu verbinden. Auf dem Zielrechner kann man somit Terminalbefehle ausführen lassen um z.B. eine Fernwartung durchzuführen. Über die Konfigurationsdatei im Puppet Modul lassen sich bezüglich der Authentifizierungsmethode und anderen Einstellungen, wie z.B. ob eine Anmeldung als Root über SSH erlaubt ist, beliebig anpassen.

- Software Firewall:

  Mit dem Firewall Modul aus der Puppet Forge, welches von Puppetlabs selbst erstellt wurde, lassen sich die Tabellen für iptables, über das Netzwerk einheitlich konfigurieren. Hiermit kann man über das Programm iptables z.B. einstellen, dass alle Datenpakete, welche nicht aus dem eigenen Netzwerk oder auf einen bestimmten Port kommen grundsätzlich nicht angenommen werden.
  Welche Regeln man hier genau festlegt wird natürlich von vielen Faktoren beeinflusst und sollte gut durchdacht sein. Ansonsten droht ein sogenannten O*ver-* oder *Under Blocking*. Sind die Regeln zu streng, wird möglicher-

weise auch gewollter Datenverkehr geblockt, was zu Problemen in der Produktion und Zeitverlust führen kann. Sind die Regeln allerdings nicht streng genug, wird eventuell gefährlicher Datenverkehr nicht gestoppt und kann zu schaden auf den Rechnern führen.

- Fail2Ban

  Fail2Ban ist ein Programm, womit beispielsweise Brute Force[1] Angriffe gestoppt werden sollen indem es aus verschiedenen Protokolldateien (u.a. /var/log/pwdfail oder /var/log/error.log) ermittelt, von welcher IP-Adresse jemand in kurzer Zeit zu viele falsche Passworteingaben getätigt hat, um diese dann für eine bestimmte Zeit zu sperren. Auch bei diesem Programm lässt sich über die Konfigurationsdatei im Puppet Modul einstellen, wie viele Versuche in welchem Zeitraum getätigt werden müssen und wie lange die entsprechende IP-Adresse gesperrt werden soll.

- Virenschutz

  Obwohl das ICS aus Debian Systemen besteht und somit die Gefahr durch Malware recht gering ist [13], wurde für das Projekt der, in der Puppet Forge erhältliche, Virenschutz ClamAV installiert. Dieser soll dadurch mehr Sicherheit für das gesamte Firmennetzwerk schaffen, dass er auch Malware erkennt, welche für Windowssysteme geschrieben wurde und sie somit entfernt, bevor sie schaden im Netzwerk anrichten können.

5.1.2 Selbst erstellte Module

Trotz der Anzahl an Modulen, welche man in der Puppet Forge findet, kann es vorkommen, dass man eine Aufgabenstellung oder ein Programm hat, für das man selbst eine Lösung entwickeln muss. Hierfür gibt es ein sehr ausführliches Tutorial von Puppetlabs [14], aber auch in anderen Foren findet man viele Tipps, wie man die ersten Manifeste gestallten muss, damit zum Schluss auch alles funktioniert.

Für das Bachelorprojekt sollte das Intrusion Detection System Snort verwendet werden um den Datenverkehr von und zu den Clients zu überwachen. Es dient als Ergänzung zur Firewall und kann bestimmte Muster in den Datenpaketen erkennen um zum Beispiel un-

[1] Bei einem Brute-Force-Angriff handelt es sich um eine Methode, die versucht Passwörter oder Schlüssel durch automatisiertes, wahlloses Ausprobieren herauszufinden. [17]

erwünschte Verbindungen vom, oder Angriffe auf das System zu Identifizieren und diese dann Protokollieren und unterbinden. Welche Verbindungen erlaubt und welche unerlaubt sind wird anhand der Snort Regeln definiert.

Des Weiteren sollen die Zertifikate, die von Puppet erstellt und signiert wurden zusätzlich in den entsprechenden Ordner unter „/etc/ssl/“ kopiert werden um diese später eventuell noch für andere Zwecke verwenden zu können.

Um die Zertifikate auf den Clients in einen anderen Ordner zu kopieren muss nur sichergestellt werden, dass die Zieldatei über die Richtigen Berechtigungen verfügt und dass die Ordner auch tatsächlich existieren. Für das Kopieren, bzw. Anlegen einer Datei oder eines Ordners wird das Konstrukt „file“ benutzt. In diesem gibt man an, ob es sich um eine Datei oder einen Ordner handeln soll, welche Berechtigungen gesetzt werden sollen, wer der Eigentümer ist oder zu welcher Nutzergruppe die Datei zugehörig ist, und die Quelle aus der Kopiert werden soll.

Um Snort auf den Zielsystemen richtig zu installieren benötigt man zwar keine komplexeren Funktionen im Manifest, allerdings muss einiges beachtet werden, wie zum Beispiel, dass bei der manuellen Installation von Snort einige Parameter abgefragt werden, welche zwingend benötigt werden. Deshalb muss man, nachdem man Snort von Puppet installieren lassen hat, neben den Regeln, die verwendet werden sollen, auch noch die Konfigurationsdatei von Snort und die Installationsparameter, welche unter „/etc/default/snort“ gespeichert werden, vorgeben. Zuletzt muss nur noch der Dienst von Snort gestartet werden.

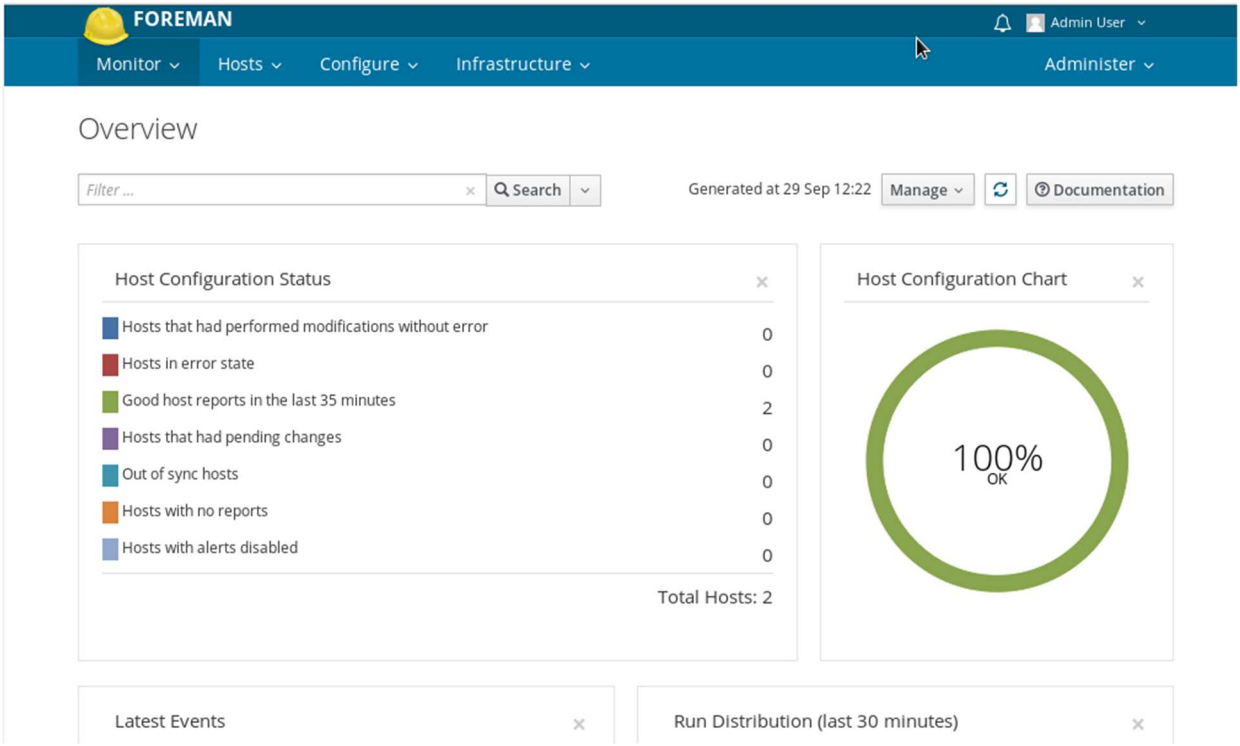
Beide Manifeste können in den Anlagen eingesehen werden.

6 Clients verwalten

Sind alle Module, welche man verwenden möchte, auf dem Puppet-Master installiert und für das Unternehmen konfiguriert, kann man Foreman diese nun importieren lassen und den einzelnen Hosts bzw. den Gruppen zuweisen. Deshalb soll in diesem Kapitel darauf eingegangen werden, wie man Module in Foreman importierts, diese bestimmten Clients zuweist und wie man Konfigurationsgruppen erstellt.

6.1 Foreman Dashboard

Das erste was man sieht, nachdem man sich in Foreman eingeloggt hat, ist das Dashboard. Dieses bietet einen schnellen Überblick, wie viele Clients vor kurzem eine neue Konfiguration erfolgreich durchgeführt haben, wie viele Clients auf einen Fehler gestoßen sind beim Umsetzten einer Konfiguration, wie viele Clients eine positive Rückmeldung gegeben haben und somit derzeit keine Konfigurationen durchführen müssen und wie viele Clients „Out of sync“ sind und somit entweder ausgeschalten sind oder keine Verbindung zum Puppet-Master herstellen können.



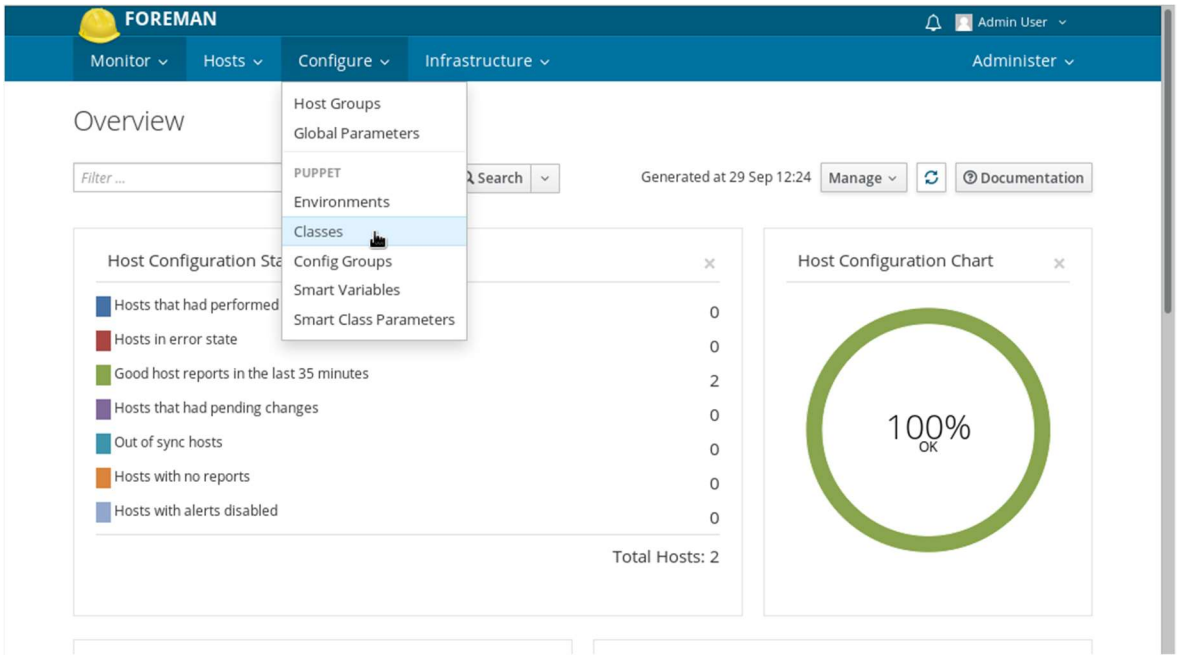
3 Foreman Dashboard

Um zum Beispiel herauszufinden, welche Clients eine Konfiguration nicht erfolgreich durchführen können, kann man auf „Hosts in error state“ klicken um eine genaue Auflis-

tung zu bekommen, welche Clients betroffen sind und welches Modul diesen Fehler jeweils ausgelöst hat.

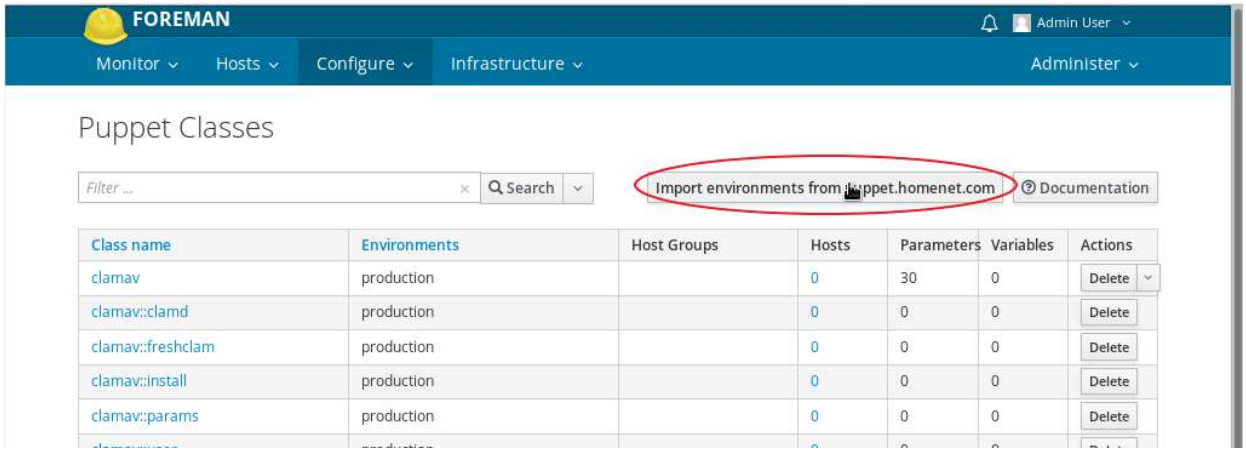
6.2 Puppet Module

Um die Puppet Module den Clients zuteilen zu können, müssen diese zuerst von Foreman importiert werden. Hierzu navigiert man über „Configure ➔ Classes“



4 Module verwalten

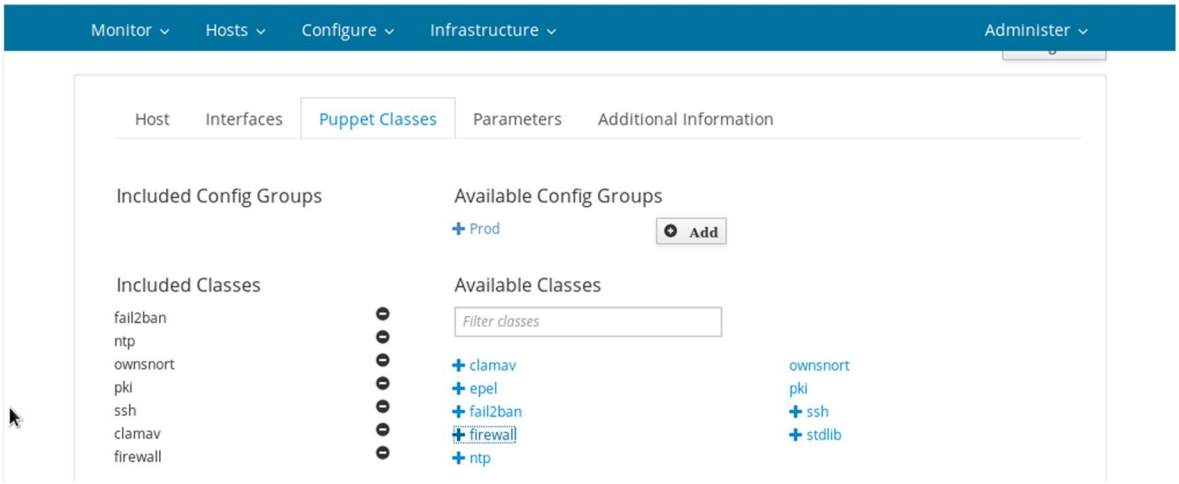
um über den Button „Import“ die installierten Module zu Foreman hinzuzufügen.



5 Module importieren

Sind die gewünschten Module in Foreman importiert, kann man über „Hosts ➔ all Hosts“ die Liste aller Puppet Clients und Server einsehen, welche über ein signiertes Zertifikat verfügen. Über den Button „Edit“ wählt man nun den gewünschten Client aus um bei der

Registerkarte „Puppet Classes“ von den Importierten Modulen jene auszuwählen, die auf diesem Client angewendet werden sollen.

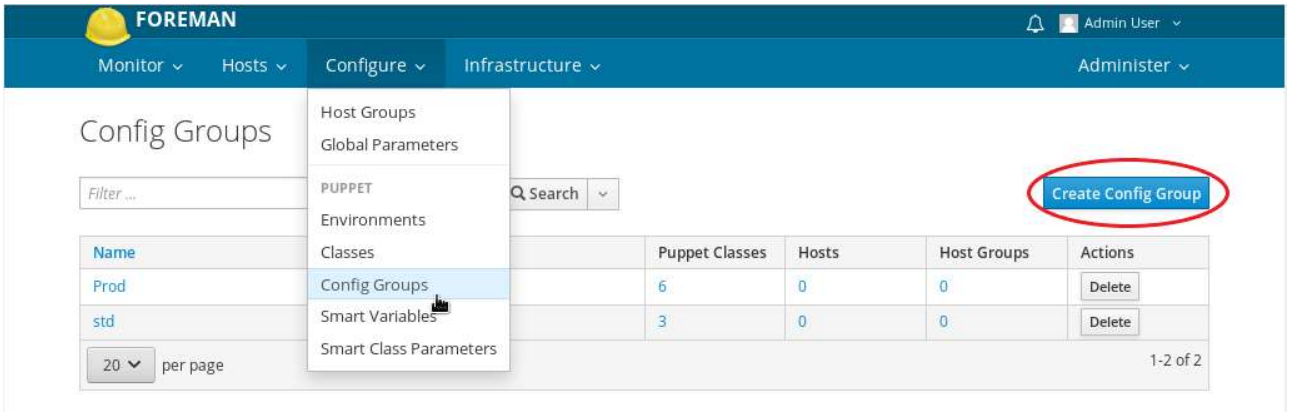


6 Module Client zuweisen

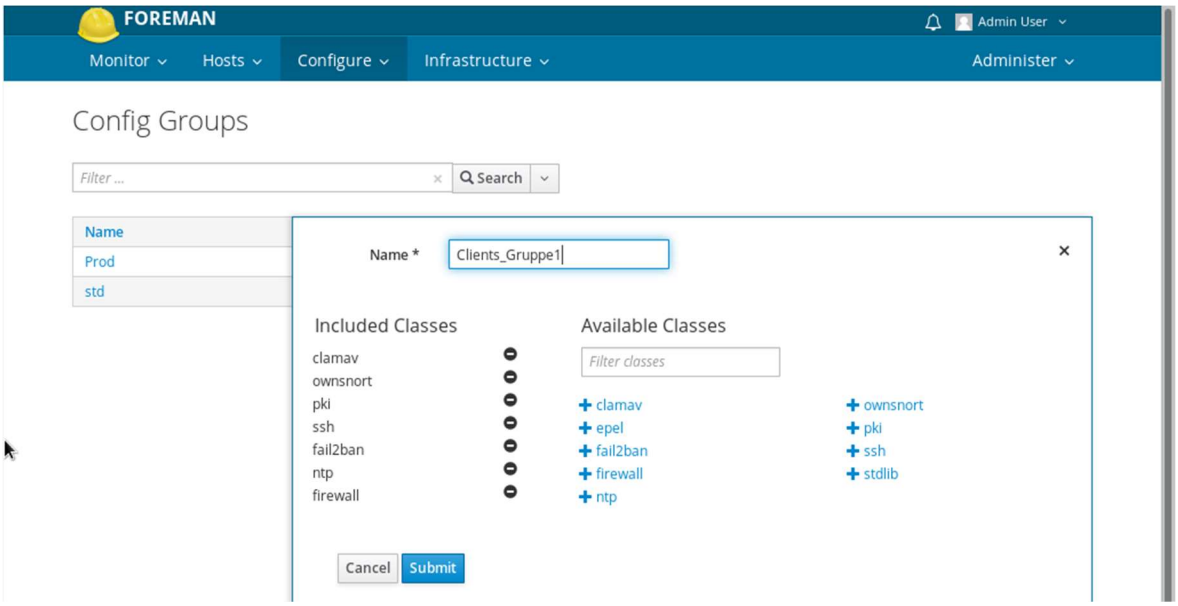
Hat man alle gewünschten Module ausgewählt, wird der Client bei seiner nächsten Anfrage an den Server versuchen diese anzuwenden und dem Server anschließend eine Erfolgs- oder Fehlermeldung zurückgeben.

6.3 Konfigurationsgruppen

In einem Unternehmen wird es mehrere Clients geben, die dieselbe Aufgabe haben bzw. dieselbe Konfiguration benötigen. Damit der Administrator solchen Clients nicht alle Module manuell zuweisen muss kann man in Foreman Konfigurationsgruppen erstellen. Hierfür navigiert man über „Configure ➔ Config Groups“ um mit dem Button „Create“ eine neue Gruppe anzulegen und dieser die gewünschten Module zuzuordnen.

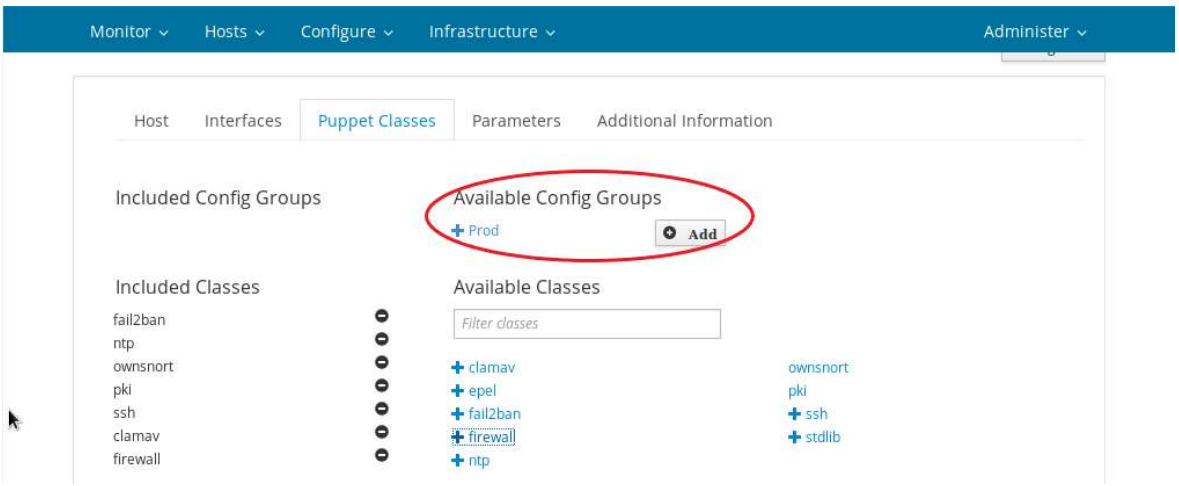


7 Konfigurationsgruppen



8 Konfigurationsgruppe erstellen

Anstatt bei neuen Clients jedes Mal alle Module manuell hinzufügen zu müssen, kann man nun einfach die betreffende Gruppe auswählen und hat somit alle benötigten Module auf einmal zugewiesen.



9 Client Konfigurationsgruppe hinzufügen

7 Praktische Umsetzung

Das Computernetzwerk, welches für diese Bachelorprojekt aufgebaut wurde, ist in einer Virtuellen Umgebung entstanden. Somit mussten einige Überlegungen, welche in einem klein- und mittelständigen Unternehmen notwendig wären nicht beachtet werden.

Im letzten Kapitel sollen diese näher betrachtet werden. Außerdem sollte bedacht werden, dass das in diesem Projekt erstellte System nur einen prototypischen Aufbau beschreiben soll und nicht als fertige Lösung zu verstehen ist.

7.1 Hardware

Die Mindestanforderungen an die Hardware um Foreman zusammen mit dem Puppet-Master betreiben zu können entsprechen vier Gigabyte Arbeitsspeicher und zwei Gigabyte Festplattenspeicher [15].

Für diese Anforderungen würden sich Einplatinencomputer, wie z.B. die Intel Up-Boards [16], anbieten. Diese würden nicht nur die Anforderungen erfüllen, sondern gleichzeitig auch preiswerter sein als ein herkömmlicher Büro Computer. Zusätzlich benötigt der Rechner, der als Puppet-Master eingerichtet werden soll eine zusätzliche Netzwerkschnittstelle, da dieser die einzige Verbindung zwischen dem ICS-Netzwerk und dem restlichen Firmennetzwerk sein soll[2].

7.2 Konfigurationen

Die Regeln für Snort und iptables müssen natürlich an das jeweilige Unternehmen angepasst werden. Da solche Einzelheiten in dem Projekt nicht berücksichtigt werden konnten, entsprechen die verwendeten Regeln den Standard Regeln, die bei der Installation der Programme bzw. Module verwendet werden.

[2] Siehe Anlage 3: Aufbau des ICS

Anlagen

Snort Manifest

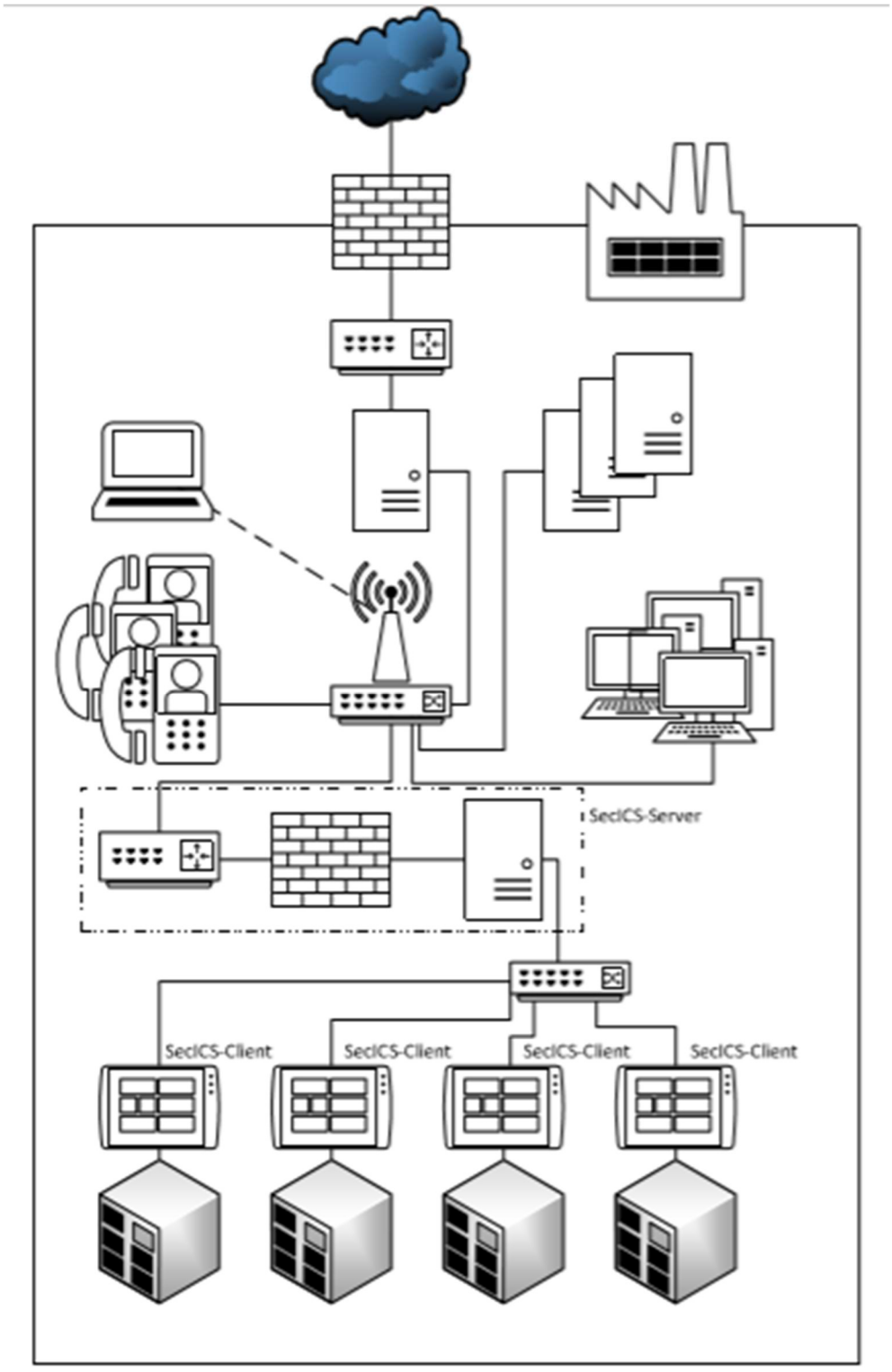
```
class ownsnort {

      package { 'snort':
            ensure => installed,
      }

      package { 'libdaq2':
            ensure => installed,
      }

      file { '/etc/snort/rules/local.rules':
            ensure  => present,
            source  => [ "pup-
pet:///modules/ownsnort/local/local.rules-${::fqdn}",
                  'puppet:///modules/ownsnort/local/local.rules' ],
            mode    => '0644',
            owner   => 'root',
            group   => 'root',
            force   => true,
            notify  => Service['snort'],
            require => Package['snort'],
      }

      file { '/etc/snort/rules':
            ensure  => directory,
            source  => 'puppet:///modules/ownsnort/rules',
            purge   => true,
            recurse => true,
            force   => true,
            mode    => '0644',
            owner   => 'root',
            group   => 'root',
            notify  => Service['snort'],
            require => Package['snort'],
      }

      file { '/etc/snort/snort.conf':
            mode    => '0644',
            owner   => 'root',
            alias   => 'snortconf',
            content => template( 'ownsnort/snort.conf.erb'),
            notify  => Service['snort'],
            require => Package['snort'],
      }

      file { '/etc/default/snort':
            source  => [ "puppet:///modules/ownsnort/sysconfig/snort-
${::fqdn}",
                  'puppet:///modules/ownsnort/sysconfig/snort' ],
            mode    => '0644',
            owner   => 'root',
```

```
            group   => 'root',
            notify  => Service['snort'],
            require => Package['snort'];
      }

      service { 'snort':
                  ensure     => running,
                  enable     => true,
                  hasstatus  => true,
                  hasrestart => true,
                  require    => Package['snort'],
      }
}
```

Zertifikat Manifest

```
class pki {

  group { 'ssl-cert':
    ensure => present,
    system => true,
  }

  $host = $::trusted['certname']

  file { '/etc/ssl/private':
      ensure     =>    directory,
      mode  =>    '641',
      group =>    'ssl-cert',
      require    =>    Group['ssl-cert'],
  }

  file { '/etc/ssl/private/host.key':
      ensure     =>    file,
      source     =>    "/var/lib/puppet/ssl/private_keys/${host}.pem",
      group =>    'ssl-cert',
      mode  =>    '0640',
  }

  file { '/etc/ssl/certs/host.crt':
      ensure     =>    file,
      mode  =>    '644',
      group =>    'ssl-cert',
      source     =>    "/var/lib/puppet/ssl/certs/${host}.pem",
  }

  file { '/etc/ssl/certs/host-ca.crt':
      ensure     =>    file,
      mode  =>    '644',
      group =>    'ssl-cert',
      source     =>    "/var/lib/puppet/ssl/certs/ca.pem",
  }
}
```

Aufbau des ICS



10 **Firmennetzwerk mit ICS-Netzwerk**

Quellen

[1] BSI, „BSI,“ 25 November 2013. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/ICS/ICS-Security_kompendium_pdf.pdf?__blob=publicationFile&v=2. [Zugriff am 13 Juli 2018].

[2] BSI, „Bundesamt für Sicherheit in der Informationstechnik,“ 4 Juli 2018. [Online]. Available: https://www.bsi.bund.de/DE/Themen/Industrie_KRITIS/ICS/Erfahrungsberichte/erfahrungsberichte_node.html;jsessionid=C8BCCE6EC5564CBE6D3692C214D89E40.2_cid341. [Zugriff am 4 Juli 2018].

[3] I. Pakalski, „Golem,“ 27 September 2010. [Online]. Available: https://www.golem.de/1009/78245.html. [Zugriff am 4 Juli 2018].

[4] D.-I. A. Floß, „Bundesamt für Sicherheit in der Informationstechnik,“ 19 Mai 2015. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Veranstaltungen/ITSiKongress/14ter/Vortraege-19-05-2015/Andreas_Floss.pdf?__blob=publicationFile. [Zugriff am 16 September 2018].

[5] „wiki.debian,“ 23 Januar 2017. [Online]. Available: https://wiki.debian.org/DebianStability. [Zugriff am 17 September 2018].

[6] „Wiki.Debian,“ 20 August 2018. [Online]. Available: https://wiki.debian.org/LTS/. [Zugriff am 17 September 2018].

[7] theforeman, „theforeman,“ [Online]. Available: https://www.theforeman.org/manuals/1.19/quickstart_guide.html. [Zugriff am 20 September 2018].

[8] Puppetlabs, „Puppet,“ Puppetlabs, [Online]. Available: https://puppet.com/docs/pe/2017.3/supported_operating_systems.html. [Zugriff am 22 August

2018].

[9] P. Kumar, „LinuxTechi,“ 08 Januar 2018. [Online]. Available: https://www.linuxtechi.com/install-configure-foreman-1-16-debian-9-ubuntu-16-04/. [Zugriff am 16 August 2018].

[10] „Webmin wiki,“ 05 Oktober 2016. [Online]. Available: https://doxfer.webmin.com/Webmin/Webmin_Users. [Zugriff am 17 August 2018].

[11] Puppetlabs, „Puppet,“ Puppetlabs, [Online]. Available: https://puppet.com/docs/puppet/5.3/modules_fundamentals.html. [Zugriff am 15 September 2018].

[12] N. Anderson, „Puppet,“ Puppetlabs, 25 April 2018. [Online]. Available: https://puppet.com/blog/welcome-new-puppet-forge. [Zugriff am 17 August 2018].

[13] Computer-Service-Remscheid, „Computer-Service-Remscheid,“ 31 Mai 2015. [Online]. Available: https://computer-service-remscheid.de/virenscanner-fuer-linux/. [Zugriff am 20 August 2018].

[14] Puppetlabs, „Puppet,“ Puppetlabs, [Online]. Available: https://puppet.com/docs/puppet/6.0/bgtm.html#concept-1345. [Zugriff am 20 August 2018].

[15] theforeman, „theforeman,“ [Online]. Available: https://puppet.com/docs/puppetserver/6.0/install_from_packages.html. [Zugriff am 22 August 2018].

[16] Intel, „up-shop,“ [Online]. Available: https://up-shop.org/home/224-up-squared-atom-quad-core-4gb-memory32gb-emmc-bulk.html?search_query=2+ethernet&results=31. [Zugriff am 13 August 2018].

[17] P. Schmitz, „security-insider,“ 17 Januar 2018. [Online]. Available: https://www.security-insider.de/was-ist-ein-brute-force-angriff-a-677192/. [Zugriff am 16 August 2018].

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Mittweida, den 08.Oktober.2018

_____

Denis Gaist