Sicherheit von ausgewählten OpenSource-SDN-Controllern

Evren Eren[1] · Jonas Sell[2]

[1]HS Bremen
eren.evren@lba.hs-bremen.de

[2]FH Dortmund
jonas.sell@fh-dortmund.de

Zusammenfassung

Software Defined Networking (SDN) erhält immer mehr Einzug in mehr oder wenig komplexe Netzwerksysteme von Firmen, Netzbetreibern etc. Gerade sich schnell verändernde Netze profitieren von SDN und der zentralen Verwaltung. Auch in Cloud-Umgebungen wird SDN zunehmend eingesetzt. Durch die Zentralisierung der Netzadministration und damit Reduktion auf Single-Point-of-Failures bedürfen jedoch SDN-Systeme besonderem Schutz, um klassische Angriffe abzuwehren. Ansonsten ist mit Beeinträchtigungen oder gar Ausfällen des Netzes zu rechnen. Dem SDN-Controller, als zentrales Element in SDN-basierten Netzen, kommt bei der Sicherheitsbetrachtung eine wesentliche Bedeutung zu. Dieses Paper zeigt die Verwundbarkeiten von verschiedenen Open-Source SDN-Controllern wie POX, Floodlight, IRIS, BEACON und OpenDaylight auf und stellt geeignete Gegenmaßnahmen zur Absicherung vor.

1 Einleitung

Durch die zunehmende Komplexität von Netzen steigt der Aufwand im Netzmanagement. Die Vielzahl heterogener Netzkomponenten wie Router, Switches, Firewalls mit unterschiedlichen Betriebssystemen sowie teilweise proprietären Schnittstellen verkomplizieren den Sachverhalt. Oft müssen die Komponenten einzeln konfiguriert werden. Dabei unterscheiden sich die Konfigurationsschnittstellen, je nach Hersteller und unter Umständen auch zwischen Produkten desselben Herstellers, recht stark.

SDN

SDN soll die Konfiguration und Wartung von Netzwerken verändern, indem die Administration an eine zentrale Stelle ausgelagert wird. [FeRZ13] Eine SDN-Umgebung besteht mindestens aus einem Controller und einem SDN-fähigen Netzwerkgerät, auf welches der Controller Zugriff über eine API hat. Der Controller ist das „Gehirn“ des Netzwerkes und kennt dessen Struktur und Einstellungen, wodurch ein direkter Zugriff auf die Konfigurationsoberflächen der Netzwerkgeräte obsolet wird. Hierdurch vereinfacht sich die Administration und die Reaktionszeiten steigen. SDN erfüllt viele Anforderungen für intelligente Netzwerkanwendungen und -funktionen bei gleichzeitig niedrigen Netzbetriebskosten durch simplere Hardware sowie einfaches und schnelles Netzwetzwerkmanagement. Durch Abstraktion der Network-OS, weil Steuerungs- und Datenebene (Control Plane und Data Plane) voneinander getrennt werden,

P. Schartner (Hrsg.) · D•A•CH Security 2016 · syssec (2016) 44-53.

können logische Netzdienste auf heterogener physikalischer Infrastruktur operieren und dadurch flachere Netze resultieren. [Ere15] Durch SDN bekommen insbesondere Cloud-Betreiber eine netzweite Softwareschicht mit zentralisierter Switching- und Routing-Logik, wobei die IT-Infrastruktur von einem SDN-Controller verwaltet und gesteuert wird, der logisch und physikalisch unabhängig von der Infrastruktur agiert. Der immer größere Erfolg von Cloud-Computing treibt den Einzug von SDN weiter voran. Der immense Zuwachs von Server- aber auch Netzwerkvirtualisierung bedeutet entsprechende Anforderungen an die Flexibilität von Layer2-Komponenten wie z.B. Switches. Netze müssen entsprechend flexibel mit virtuellen Verbindungen über virtuelle Switches und Router mit einer variablen Anzahl an Ports gehalten werden, und dieses zentral gesteuert. SDN-Konzepte bedingen jedoch spezielle Protokolle, die die Kommunikation zwischen Controllern und Netzwerkgeräten abwickeln. Aus Kompatibilitätsgründen muss das Protokoll standardisiert sein und von jedem Gerät unterstützt werden. Eine Entwicklung in diese Richtung ist OpenFlow.

OpenFlow

Als Kommunikationsschnittstelle zwischen der Steuerungs- und Weiterleitungsebene einer SDN-Architektur spielt dabei OpenFlow eine wesentliche Rolle. Dieser Standard wird durch die Open Networking Foundation (ONF) [Fou15b] verwaltet. Die Open Networking Foundation ist eine Anwenderorganisation, die sich der Verbreitung und Implementierung von SDN, inkl. der damit zusammenhängenden Protokolle, widmet. OpenFlow wird von Google, Deutsche Telekom, Facebook, Microsoft, Verizon und Yahoo! eingesetzt [Fou15a]. OpenFlow stellt eine standardisierte, über das Netzwerk erreichbare Schnittstelle zur Verfügung, um Datenflüsse in Netzwerkgeräten zu steuern. Es ermöglicht den direkten Zugriff auf die Forwarding-Schicht eines Switches oder Routers. Dies wird sowohl physikalisch als auch virtuell auf Basis eines Hypervisors ermöglicht. Der gesamte Steuerungsfluss aller Netzwerkgeräte kann somit zentralisiert erfolgen. Hierdurch wird das Netzwerkmanagement dahingehend unterstützt, das Netzwerk einfacher verwalten zu können, da eine manuelle Konfiguration der Hardware entfällt. Insbesondere durch Virtualisierung und IT-Trends wie Microservices und containerbasiertes Deployment ist es essentiell, Netzwerke sich an dynamische Lastsituationen anpassen zu können. Beispielsweise können bei steigender Nutzerzahl automatisch zusätzliche Container mit der benötigten Anwendungssoftware hochgefahren und mit in den Clusterverbund aufgenommen werden. Hierzu ist es notwendig, nötige Routen, Firewall-Freischaltungen (beispielsweise Freischaltung von Ports, Protokollen oder IP-Adressen [Hogg15]) und Loadbalancer-Konfigurationen automatisiert vorzunehmen. Mit OpenFlow ist dies ohne Hardwareänderungen automatisiert und zentral möglich. [Fou12]

Controller

Seit Aufkommen von SDN und OpenFlow wurden viele verschiedene Controller-Implementierungen realisiert und einige stellen zusätzliche Funktionen bereit, wie bspw. die Absicherung über TLS, Authentifizierung, REST-API, Web-Frontends zur Datenvisualisierung, Steuerung und Überwachung oder SSH-Zugriff auf die Controller-Schnittstelle.

Sicherheit

Problematisch wird das SDN-Konzept, wenn es in kritischen Netzwerkstrukturen zum Einsatz kommt. Dann hängt die Funktion des Netzwerkes von wenigen oder gar nur einem einzigen, zentralen Punkt ab. Fällt dieser aus oder kann er seine Aufgaben oder die Steuerung des Netzwerkes nicht vollständig umsetzen, ist das Netz gestört oder kann komplett ausfallen. Ausfälle können, neben technischen Defekten, auch von Störungen von außen, wie etwa Angriffe auf die Infrastruktur, hervorgerufen werden. Wird der Controller übernommen, hat der Angreifer

kompletten Zugriff auf Bereiche des Netzwerks, kann Netzwerkgeräte umprogrammieren und Informationen zu weiteren Zielen und Angriffen im Netzwerk sammeln. Angriffe können beispielsweise über Social-Engineering erfolgen, mit dem Ziel, dass Schadsoftware auf dem Controller durch die Administration installiert wird. [McGi14] Aber auch direkte Angriffe auf die Controllersoftware sind möglich. Da nicht mehr jede einzelne Komponente relativ unabhängig agiert, sondern von einer Zentrale gesteuert wird, ist diese ein prädestiniertes Angriffsziel und benötigt besonderen Schutz. Der grundsätzliche Schutz jedoch muss in der Steuerungssoftware selbst umgesetzt werden. Darunter fällt die Absicherung der Kommunikation mit dem Netzwerk/den Netzwerkgeräten, von Netzwerkdiensten, welche von der Software bereitgestellt und von außen erreichbar sind (z.B. SSH-Server, HTTP-Server) sowie des Remotezugriffs (Zugriffssteuerung). Dies sind einige Punkte von neuen Herausforderungen, welche mit der Einführung von SDN in Netzwerke aufkommen. In unseren Untersuchungen fokussierten wir die Sicherheit der Steuerungssoftware. Hierzu wurden mögliche Schwachstellen durch diverse Angriffsanalysen untersucht und entsprechende Maßnahmen zur Absicherung erarbeitet.

2 Schwachstellenanalyse und Angriffsklassen

2.1 Angriffsziele

Prinzipiell existieren in einer SDN-Umgebung drei mögliche Angriffsziele: Controller, OpenFlow-Protokoll und Netzwerkgeräte wie Switche oder Router. Angriffe auf den Controller und den Switch können direkt ausgeführt werden. Dazu muss die IP-Adresse der Systeme bekannt sein, welche sich durch Scans (Scannen des Netzwerkes nach aktiven Hosts, Portscans nach SDN-Ports, beispielsweise mit nmap) herausfinden lässt. Ein Angriff des OpenFlow-Protokolls bedingt, dass der Angreifer Zugriff auf den Datenstrom bekommt. Dies ist beispielsweise durch Umleiten der Daten mittels einer Man-in-the-Middle Attacke möglich. O.g. Angriffe bedingen, dass sich der Angreifer im Netzwerk befindet und die Netzwerkkomponenten erreichen kann.

2.2 Testaufbau

Zur Simulation und Durchführung von Angriffen wurde ein virtuelles Netzwerk auf Basis von OpenVSwitch (http://openvswitch.org), zur Erstellung von virtuellen, SDN-fähigen Switchen und KVM/libvirt (https://libvirt.org), als Laufzeitumgebung für virtuelle Maschinen, realisiert. Dabei wurde ein zentraler OpenVSwitch von einer Controller-VM mit unterschiedlicher Controller-Software gesteuert. Andere VMs im Netzwerk hatten die Rolle von Gateways, Netzwerk-Clients und Angreifer-Maschine.

2.3 Controller-unabhängige Angriffe

Einige der durchgeführten Angriffe sind nicht auf einen bestimmten Controller beschränkt, sondern sind unabhängig von der eingesetzten Controller-Software. Diese Angriffe zielen auf die Übernahme, das Stören und den Mitschnitt von Daten und greifen daher auch das Betriebssystem des Controllers an.

Interception

Damit sich ein Angreifer in die Kommunikation einschleusen, Daten mitschneiden und ggfs. manipulieren kann, reicht ein klassischer Man-in-the-Middle-Angriff aus. Für die Tests wurde das Tool arpspoof gewählt, welches Teil des dsniff-Paketes (https://www.monkey.org/~dugsong/dsniff) ist. Wird der Traffic erfolgreich auf den Angreifer umgeleitet, können die Daten

mitgelesen, verändert und weitere Informationen über das Netzwerk gesammelt werden. [Sand10] Wie in Abbildung 1 zu sehen ist, ist der Datenverkehr unverschlüsselt und teilweise von Menschen lesbar.

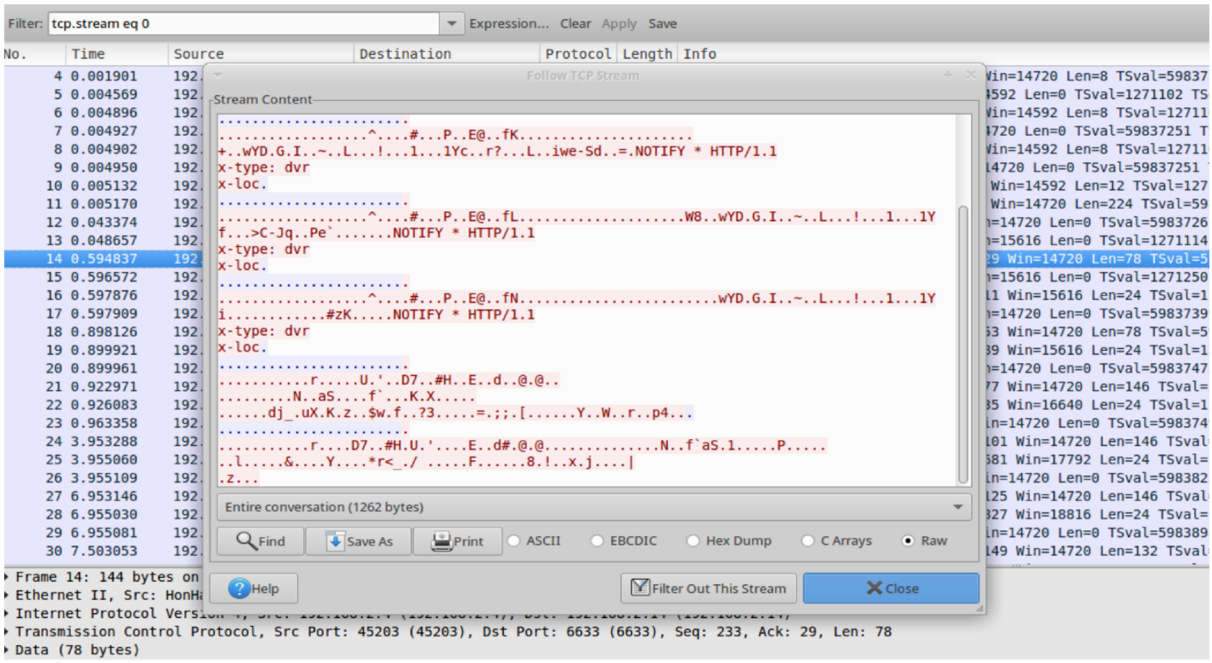

Abb. 1: OpenFlow-Traffic mit Wireshark-Hijacking

Grundlage zur Übernahme des Controllers ist ebenfalls ein Man-in-the-Middle-Angriff. Auf der Angriffs-VM wurde anschließend ein eigener OpenFlow-Controller gestartet und die OpenFlow-Ports zum gefälschten Controller weitergeleitet, wodurch dieser OpenFlow-Verbindungen entgegennahm und eigene Anweisungen an die dann verbundenen Switche schicken konnte. Dabei waren jegliche Angriffe denkbar, z.B. die künstliche Simulation von Paketverlusten für die Teilnehmer des Netzwerks oder weitere Angriffe auf den Rest des Netzwerkes. Schließlich war es nunmehr möglich, Traffic beliebig umzuleiten, zu blockieren und zu verändern. Die wirksamste Methode, den Controller und Netzwerkgeräte gegen Übernahme und Traffic-Mitschnitt zu schützen, ist die Absicherung von OpenFlow mittels TLS. Je nach Controller sollte dabei die zweiseitig authentifizierte Übertragung gewählt werden. Die Zertifikate wurden von derselben CA ausgestellt, sodass eine Vertrauensbasis geschaffen wurde. Verbindungen zu gefälschten Controllern schlugen fehl, da sich dieser nicht korrekt authentifizieren konnte.

Denial-of-Service (DoS)

Ein Denial-of-Service-Angriff auf den Controller bedingt die Kenntnis der IP-Adresse der Maschine und der Ports, auf welchen die Controller-Software Verbindungen annimmt. I.d.R. wird die OpenFlow-Kommunikation über TCP abgewickelt. Der dafür genutzte Port ist entweder bekannt oder er lässt sich durch ein Portscan ermitteln. Ziel des Angriffes ist es, den Controller mit ungültigen Anfragen zu überlasten, damit legitime Verbindungen abgewiesen und nicht bearbeitet werden können. Mit der Nutzung von mehreren Maschinen kann der Angriff verstärkt werden (Distributed Denial-of-Service (DDoS). Ohne DDoS konnte die Controller-VM mit fast 300 MBit/s (Messung mit iperf zwischen Controller und VM im Netzwerk) erreicht werden. Bei einem aktiven Angriff brach die Übertragungsgeschwindigkeit deutlich ein (ca. 80

kBit/s, siehe Abbildung 2). Zwischen den Netzwerkteilnehmern war der Angriff ebenfalls deutlich spürbar. Der Durchsatz zwischen zwei am Angriff unbeteiligten VMs sank pro zusätzlicher Angriffsmaschine um ca. 100 Mbit/s. [McDo09]

```
[root@fedora1 ~]# iperf -c 172.22.161.214
------------------------------------------------------------
Client connecting to 172.22.161.214, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.11 port 56312 connected with 172.22.161.214 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   355 MBytes   297 Mbits/sec
[root@fedora1 ~]# iperf -c 172.22.161.214
------------------------------------------------------------
Client connecting to 172.22.161.214, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.11 port 56314 connected with 172.22.161.214 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-15.0 sec   143 KBytes  78.0 Kbits/sec
```

Abb. 2: Sinkende Übertragungsraten zwischen Clients bei aktivem DoS

3 Untersuchung von SDN-Controllern

Einige SDN-Controller stellen eigene Schnittstellen bereit, beispielsweise für den Remotezugriff. Sind diese nicht ausreichend abgesichert, ist eine Übernahme des Controllers denkbar.

Im Folgenden werden ausgewählte Controller auf die Sicherheit der zusätzlichen Funktionen untersucht und potentielle Maßnahmen zur Verbesserung der Absicherung getroffen.

3.1 POX

Neben den Standardfunktionen von OpenFlow-Controllern bietet POX einen Messenger an, über welchen der Controller mit Software Dritter kommunizieren kann. Eine Authentifizierung von Zugriffen auf die Schnittstelle erfolgt dabei nicht und die Kommunikation erfolgt im Klartext. Weiterhin existiert eine JSON-RPC-Schnittstelle, über welche Informationen über verbundene Switche ausgelesen und Flow-Tables gesetzt werden können. Die Absicherung der OpenFlow-Kommunikation ist erst ab POX Version „DART“ (seit Juli 2014) mit beidseitigem TLS möglich. Der Messenger oder die JSON-RPC-Schnittstelle sind nicht abgesichert und können weiterhin unautorisiert benutzt werden. [AlSh12]

3.2 Floodlight

Floodlight ist ein auf Java basierender OpenFlow-Controller. Nach der Installation präsentiert es sich mit geöffneten Ports für eine REST-API und einen Webserver. Erstere ist in der Standardkonfiguration ungeschützt. Eine Authentifizierung mit Benutzername und Passwort existiert nicht. Funktionen der API umfassen beispielsweise das Auslesen aller verbundenen Switche oder das Verändern des Status eines verbundenen Gerätes. Floodlight unterstützt die Absicherung über TLS sowohl für OpenFlow als auch für die REST-API, welche den Betrieb in drei Modi erlaubt:

- Ungesichertes HTTP
- TLS, nur einseitig authentifiziert (Controller) und
- TLS, beidseitig authentifiziert

Lediglich bei beidseitig authentifiziertem TLS ist die Schnittstelle ausreichend geschützt. Im Modus 1 war ein Man-in-the-Middle-Angriff problemlos möglich, inklusive Manipulation und Mitschnitt von Daten. Modus 2 verhindert zwar Manipulation und Mitschnitt, jedoch nur wenn die aufrufende Seite das vom Controller präsentierte Zertifikat gegen eine CA überprüft. [Izar15]

3.3 BEACON

Ein weiterer OpenFlow-Controller auf Java-Basis ist BEACON (https://openflow.stanford.edu/display/Beacon/Home). [Stan13] Nach der Installation wird eine Weboberfläche angeboten, über welche alle im System verfügbaren Schnittstellen erreichbar sind und detaillierte Informationen wie MAC- und IP-Adressen der am Controller verbundenen Switche bekannten Geräte abgefragt werden können. Obgleich über die Oberfläche sehr viele Informationen (natürlich auch für potentielle Angreifer interessante) sichtbar sind, ist eine Absicherung nicht einstellbar. Eine Möglichkeit der Absicherung der Informationsseite mittels Drittanbietersoftware ist die Nutzung eines Reverse Proxy, beispielsweise mit nginx [Ngin09] oder Apache [Apac16]. Danach ist die Schnittstelle mit TLS abgesichert und die Benutzerauthentifizierung kann ebenfalls über zweiseitig authentifiziertes TLS oder über Benutzername und Passwort per HTTP-BASIC-AUTH erfolgen.

3.4 IRIS

Der OpenFlow-Controller IRIS (http://openiris.etri.re.kr) baut auf BEACON auf und bietet ähnliche Features an. [Iris] Nach der Installation ist auch hier ein Webinterface verfügbar. Im Gegensatz zu BEACON sind mehr Features verfügbar wie z.B. eine grafische Übersicht zur Visualisierung der Netzwerktopologie. Wie auch bei BEACON ist eine Absicherung nur über Drittanbietersoftware möglich.

3.5 OpenDaylight

OpenDaylight (https://www.opendaylight.org) wird von vielen IT-Firmen unterstützt. Mitglieder im OpenDaylight-Projekt sind z.B. Cisco, Citrix, Brocade, Dell, HP, Intel, Microsoft. OpenDaylight ist die Grundlage für einige kommerzielle SDN-Controller. Da diese kommerziell vertrieben werden und nicht frei zugänglich sind, beschränkt sich die Betrachtung auf die Open-Source Version von OpenDaylight. In der Dokumentation sind Empfehlungen zur Absicherung einiger Schnittstellen zu finden. Werden alle Empfehlungen berücksichtigt, ist der Controller gegen sehr viele Angriffe wie Man-in-the-Middle, Hijacking und Interception abgesichert. [EGK+]

3.5.1 Apache Karaf

OpenDaylight (in Version Lithium-SR3) nutzt Apache Karaf (https://karaf.apache.org) [Mats14], eine Laufzeitumgebung für Java-Anwendungen. [Kara] Apache Karaf bietet, neben einer Umgebung für Java-Applikationen, Funktionen zur Remotesteuerung und -konfiguration via SSH an. Authentifizierung erfolgt in der Standardkonfiguration über Benutzername und Passwort, die im Klartext hinterlegt sind, jedoch problemlos ausgelesen werden können. Voraussetzung dazu ist der Zugriff auf den Controller. Durch Aktivieren von Passwort-Hashing mit modernen Algorithmen wie SHA-2 lassen sich die Zugangsdaten schützen. Einen SSH-Login über Benutzername und Passwort kann man über einen Brute-Force-Angriff erfolgreich

attackieren. Aufgrund der Protokollierung von Zugriffen kann solch ein Angriff bspw. mit entsprechenden Tools wie fail2ban erkannt werden, um den Angreifer zu sperren. Zur Erhöhung der Sicherheit sollten SSH-Keys statt Passwörter Einsatz finden. Erfolgreiche Brute-Force-Angriffe sind dann nicht mehr möglich.

3.5.2 Schnittstellen: RESTconf und DLUX

Für den Zugriff von außen stellt OpenDaylight eine REST-ähnliche Schnittstelle (RESTconf) per Webserver bereit. Über den Server wird, neben der RESTconf-Schnittstelle, ein DLUX-Webinterface angeboten. Für die Nutzung beider Schnittstellen ist eine Authentifizierung erforderlich. Der erste erfolgreiche Angriff auf die RESTconf-Schnittstelle (geschützt per HTTP-BASIC-AUTH) war ein Brute-Force-Angriff auf den Login-Prozess. Auffällig war, dass scheinbar keine Groß- und Kleinschreibung für das Passwort berücksichtigt wird. So enthielt das angegriffene Passwort Großbuchstaben, es wurden aber auch Varianten mit nur Kleinbuchstaben akzeptiert (siehe Abbildung 3). Dadurch werden die Anzahl der Permutationen und damit die Zeit deutlich reduziert, da die Entropie stark verkleinert wird. Fehlgeschlagene Zugriffe werden nicht protokolliert. Somit ist es nicht möglich, Tools wie fail2ban einzusetzen und Angreifer zu sperren.

```
root@kali:/usr/share/wordlists# hydra -l admin -P rockyou.txt controller http-head -s 8181 /auth/v1/users
Hydra v8.1 (c) 2014 by van Hauser/THC - Please do not use in military or secret service organizations,
or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2016-01-11 12:32:38
[WARNING] http-head auth does not work with every server, better use http-get
[DATA] max 16 tasks per 1 server, overall 64 tasks, 14344404 login tries (l:1/p:14344404), ~14008 tries per task
[DATA] attacking service http-head on port 8181
[8181][http-head] host: controller   login: admin   password: projekt123
[8181][http-head] host: controller   login: admin   password: Projekt123
1 of 1 target successfully completed, 2 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2016-01-11 12:32:40
```

Abb. 3: BruteForce-Angriff auf RESTconf

Neben der RESTconf-Schnittstelle wird über den Webserver ein Webinterface (DLUX) bereitgestellt. Dieses liefert für Angreifer wertvolle Informationen über die Netzstruktur wie IP-/MAC-Adressen von Hosts und Netzwerkkomponenten und Netzwerkverbindungen. Wie auch die RESTconf-Schnittstelle wird das Webinterface unverschlüsselt via HTTP aufgerufen. Der mittels Man-in-the-Middle umgeleitete und mit Wireshark mitgeschnittene Datenverkehr zeigte allerdings keinen Login per HTTP-POST, sondern nur Anfragen über HTTP-GET. Der Grund dafür ist, dass die Login-Seite die Zugangsdaten für eine Anmeldung an der RESTconf-Schnittstelle nutzt, welche auch Quelle für die in DLUX dargestellten Daten wie Netzwerkaufbau mit Switchen, Maschinen etc. ist. Die Zugangsdaten werden daher per HTTP-BASIC-AUTH übertragen. Zwar sind diese zunächst per BASE64 kodiert, können aber sehr leicht wieder dekodiert werden und bieten keinen Schutz (siehe Abbildung 4).

Jedes Mal, wenn auf die RESTconf-API zugegriffen wird, werden Benutzername und Passwort an den Controller gesendet. Deshalb ist es nicht nötig, die Verbindung während der Anmeldung mitzuschneiden. Es genügt, zu einem beliebigen Zeitpunkt die Verbindung zu übernehmen. Als Gegenmaßnahme bietet OpenDaylight die Absicherung über TLS an. Wird die Webschnittstelle mit TLS gesichert, ist sowohl die RESTconf-API als auch DLUX über HTTPS erreichbar. Allerdings gibt es dann Probleme mit aktuellen Browsern, da die Cross-Origin-Policy verletzt wird: Calls auf die RESTconf-API werden weiterhin via HTTP abgewickelt, wodurch Dokumente nicht mehr von der gleichen Quelle geladen werden, wie das ursprünglich aufgerufene Dokument, da HTTPS einen anderen Port nutzt als HTTP (HTTPS: 443, HTTP: 80). Durch diesen Unterschied ändert sich die Quelle des Aufrufs (bestehend aus Host und Port einer URL),

mit dem Ergebnis, dass HTTP-Inhalte über eine per HTTPS aufgerufene Verbindung geblockt und nicht geladen werden. [Rude] Somit muss DLUX weiterhin über ungeschütztes HTTP aufgerufen werden, um nutzbar zu bleiben. Die RESTconf-Schnittstelle ist von den Einschränkungen nicht betroffen. Man-in-the-middle-Angriffe auf die TLS-Verbindung nutzen gefälschte Zertifikate, welche dem Original möglichst ähnlich sind, um bei manueller Prüfung möglichst nicht aufzufallen. Korrekte Konfiguration von Client und Server lassen Fälschungen erkennen, sodass Verbindungen unterbunden werden. Werden selbstsignierte Zertifikate eingesetzt, müssen die Daten vom Original bekannt oder gegen das Zertifikat der Zertifizierungsstelle abgeglichen werden, um das Imitat zu identifizieren. Dass eine Überprüfung der Zertifikate bei aktiviertem TLS wichtig ist, zeigte ebenfalls der Angriff auf die RESTconf-Schnittstelle. Wie beim Angriff auf DLUX wurde der Traffic zwischen Controller und aufrufender Maschine per Man-in-the-middle und gefälschtem TLS-Zertifikat kompromittiert und die Schnittstelle mit aktivierter und deaktivierter Zertifikatsprüfung aufgerufen. Nur bei aktivierter Zertifikatsprüfung wurde der Angriff erkannt, die Verbindung kam nicht zustande und übertragene Zugangsdaten wurden nicht abgefangen.

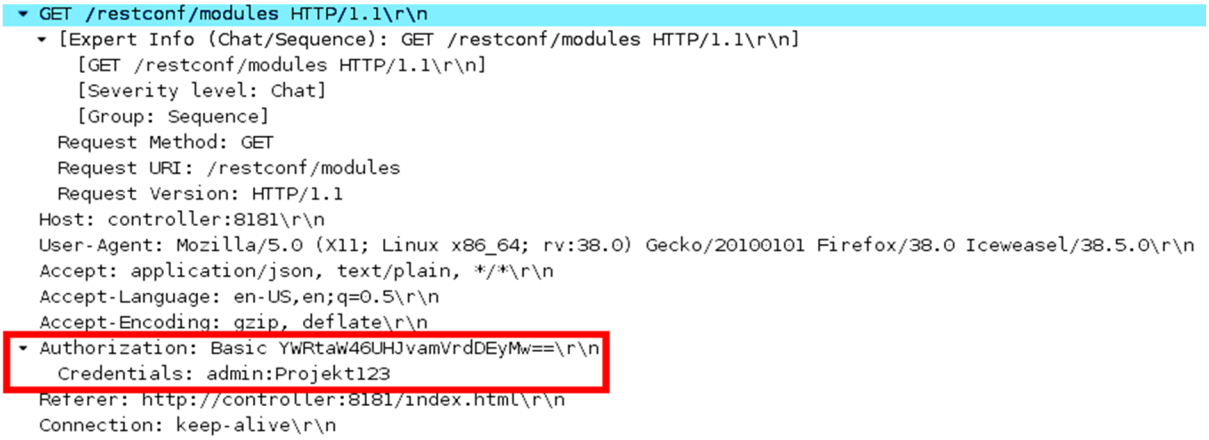

Abb. 4: Mit Wireshark mitgeschnittener DLUX-Login per HTTP-BASIC-AUTH

4 Fazit

Die Sicherheit von über OpenFlow gesteuerten SDN-Netzen hängt stark vom eingesetzten Controller und dessen Konfiguration ab. Obwohl die OpenFlow-Spezifikation den Einsatz von TLS empfiehlt, war keiner der untersuchten SDN-Controller standardmäßig entsprechend vorkonfiguriert. Bei BEACON und IRIS fehlt die TLS-Option gänzlich. Aus diesem Grunde eignen diese sich nicht für den Produktivbetrieb. Die meisten Controller bieten zudem Schnittstellen, welche die Steuerung von außen ermöglichen. Auch hier gibt es große Unterschiede, wie weit eine Absicherung des Zugriffs erfolgt. Während manche Controller eine Authentifizierung mit Benutzername und Passwort fordern, sind die Schnittstellen bei anderen komplett ungesichert. Auch diese Controller sollten nicht in kritischen Infrastrukturen eingesetzt werden. Am positivsten fielen dabei Floodlight und OpenDaylight auf, da hier Wert auf Sicherheit gelegt wird. Zwar sind die entsprechenden Einstellungen in der Grundinstallation nicht gesetzt, sind jedoch ausreichend dokumentiert und können entsprechend der Empfehlungen umgesetzt werden. Ein Grund dafür dürfte die starke Unterstützung durch große IT-Firmen sein, welche z.B. OpenDaylight als Grundlage für ihre kommerziellen Controller nutzen. Eine Übersicht über durchgeführte Angriffe und deren Gegenmaßnahmen zeigt Tabelle 1.

Tab. 1: Übersicht über Angriffsmöglichkeiten und Gegenmaßnahmen der betrachteten Controller

Produkt	Angriffsmöglichkeiten	Gegenmaßnahmen
POX	Manipulation über JSON-RPC-Schnittstelle/Messenger	keine
	Interception von OpenFlow	zweiseitige Authentifizierung mit TLS
Floodlight	Manipulation über REST-API	TLS (einseitige und zweiseitige Authentifizierung)
BEACON	Auslesen von Informationen über Webschnittstelle	keine, nur über Drittanbietertools (Reverse Proxy)
IRIS	siehe BEACON	siehe BEACON
OpenDaylight	Auslesen von SSH-Passwörtern auf dem Controller	Passwort-Hashing in der Konfiguration
	Brute-Force auf SSH	keine, nur über Drittanbietertools (fail2ban)
	Brute-Force auf RESTconf	keine
	Mitschneiden von Login-Daten (DLUX und RESTconf)	Aktivieren von TLS
	Man-in-the-middle auf TLS-Verbindung	Überprüfen des Zertifikats gegen Zertifikat der ausstellenden CA

Selbst wenn alle Maßnahmen zur Absicherung getroffen wurden, muss die Controller-Software aktuell gehalten werden, da auch hier Sicherheitslücken auftreten können. [Open]

Andere Aspekte, welche ebenfalls eine Rolle für die Sicherheit von SDN-Netzen spielen, lassen sich nicht von den Anbietern beeinflussen. Dies ist z.B. die Absicherung von Switchen gegen Angriffe aus dem Netzwerk und die Härtung der Maschine, auf welcher der Controller installiert wurde. Da die Controller teilweise auf kein bestimmtes Betriebssystem angepasst wurden, können hier keine konkreten Empfehlungen gegeben werden.

Literatur

[AlSh12] Ali Al-Shabibi, Getting Started with the Messenger Component, POX Wiki, 2012. https://openflow.stanford.edu/display/ONL/POX+Wiki#POXWiki-GettingStartedwiththeMessengerComponent

[Apac16] Apache, mod_proxy – Apache HTTP Server Version 2.4, 2016. https://httpd.apache.org/docs/current/mod/mod_proxy.html

[Ere15] Eren, E.: „SDN mit TLS absichern“; NET (Zeitschrift für Kommunikationsmanagement), ISSN 0947-4765, 10/2015

[FeRZ13] Nick Feamster, Jennifer Rexford, Ellen Zegura, The Road to SDN, acmqueue, 2013, https://queue.acm.org/detail.cfm?id=2560327

[Fou12] Open Networking Foundation. Software-defined networking: The new norm for networks, Technical report, April 2012, https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[Fou15a] Open Networking Foundation, Member Listing, 2015 https://www.opennetworking.org/our-members

[Fou15c] Open Networking Foundation https://www.opennetworking.org

[EGK+] OpenDaylight Security Analysis, OpenDaylight Project. https://wiki.opendaylight.org/view/CrossProject:OpenDaylight_Security_Analysis

[Hogg15] Is an SDN Switch A New Form of a Firewall?, NetworkWorld, 2015, http://www.networkworld.com/article/2905257/sdn/is-an-sdn-switch-a-new-form-of-a-firewall.html

[Iris] IRIS: The Recursive SDN-Openflow Controller by ETRI. http://openiris.etri.re.kr

[Izar15] Ryan Izard, OpenFlow and REST API Security Configuration, Project Floodlight,2015, https://floodlight.atlassian.net/wiki/display/floodlightcontroller/OpenFlow+and+REST+API+Security+Configuration

[Kara] Apache Karaf, Overview. https://karaf.apache.org/manual/latest/overview.html

[Mats14] Craig Matsumoto, How You'll Get Your OpenDaylight: Drink from the Karaf, 2014, https://www.sdxcentral.com/articles/news/youll-get-opendaylight-drink-karaf/2014/09/

[McDo09] Mindi McDowell, Understanding Denial-of-Service Attacks, United States Computer Emergency Readiness Team, 2009, https://www.us-cert.gov/ncas/tips/ST04-015

[McGi14] Shamus McGillicuddy, SDN security issues: How secure is the SDN stack?, techtarget, 2014, http://searchsdn.techtarget.com/news/2240214438/SDN-security-issues-How-secure-is-the-SDN-stack

[Ngin09] Nginx, Nginx, 2009. https://www.nginx.com/resources/admin-guide/reverse-proxy/

[Open] OpenDaylight Project, Security Advisories, OpenDaylight Project. https://wiki.opendaylight.org/view/Security_Advisories

[Rude] Mozilla Developer Network, Same-Origin-Policie, Mozilla Foundation. https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy

[Sand10] Chris Sanders, Understanding-Man-in-the-Middle-Attacks-ARP Cache Posisoning, WindowsSecurity.com, 2013, http://www.windowsecurity.com/articles-tutorials/authentication_and_encryption/Understanding-Man-in-the-Middle-Attacks-ARP-Part1.html

[Stan13] Stanford University, BEACON, 2013, https://openflow.stanford.edu/display/Beacon/Home

[SDxC] sdxcentral, SDN Security Challenges in SDN Environments, sdxcentral. https://www.sdxcentral.com/resources/security/security-challenges-sdn-software-defined-networks