APIC’s Adventures in Wonderland
APIC’s Adventures in Wonderland

Oliver Matula
omatula@ernw.de

Frank Block
fblock@ernw.de
APIC’s Adventures in ACI Wonderland

Oliver Matula
omatula@ernw.de

Frank Block
fblock@ernw.de

www.ernw.de
www.insinuator.net
@WEareTROOPERS
Agenda

Who is this APIC you are talking about?

You said you have found vulns! Where are they?

Ok, fine! But what can I take away from this?
Introduction
The What
Vulnerability Assessment of Cisco Application Centric Infrastructure (ACI)

The Why
Not much research has been done, since Cisco ACI is expensive
Lab Setup
Application Centric Infrastructure

Nexus 9k Spine Switches

Nexus 9k Leaf Switches

Different Systems

Hypervisor

Bare Metal Server

Database Server

ACI

APIC

ACI

ACI

ACI

ACI

ACI

APIC

APIC

APIC
Nexus 9k Leaf/Spine Switches
- Intel Xeon CPU (64 bit)
- Analyzed mainly Software Version 14.0(3d)
- Wind River Linux (kernel 3.14.62)
- ~300 processes, only two running as non-root user

Application Policy Infrastructure Controller (APIC)
- Intel Xeon CPU (64 bit)
- Analyzed mainly Software Version 14.1(1j)
- CentOS 7 Linux (kernel 4.14.104)
- ~500 processes, only ~20 running as non-root user
Vulnerability #1
Sinister secret backdoor found in networking gear perfect for government espionage: The Chinese are – oh no, wait, it's Cisco again

Better ban this gear from non-US core networks, right?

By Iain Thomson in San Francisco 2 May 2019 at 07:02

https://www.theregister.co.uk/2019/05/02/cisco_vulnerabilities/
Target (Nexus 9k Leaf Switch)

SSH Daemon #1
IPv4 0.0.0.0:22
IPv6 :::22

SSH Daemon #2
IPv4 127.0.0.1:1026
IPv6 :::1026

Attacker System

Mgmt Interface
Target (Nexus 9k Leaf Switch)

SSH Daemon #1
IPv4 0.0.0.0:22
IPv6 ::::22

SSH Daemon #2
IPv4 127.0.0.1:1026
IPv6 ::::1026

User 'local'
Public Key

Attacker System

Mgmt Interface
Target (Nexus 9k Leaf Switch)

SSH Daemon #1
IPv4 0.0.0.0:22
IPv6 :::22

SSH Daemon #2
IPv4 127.0.0.1:1026
IPv6 :::1026

Attacker copies private key

User 'local'
Public Key
Private Key

Attacker System

Attacker Controlled Nexus 9k Leaf Switch

Firmware Image
Target (Nexus 9k Leaf Switch)

SSH Daemon #1
IPv4 0.0.0.0:22
IPv6 :::22

SSH Daemon #2
IPv4 127.0.0.1:1026
IPv6 :::1026

User 'local'

Public Key
Private Key

Attacker System

Mgmt Interface

Public Key
Private Key
Target (Nexus 9K Leaf Switch)

SSH Daemon #1
IPv4 0.0.0.0:22
IPv6 :::22

SSH Daemon #2
IPv4 127.0.0.1:1026
IPv6 :::1026

User 'local'

Public Key

Private Key

ip6tables

Attacker System

Mgmt Interface

User 'local'

Public Key

Private Key
Target
(Nexus 9k Leaf Switch)

Mgmt Interface

SSH Daemon #1
IPv4 0.0.0.0:22
IPv6 :::22

SSH Daemon #2
IPv4 127.0.0.1:1026
IPv6 :::1026

User 'local'
Public Key
Private Key

Source port 1025 whitelisted for IPv6 traffic

ip6tables

Attacker System
```bash
echo 'test' > /volatile/file
```
Path Traversal Fails

echo 'test' > /tmp/file

```bash
echo 'test' > ../tmp/file
```

echo 'test' > bootflash:../tmp/file

echo 'test' > volatile:../tmp/file
Path Traversal Win

```bash
echo 'test' >
bootflash:lxc/CentOS7/rootfs/tmp/../../tmp/file
```

/bootflash/lxc/CentOS7/rootfs/tmp
is symbolic link to /var/volatile/tmp
Path Traversal Win

echo 'test' >

bootflash:/lxc/CentOS7/rootfs/tmp/../../../tmp/file

/bootflash:/lxc/CentOS7/rootfs/tmp

is symbolic link to /var/volatile/tmp

Can write arbitrary files with arbitrary content as user 'local' (CVE-2019-1836)
Cron job /bin/bg-action.sh run by root once per min.

- bg-action.sh
  - exec
    - setup-hwclock.sh
      - filter
        - Execute constructed command

- /tmp/setup-hwclock
  - exec if exists
    - take content
    - /tmp/setup-hwclock
Cron job /bin/bg-action.sh run by root once per min.

```
/bin/bg-action.sh
exec
/tmp/setup-hwclock
```

```
/setup-hwclock.sh
filter
Execute constructed command
```

Filters ";", "||", "&&", but not "$()$"
Cron job /bin/bg-action.sh run by root once per min.

Filters ";", "||", "&&", but not "$(())"

Can run arbitrary commands as root user via /tmp/setup-hwclock file (CVE-2019-1803)
Finally, chain vulnerabilities to

1. Upload reverse shell
2. Execute reverse shell as root
Finally, chain vulnerabilities to

1. Upload reverse shell
2. Execute reverse shell as root

Demo time!
Vulnerability #2
Spine

Network Cable

Infrastructure VLAN

VLAN / VXLAN
for isolation mechanism

Transition
VLAN ↔ VXLAN
VLAN IDs ↔ VNIs

Leaf

Server
System #1

Server
System #2

Leaf

Attacker
Controlled
System

Black Hat USA 2019
Network Communication

**APIC**

1. LLDP Broadcast
2. DHCP
3. TCP SYN
4. TCP RST, ACK
5. HTTP
   - GET /fwrepo/boot/node-FD022480FLU
6. TLSv1.2
   - Client Hello

**Leaf Switch**

1. LLDP Broadcast
2. DHCP
3. TCP SYN
4. TCP RST, ACK
5. HTTP
   - GET /fwrepo/boot/node-FD022480FLU
6. TCP port 12183

- nginx
  - TCP port 7777

- svc_ifc_policyelem
  - TCP port 12183
  - TCP conn. to port 12183 again

- svc_ifc_policyelem
  - TCP port 12183
**Attack Scenario**

- **Network Cable**
- **Infrastructure VLAN**
- **VLAN / VXLAN**
  for isolation mechanism

---

**Transition**

VLAN ↔ VXLAN
VLAN IDs ↔ VNIs

---

**Interesting information!**

---

**Leaf**

---

**Server**

System #1

---

**Spine**

---

**Leaf**

---

**Server**

System #2

---

**Attacker Controlled System**

---

**LLDP Broadcast**
Psst, Leaf. I'm also an APIC.
CVE-2019-1890
Attacker controlled system can join infra VLAN and access internal services!

- APIC ~60, Leaf & Spine ~15 services on infra VLAN
- VXLAN tunnel endpoints exposed
- Services on management interface also exposed
CVE-2019-1890
Attacker controlled system can join infra VLAN and access internal services!

- APIC ~60, Leaf & Spine ~15 services on infra VLAN
- VXLAN tunnel endpoints exposed
- Services on management interface also exposed

Demo time!
Going down the rabbit hole
The image illustrates a network diagram with the following components:

- **Spine**: Central network component.
- **Leaf**: Edge network components.
- **VXLAN Endpoint**: Network endpoint using VXLAN for isolation.
- **Network Cable**: Connection between components.
- **Infrastructure VLAN**: VLAN for infrastructure.
- **VLAN / VXLAN**: VLAN or VXLAN for isolation mechanism.

The diagram shows the connectivity between the Spine and Leaf with running VXLAN Endpoint, and the interaction with Server System #1 and #2, indicating a controlled system and an attacker system.
Crafting a VXLAN Packet

<table>
<thead>
<tr>
<th>Layer</th>
<th>Protocol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7….</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>7.3</td>
<td>IP</td>
<td>src = 192.168.200.11, dst = 192.168.200.20</td>
</tr>
<tr>
<td>7</td>
<td>VXLAN</td>
<td>vni = target VNI</td>
</tr>
<tr>
<td>4</td>
<td>UDP</td>
<td>dst = VXLAN Endpoint</td>
</tr>
<tr>
<td>3</td>
<td>IP</td>
<td>dst = Address of Leaf</td>
</tr>
</tbody>
</table>
Psst, Leaf. Here is a packet that I want to inject.
Psst, Leaf. Here is a packet that I want to inject.
Network Cable
Infrastructure VLAN
VLAN / VXLAN for isolation mechanism

Leaf with running VXLAN Endpoint
Server System #1
APIC

Spine

Server System #2

Leaf with running VXLAN Endpoint

Attacker Controlled System

Demo time!
Vulnerability #3
LLDP is an OSI-layer 2 protocol using Type-Length-Value Structures.

### TLV Type Values

<table>
<thead>
<tr>
<th>TLV Type</th>
<th>TLV Names</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>End of LLDP Data Unit</td>
<td>Mandatory</td>
</tr>
<tr>
<td>1</td>
<td>Chassis ID</td>
<td>Mandatory</td>
</tr>
<tr>
<td>2</td>
<td>Port ID</td>
<td>Mandatory</td>
</tr>
<tr>
<td>3</td>
<td>Time To Live</td>
<td>Mandatory</td>
</tr>
<tr>
<td>4</td>
<td>Port Description</td>
<td>Optional</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>127</td>
<td>Custom TLVs</td>
<td>Optional</td>
</tr>
</tbody>
</table>
Cisco Systems, Inc - ACI Unknown-D8: 00:00

1111 111. ..... .... = TLV Type: Organization Specific (127)
.... ...0 0000 0110 = TLV Length: 6
Organization Unique Code: 00:01:42 (Cisco Systems, Inc)
Cisco Subtype: ACI Unknown-D8 (0xd8)
Unknown 0xD8: 0000

| 00f0 | 05 00 00 00 00 fe 05 00 01 42 01 01 fe 06 00 01 |
| 0100 | 42 d8 00 00 fe 05 00 01 42 c9 01 fe 0f 00 01 42 |
LLDP Buffer Overflow

• LLDP running as root on all leafs and spines.
• NX and PIE activated.

• What happens when the length value for subtype 0xd8 is modified?
LLDP Buffer Overflow

- LLDP running as root on all leafs and spines.
- NX and PIE activated.

- What happens when the length value for subtype 0xd8 is modified?

CVE-2019-1901
Psst, Leaf. Here is a new device.

Malicious LLDP Broadcast

Attacker Controlled System
### Remaining Problems

<table>
<thead>
<tr>
<th>Command</th>
<th>Output</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leaf1# fgrep libc-</td>
<td>/proc/14432/maps</td>
<td>/lib/libc-2.15.so</td>
</tr>
<tr>
<td>eea77000 - eeclc000</td>
<td>r-xp 00000000   00:0e 39765</td>
<td></td>
</tr>
<tr>
<td>eec20000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>eea41000 - eebe6000</td>
<td>r-xp 00000000   00:0e 39765</td>
<td></td>
</tr>
<tr>
<td>eebe7000 - eebe9000</td>
<td>r-xp 001a5000   00:0e 39765</td>
<td></td>
</tr>
<tr>
<td>eebe9000 - eebea000</td>
<td>r-xp 001a7000   00:0e 39765</td>
<td></td>
</tr>
</tbody>
</table>

---

# BHUSA      @BLACK HAT EVENTS
Remaining Problems

Leaf1# fgrep libc- /proc/14432/maps
eea77000-eec1c000 r-xp 00000000 00:0e 39765 /lib/libc-2.15.so
eec1c000-eec1d000 ---p 001a5000 00:0e 39765 /lib/libc-2.15.so
eec1d000-eec1f000 r-xp 001a5000 00:0e 39765 /lib/libc-2.15.so

[root@apic1 ~]# acidiag fvnread

<table>
<thead>
<tr>
<th>ID</th>
<th>Pod ID</th>
<th>Name</th>
<th>Serial Number</th>
<th>IP Address</th>
<th>Role</th>
<th>State</th>
<th>LastUpdMsgId</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>1</td>
<td>Leaf1</td>
<td>FD022480FLU</td>
<td>/32</td>
<td>leaf</td>
<td>inactive</td>
<td>0x20000000cde97c</td>
</tr>
<tr>
<td>102</td>
<td>1</td>
<td>Spine</td>
<td>FD022472FAZ</td>
<td>/32</td>
<td>spine</td>
<td>active</td>
<td>0</td>
</tr>
<tr>
<td>103</td>
<td>1</td>
<td>Leaf2</td>
<td>FD022480FHY</td>
<td>/32</td>
<td>leaf</td>
<td>active</td>
<td>0</td>
</tr>
</tbody>
</table>

Total 3 nodes

eed93000-eeb94000 ---p 001a5000 00:0e 42419 /lib/libc-2.15.so
eed94000-eeb96000 r--p 001a5000 00:0e 42419 /lib/libc-2.15.so
eed96000-eeb97000 rw-p 001a7000 00:0e 42419 /lib/libc-2.15.so
# Remaining Problems

Leaff1# fgrep libc- /proc/14432/maps
eea77000-eec1c000 r-xp 00000000 00:0e 39765 /lib/libc-2.15.so
eec1c000-eec1d000 ---p 001a5000 00:0e 39765 /lib/libc-2.15.so

[root@apici ~]# acidag invread
<table>
<thead>
<tr>
<th>ID</th>
<th>Pod ID</th>
<th>Name</th>
<th>Seriet Number</th>
<th>IP</th>
<th>Role</th>
<th>State</th>
<th>LastUpdate</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>1</td>
<td>Leaf1</td>
<td>FD022480FLU</td>
<td>/32</td>
<td>leaf</td>
<td>inactive</td>
<td>0x20000000</td>
</tr>
<tr>
<td>102</td>
<td>1</td>
<td>Spine</td>
<td>FD022472FA</td>
<td>/32</td>
<td>spine</td>
<td>active</td>
<td>0</td>
</tr>
<tr>
<td>103</td>
<td>1</td>
<td>Leaf2</td>
<td>FD022471</td>
<td>/32</td>
<td>leaf</td>
<td>active</td>
<td>0</td>
</tr>
</tbody>
</table>

Leaff1# fgrep libc- /proc/13562/maps
eec90000-eeb93000 r-xp 00000000 00:0e 42419 /lib/libc-2.15.so

Total 39 nodes
Going further down the rabbit hole
Going further down the rabbit hole

```
Leaf2# ifconfig -a 2>&1 | egrep '^[^ \t]'
eth0   Link encap:Ethernet  HWaddr
inband_hi Link encap:Ethernet  HWaddr
inband_lo Link encap:Ethernet  HWaddr
kpm_inb  Link encap:Ethernet  HWaddr
kpm_mgmt Link encap:Ethernet  HWaddr
lo      Link encap:Local Loopback
mgmt0   Link encap:Ethernet  HWaddr
psdev0  Link encap:Ethernet  HWaddr
psdev1  Link encap:Ethernet  HWaddr
psdev2  Link encap:Ethernet  HWaddr
```
Going further down the rabbit hole
net_l2_register(socket_fd, 1, &a3, &ethertype.type, 1, 0)

net_l2_send(socket_fd, &a2, &intf.if_id, pstruct.padding, l2_message_length, l2_frame.dst_address)

struct l2_frame {
    char dst_address[6];
    char src_address[6];
    char ethertype[2];
    char msg[payload_length];
};
Psst, Spine. Here is a new device.
Attack Scenario

Network Cable

Leaf

Server System #1

Server System #2

Malicious LLDP Broadcast

Spine
Psst, Server #2. Here is Server #1.
Attack Scenario

- Network Cable
- Infrastructure VLAN
- VLAN / VXLAN A
- VLAN / VXLAN B

Demo time!
Vulnerability #4
Going down the APIC hole

Packages

Quick Start

Summary

The Packages menu allows you to import L4-L7 device packages, which are used to define, configure, and monitor a network service device such as a firewall, SSL offload, load balancer, context switch, SSL termination device, or intrusion prevention system (IPS). Device packages contain descriptions of the functional capability and settings along with interfaces and...
Going down the APIC hole

```
surf@machine /tmp % unzip -l asa-device-pkg-1.0.1.zip | tail
  0 2019-05-08 18:46  utils/
  420 2014-07-28 15:05  utils/env.py
  97 2014-07-28 15:05  utils/__init__.py
  844 2014-07-28 15:05  utils/errors.py
  4979 2014-07-28 15:05  utils/service.py
  22462 2014-07-28 15:05  utils/util.py
  939 2014-07-28 15:05  utils/protocol.py
  581 2019-05-08 18:45  ../../../../../../etc/cron.d/ernw_cronjob
----------------- -----------------
  581500          68 files
```
**Going down the APIC hole**

```
surf@machine /tmp % unzip -l asa-device-pkg-1.0.1.zip | tail

<table>
<thead>
<tr>
<th></th>
<th>Date</th>
<th>Time</th>
<th>Size</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2019-05-08</td>
<td>18:46</td>
<td>0</td>
<td>utilities/</td>
</tr>
<tr>
<td>420</td>
<td>2014-07-28</td>
<td>15:05</td>
<td>420</td>
<td>utilities/env.py</td>
</tr>
<tr>
<td>97</td>
<td>2014-07-28</td>
<td>15:05</td>
<td>97</td>
<td>utilities/<strong>init</strong>.py</td>
</tr>
<tr>
<td>844</td>
<td>2014-07-28</td>
<td>15:05</td>
<td>844</td>
<td>utilities/errors.py</td>
</tr>
<tr>
<td>4979</td>
<td>2014-07-28</td>
<td>15:05</td>
<td>4979</td>
<td>utilities/service.py</td>
</tr>
<tr>
<td>22462</td>
<td>2014-07-28</td>
<td>15:05</td>
<td>22462</td>
<td>utilities/util.py</td>
</tr>
<tr>
<td>939</td>
<td>2014-07-28</td>
<td>15:05</td>
<td>939</td>
<td>utilities/protocol.py</td>
</tr>
<tr>
<td>581</td>
<td>2019-05-08</td>
<td>18:45</td>
<td></td>
<td>/etc/cron.d/ernw cronjob</td>
</tr>
</tbody>
</table>

581500 68 files
```

**CVE-2019-1889**
• Update immediately!
• Watch out for new Updates.
• Think about how to use your ACI fabric.
• Restrict Access to the management interfaces.
• Deactivate LLDP wherever it is not necessary.
• Do not import Device Packages from Spam/4chan/Stackoverflow!
Thanks for your Attention!

Questions?

See Whitepaper and exploit files for more details!
Security Advisories

• Vulnerability #1
  • Cisco Nexus 9000 Series Fabric Switches Application Centric Infrastructure Mode Symbolic Link Path Traversal Vulnerability

• Cisco Nexus 9000 Series Fabric Switches Application Centric Infrastructure Mode Root Privilege Escalation Vulnerability

• Cisco Nexus 9000 Series Fabric Switches Application Centric Infrastructure Mode Default SSH Key Vulnerability
Security Advisories

- **Vulnerability #2**
  - Cisco Nexus 9000 Series Fabric Switches ACI Mode Fabric Infrastructure VLAN Unauthorized Access Vulnerability (High)
    - CVE-2019-1890
      - https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20190703-n9kaci-bypass

- **Vulnerability #3**
  - Cisco Nexus 9000 Series Fabric Switches Application Centric Infrastructure Mode Link Layer Discovery Protocol Buffer Overflow Vulnerability
    - CVE-2019-1901
      - https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20190731-nxos-bo
Security Advisories

- Vulnerability #4
  - Cisco Application Policy Infrastructure Controller REST API Privilege Escalation Vulnerability
  - CVE-2019-1889
    https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20190703-ccapic-restapi