RSAC Sandbox

RSAConference2020

HUMAN ELEMENT

SESSION ID: **SBX1-W1**

Mitigating Network-Based Attacks Using MUD

Improving Security of Small-Business and Home IoT Devices

Practical Use of the MUD Specification

Dr. Parisa Grayeli

Principal Information Systems Engineer
MITRE/NIST National Cybersecurity Center of Excellence (NCCoE)

Blaine Mulugeta

Cyber Engineer
MITRE/NIST National Cybersecurity Center of Excellence (NCCoE)

NIST NCCoE's Mission

Accelerate adoption of secure technologies: Collaborate with innovators to provide real-world, standards-based cybersecurity capabilities that address business needs

Engagement and Business Model

DEFINE

OUTCOME:
Define a scope of work with industry to solve a pressing cybersecurity challenge

ASSEMBLE

OUTCOME:
Assemble teams of industry orgs, govt agencies, and academic institutions to address all aspects of the cybersecurity challenge

BUILD

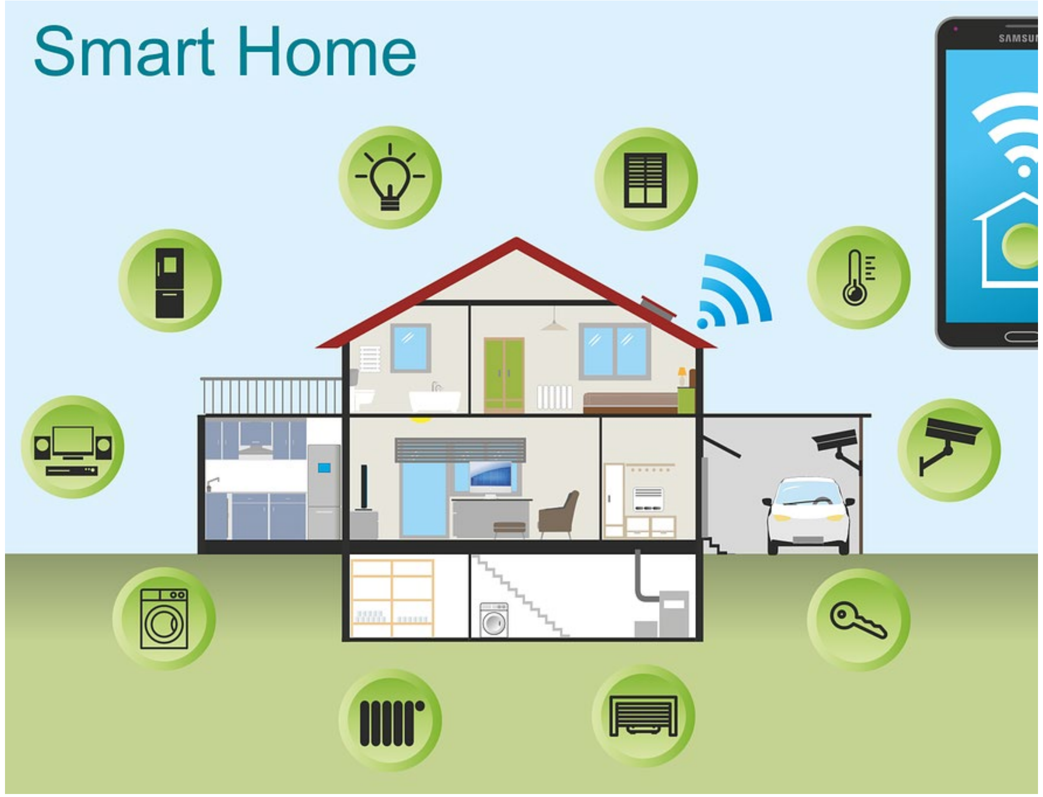
OUTCOME:
Build a practical, usable, repeatable implementation to address the cybersecurity challenge

ADVOCATE

OUTCOME:
Advocate adoption of the example implementation by using the practice guide

Introduction

- There will be 25 billion connected things in use by 2021 (per Gartner)
- As IoT devices become more common in homes and businesses, security concerns are also increasing
- IoT devices represent one of the largest attack surfaces – Some have minimal security, are unprotected or are difficult to secure
- IoT devices have been exploited to launch DDoS attacks (e.g. Mirai)



Mitigating Network-Based Attacks Using MUD

Improving the security of small-business and home IoT devices

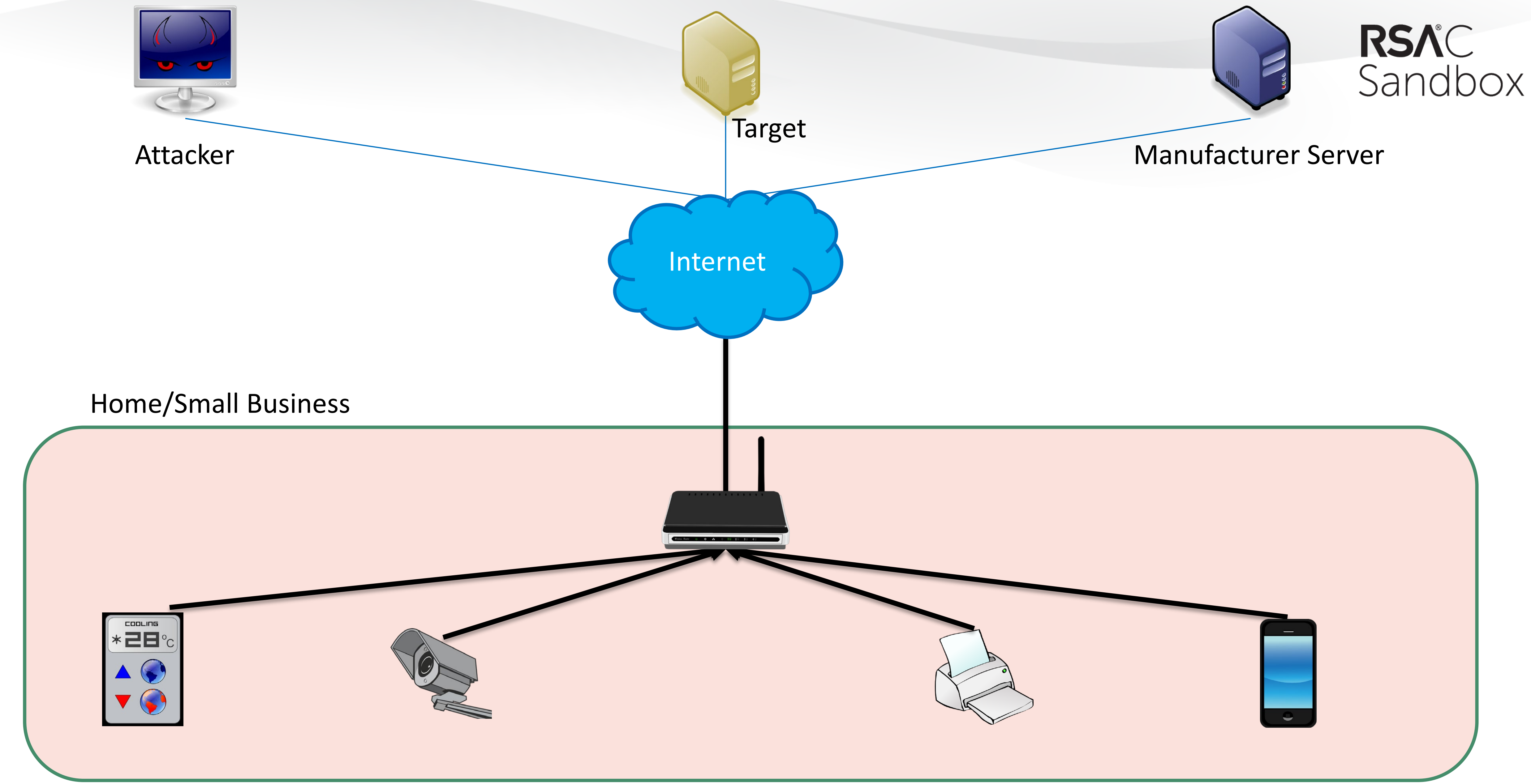
Challenge

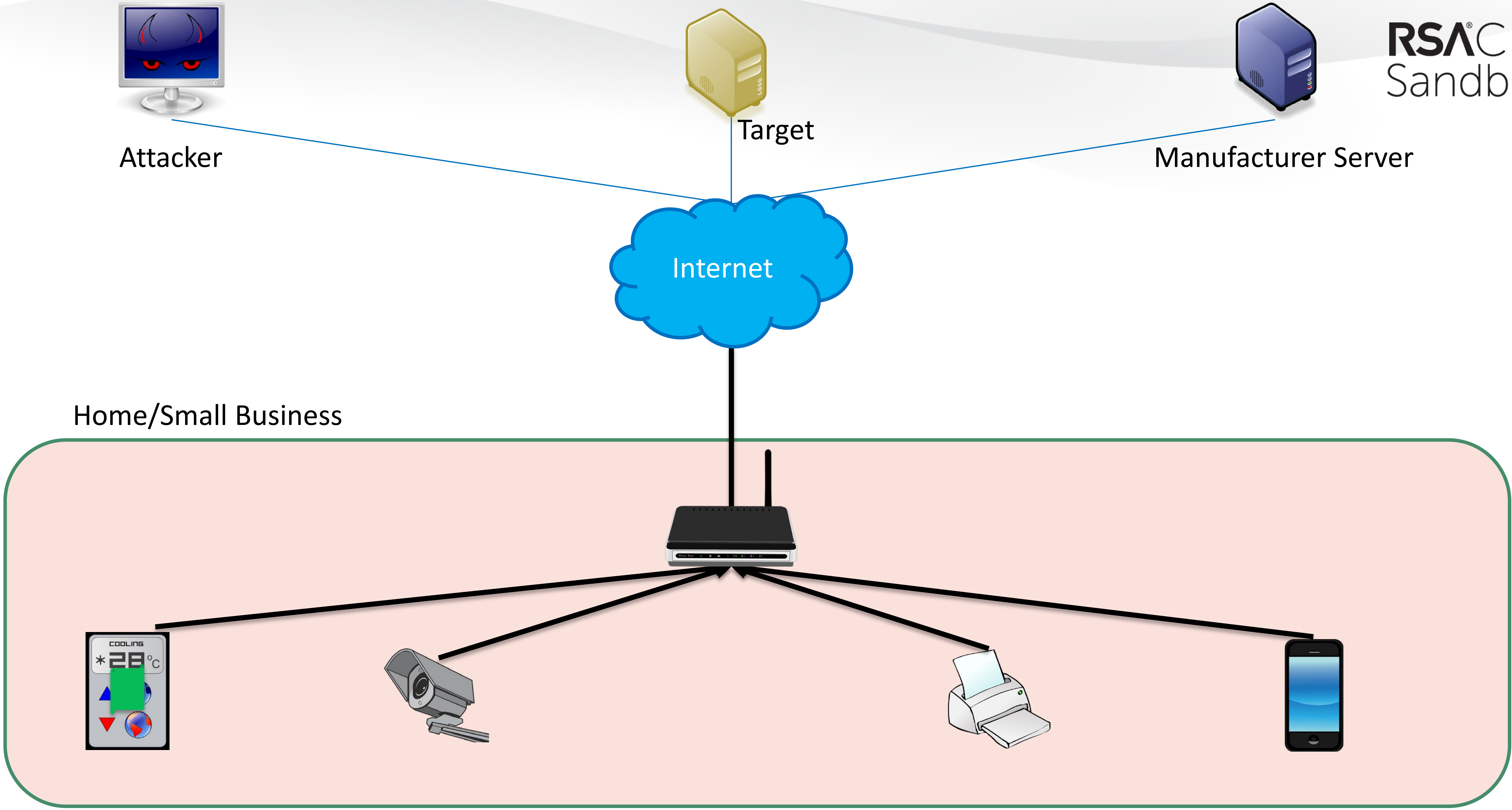
- IoT devices are given full connectivity to the internet by default
- Device security may not be a priority due to processing, timing, memory, and power constraints
- Networked devices can be detected within minutes and exploited due to known security flaws, leading to easily scalable attacks

Solution

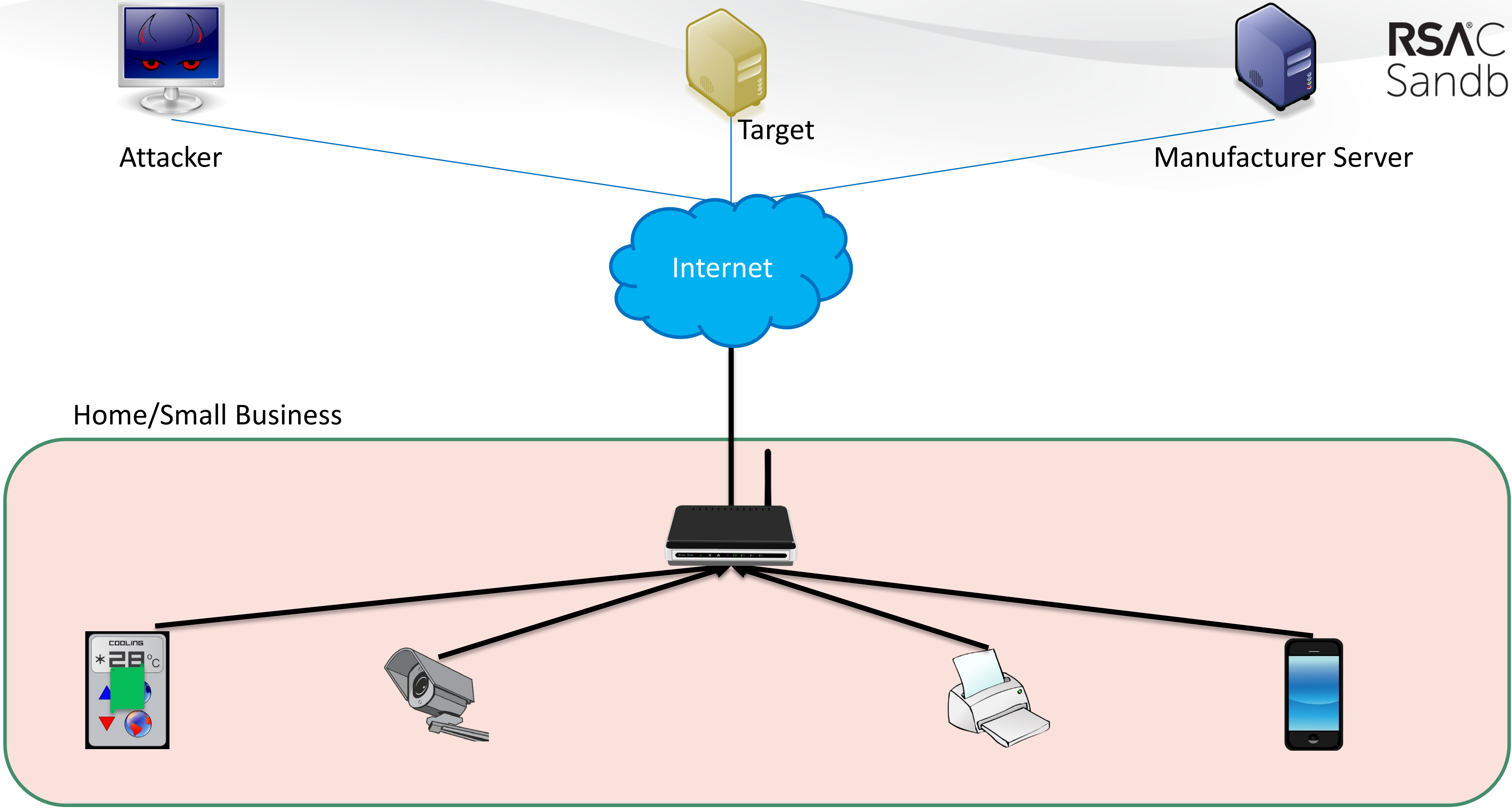
- NCCoE developed a proof of concept implementation for the home or small business network to address some of these security concerns in collaboration with Industry collaborators
- Use network gateway components and security-aware IoT devices that leverage the Manufacturer Usage Description (MUD) Specification (RFC 8520)
- Using MUD the network will automatically permit the IoT device to send and receive the traffic it requires to perform as intended, and the network will prohibit all other communication with device

Typical Home/Small Business Network (Without MUD)


Attacker
Target
Manufacturer Server
RSAC Sandbox
Internet
Home/Small Business
COOLING
28°C



RSAC Sandbox



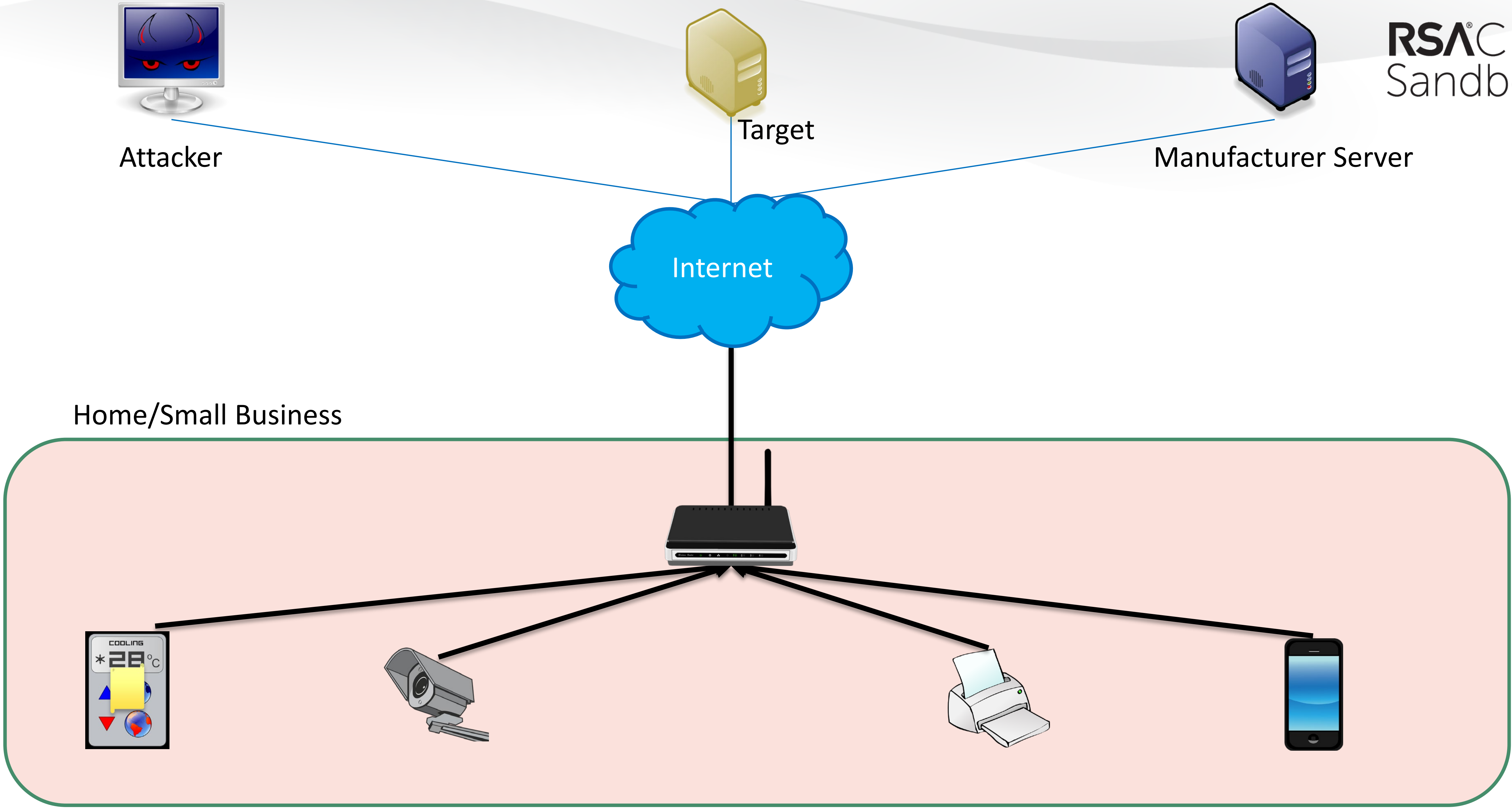
Attacker

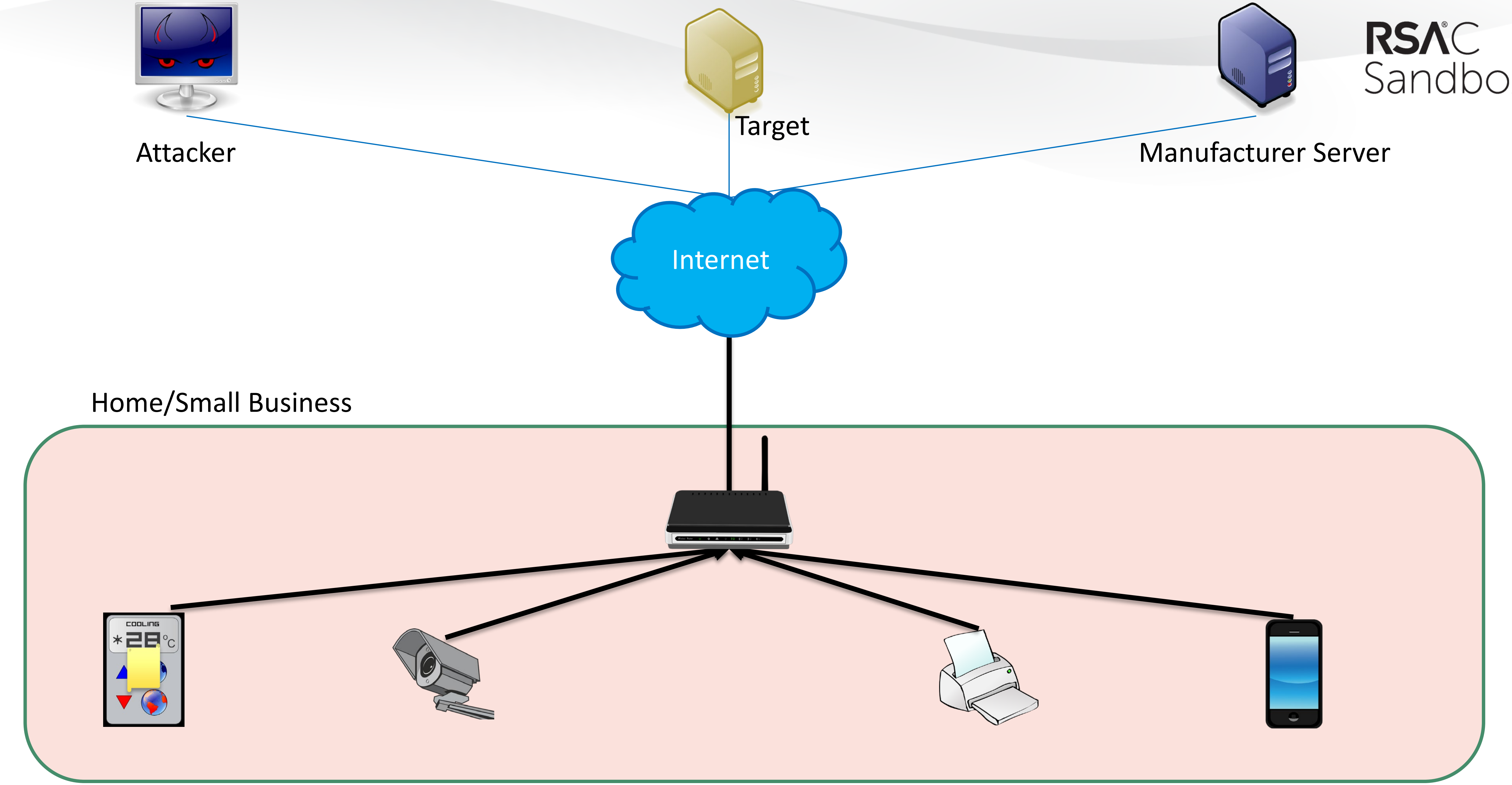
Target

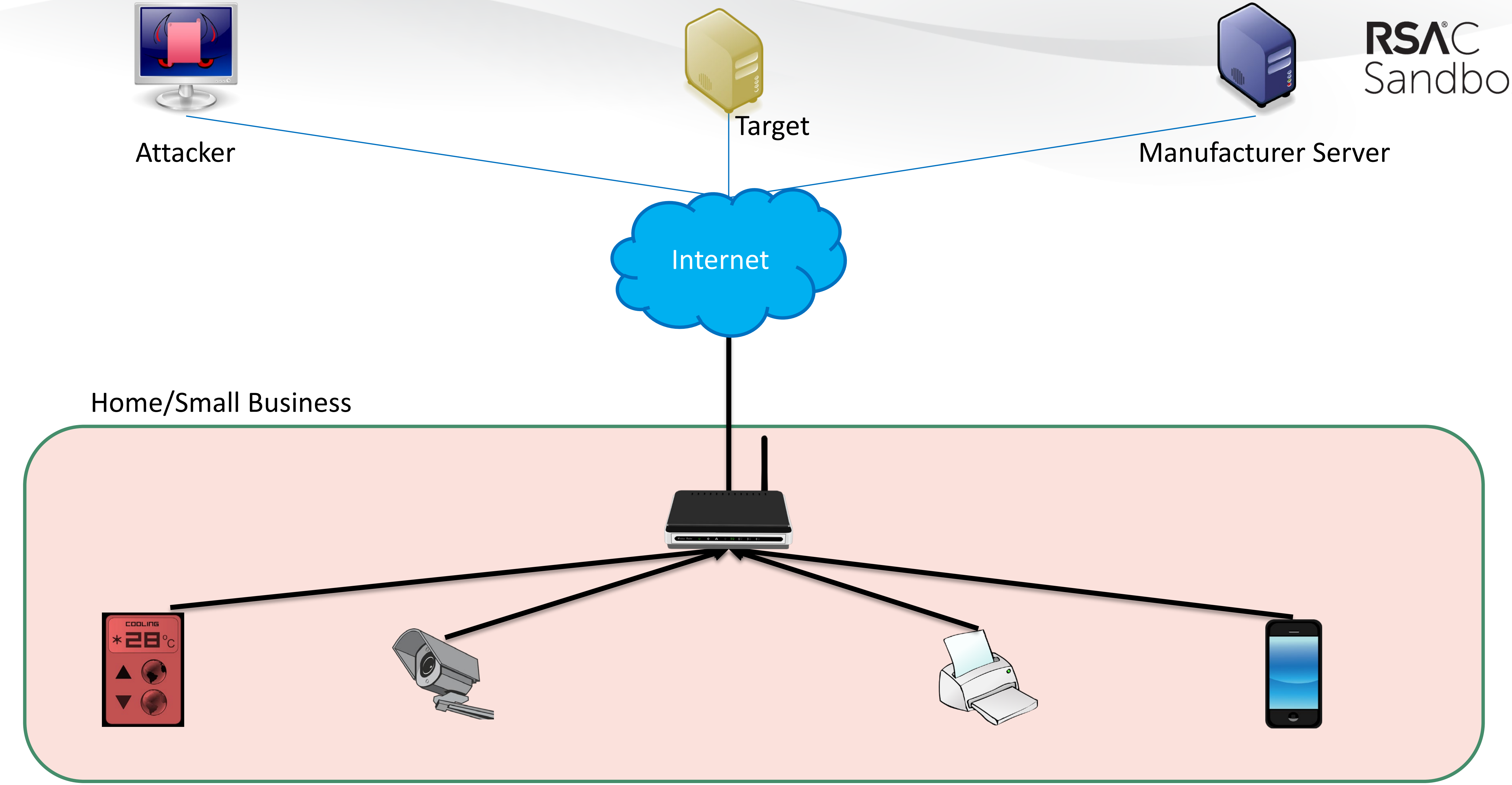
Manufacturer Server

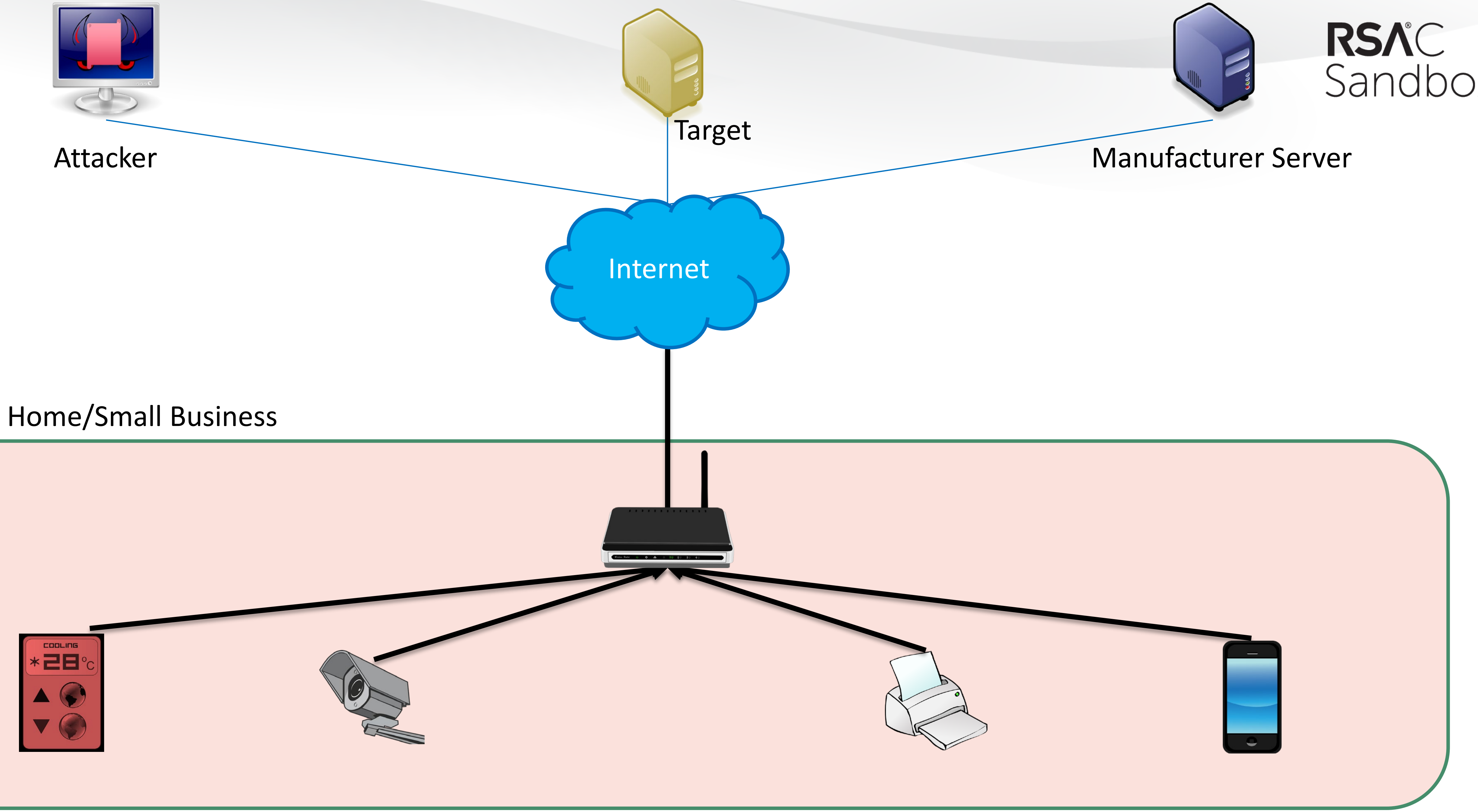
Internet

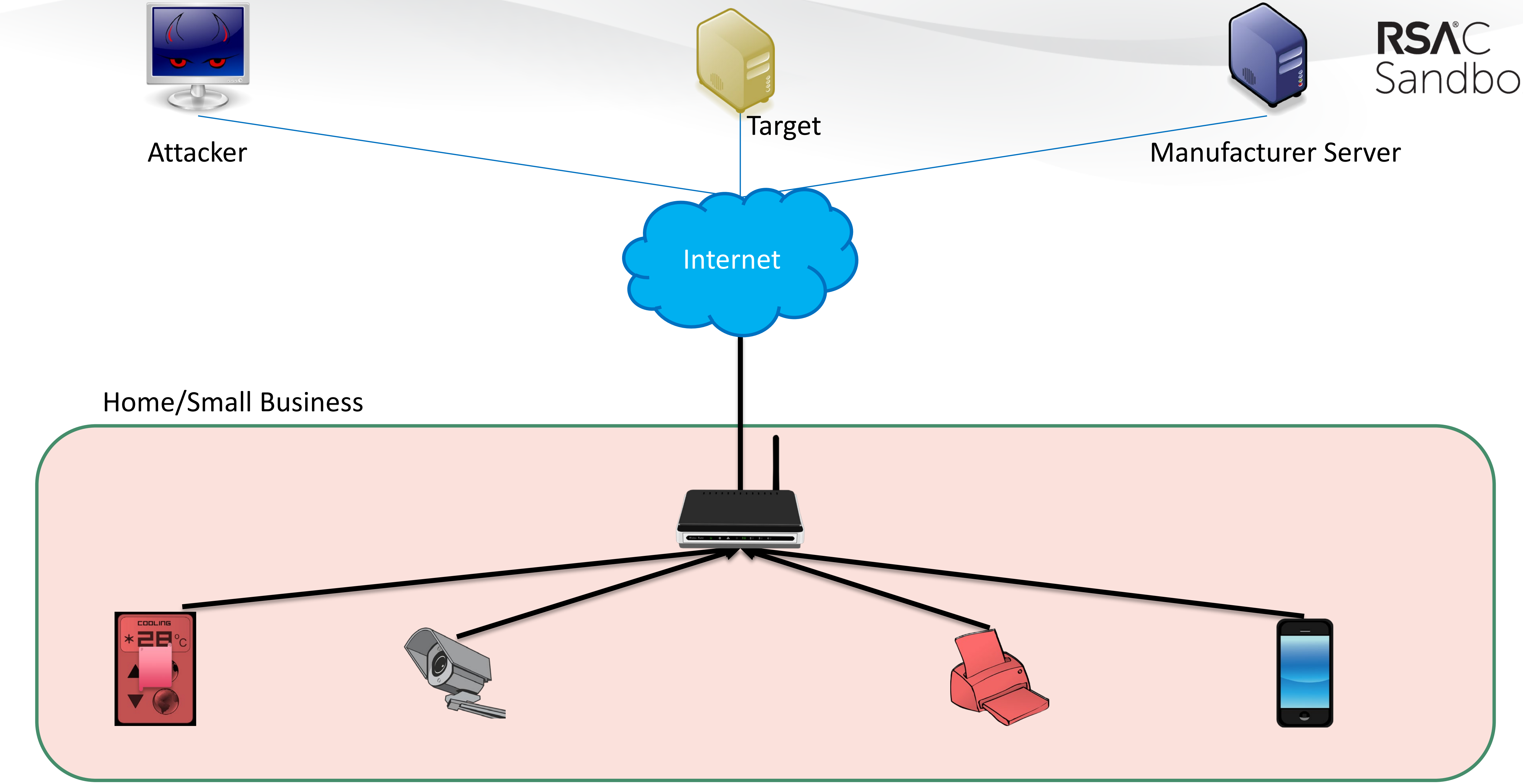
Home/Small Business

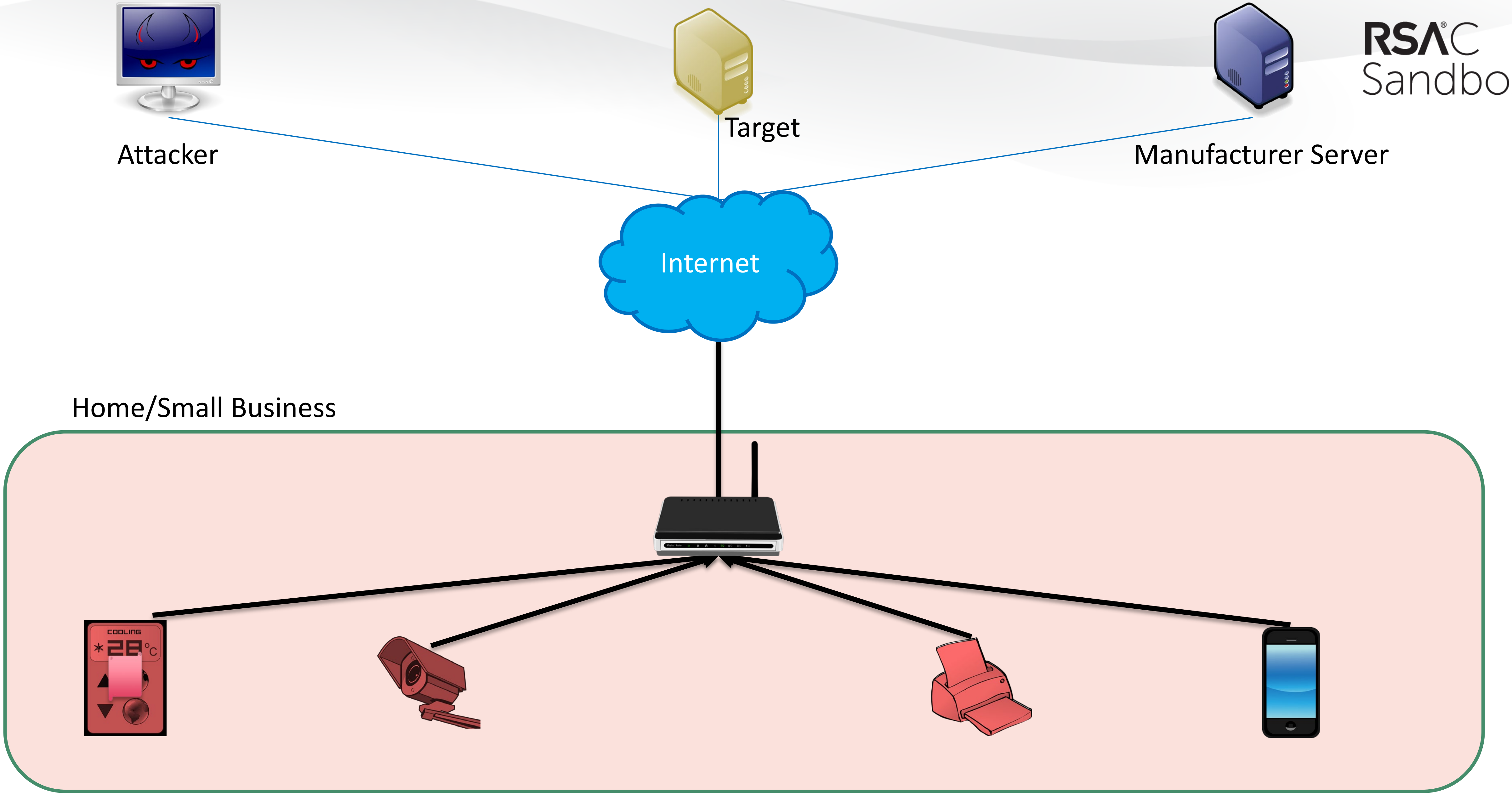

Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C

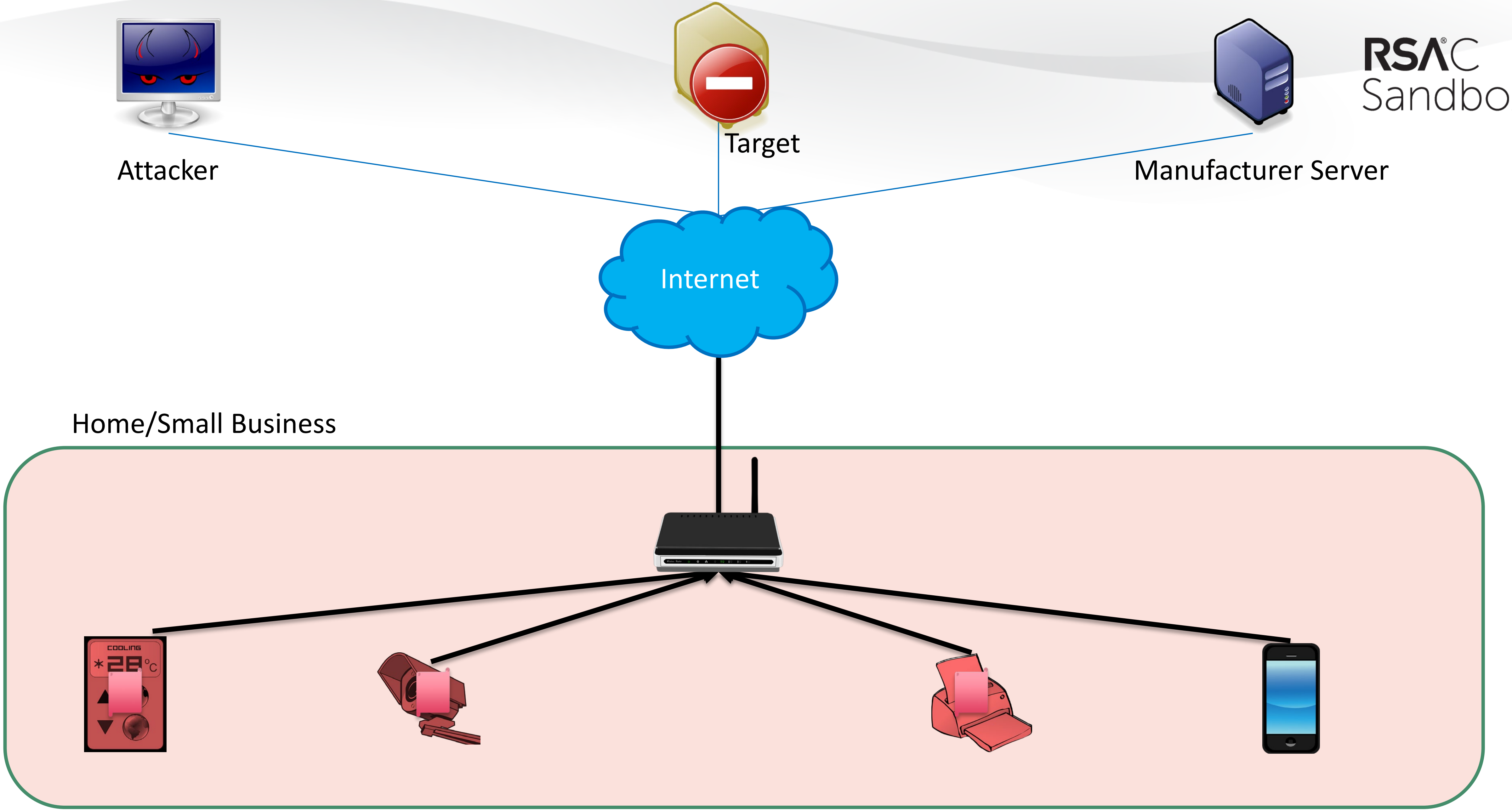

Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C


Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C


Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C


Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C


Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C



Attacker

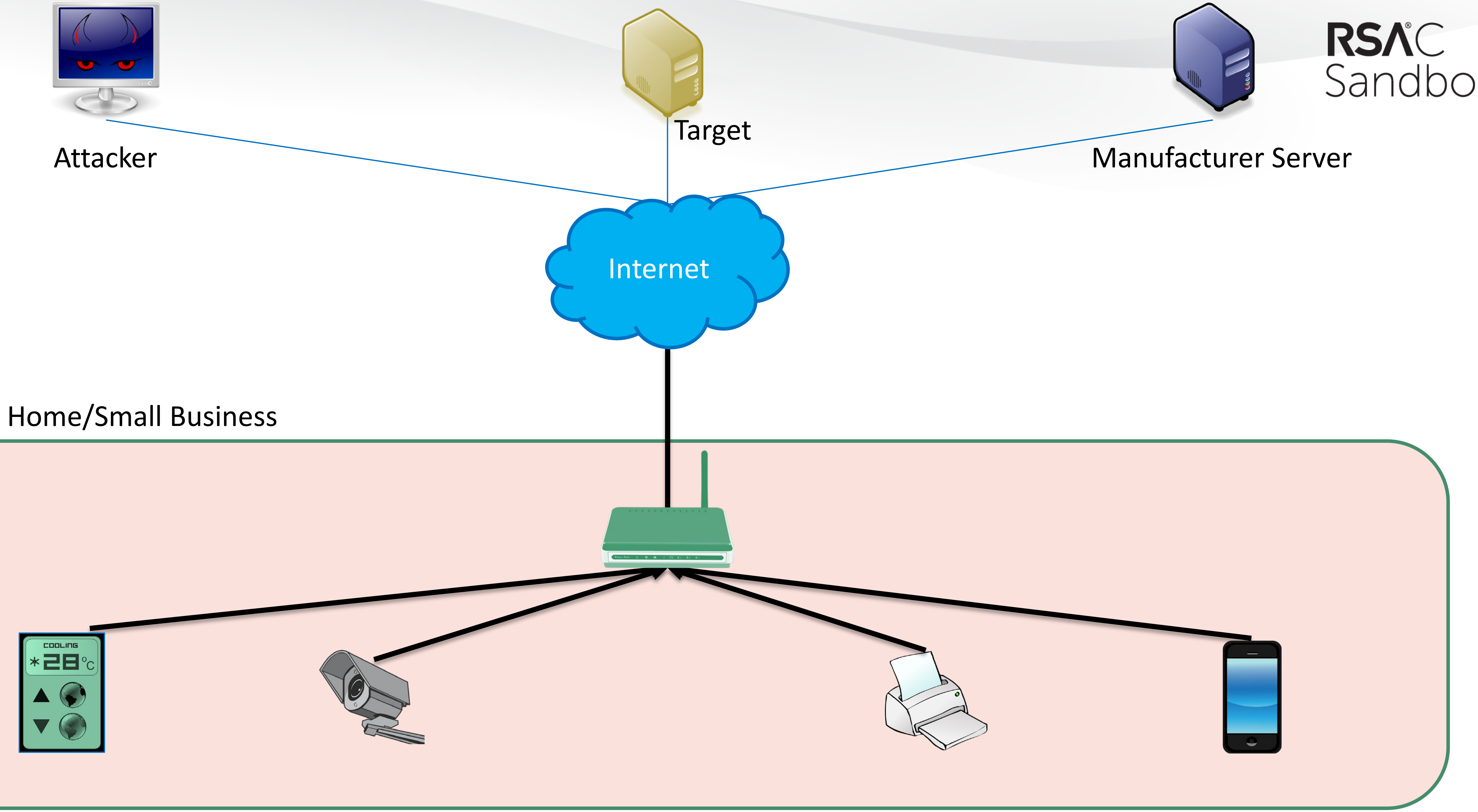
Target

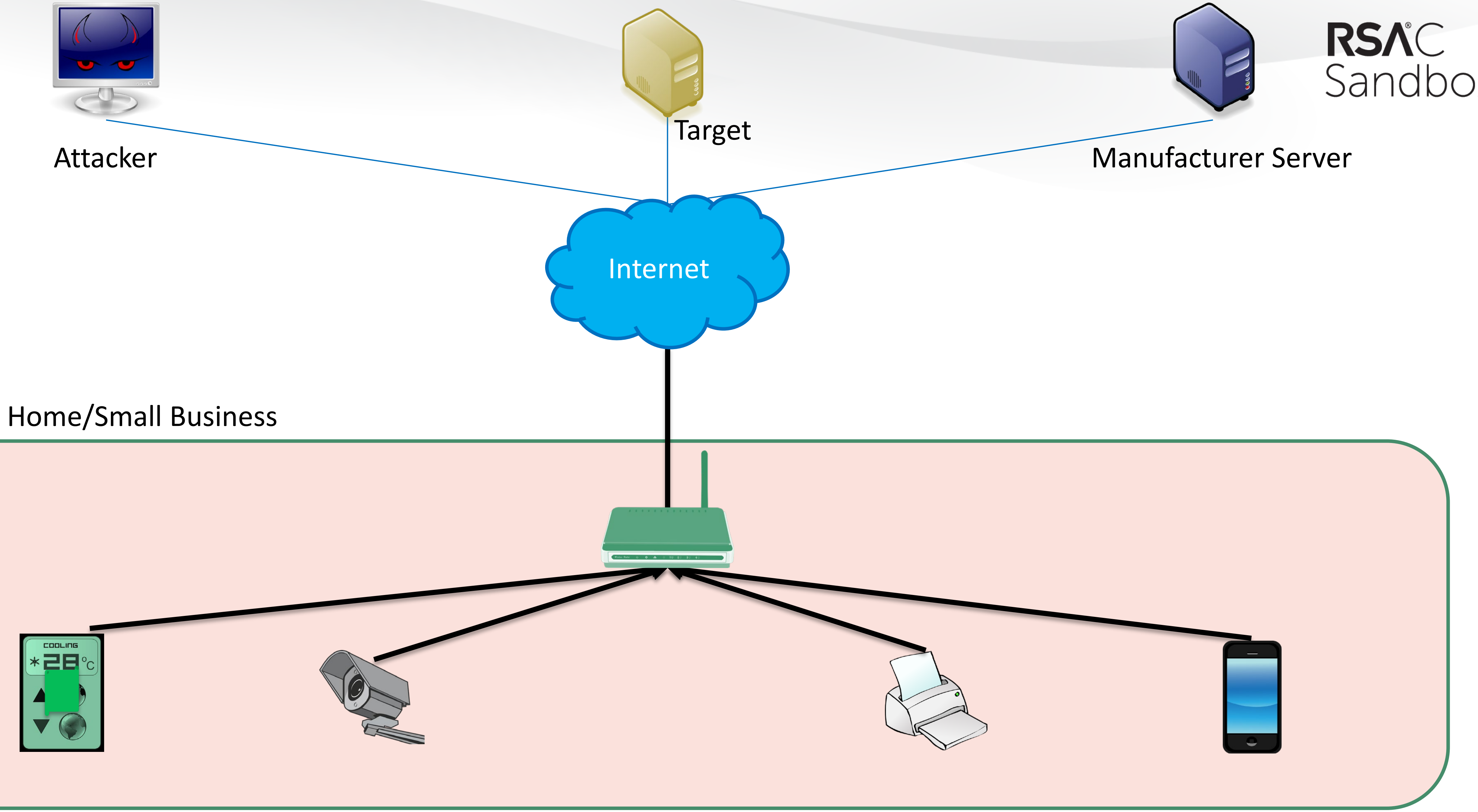
Manufacturer Server

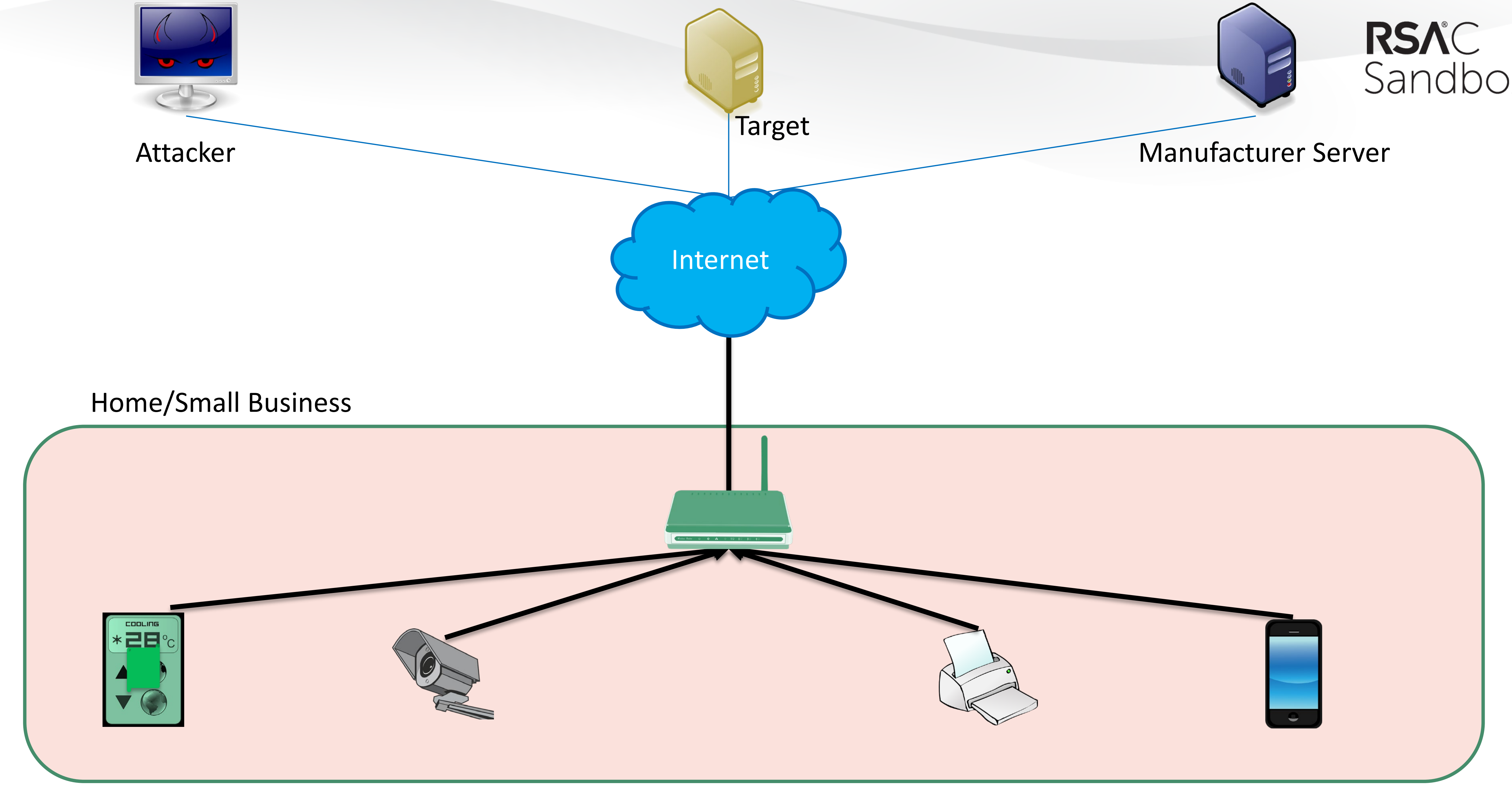
Internet

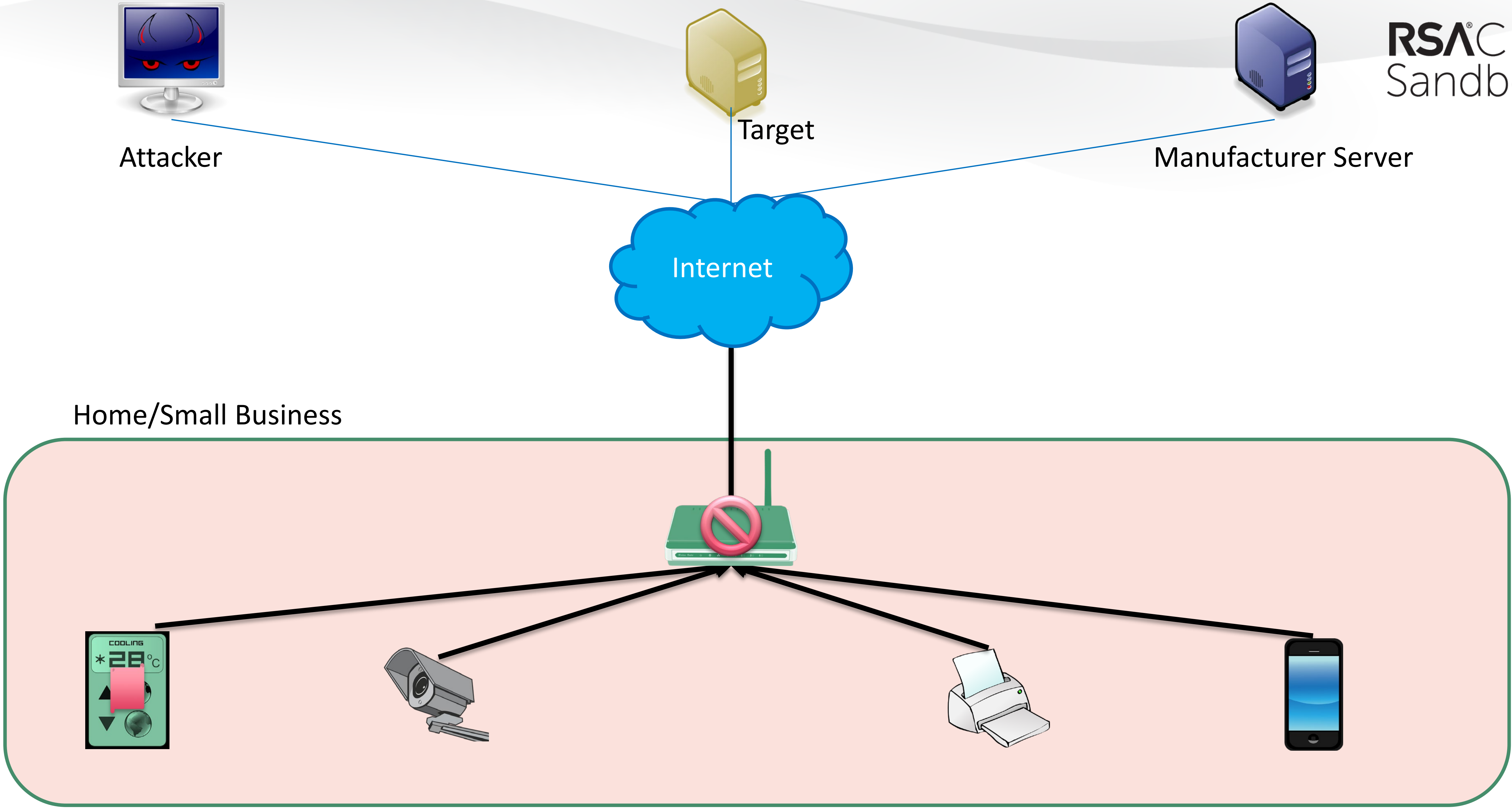
Home/Small Business

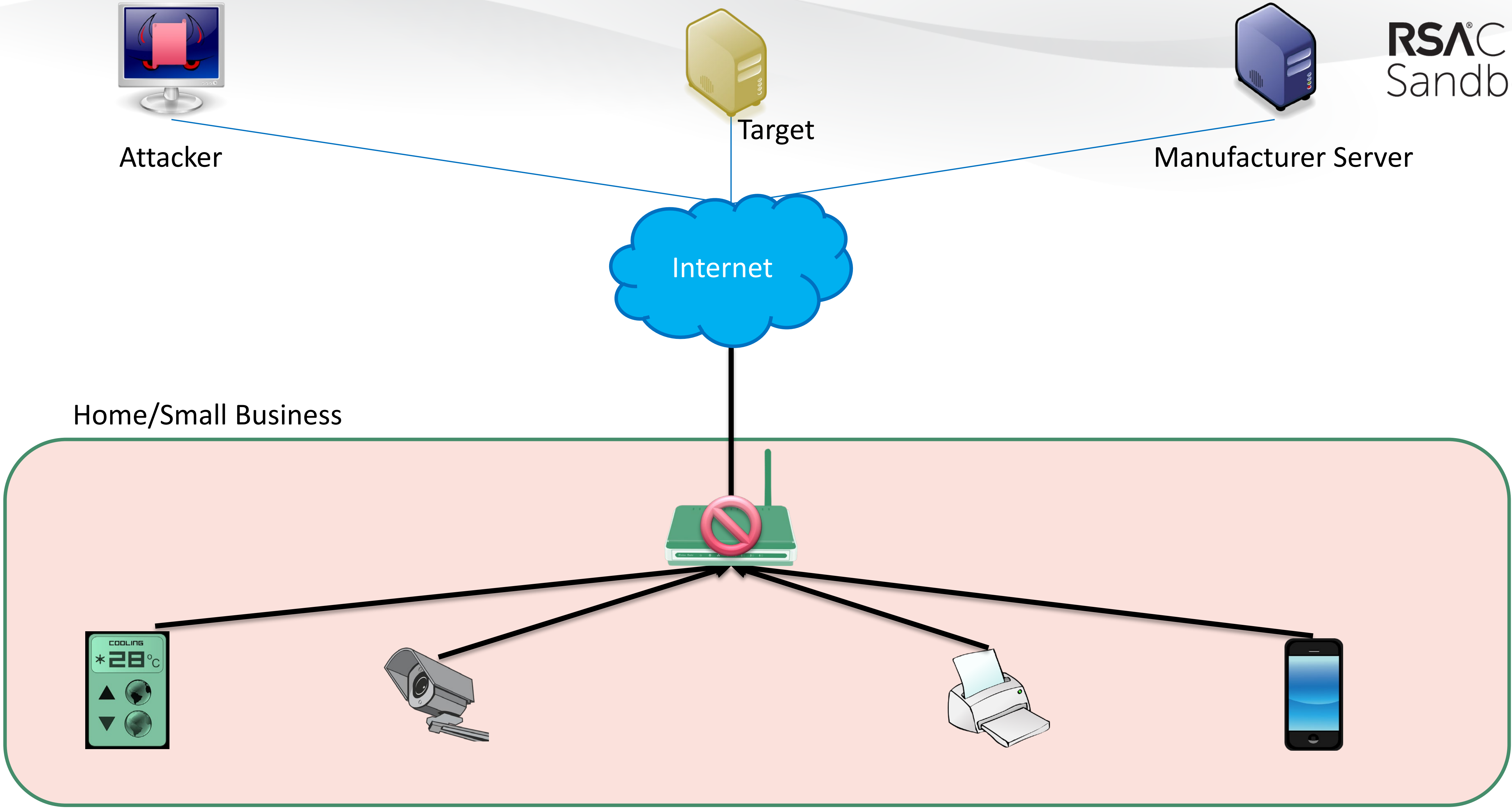
Typical Home/Small Business Network (With MUD)


Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C


Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C

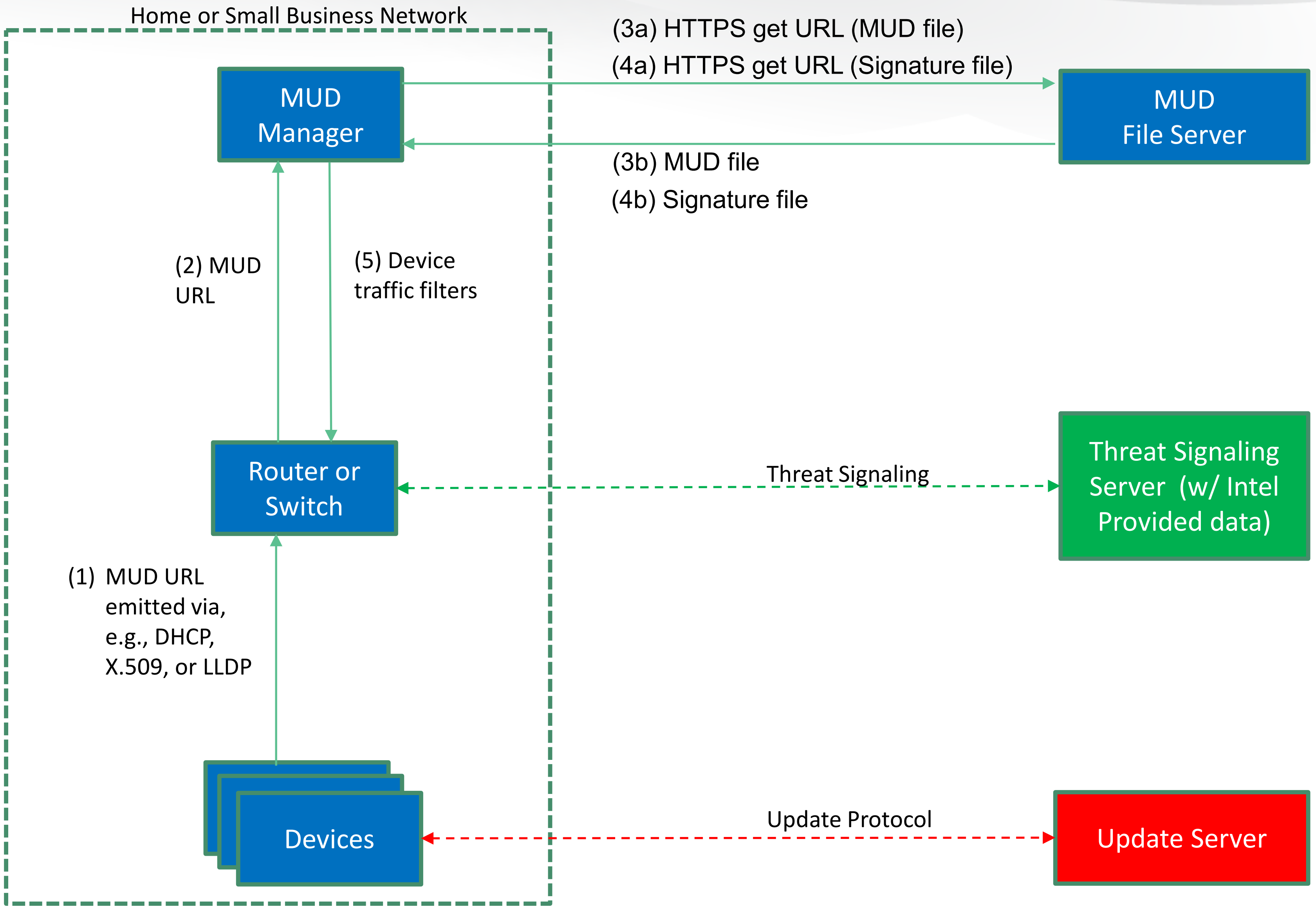

Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C


Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C


Attacker
Target
Manufacturer Server
Internet
Home/Small Business
COOLING
28°C

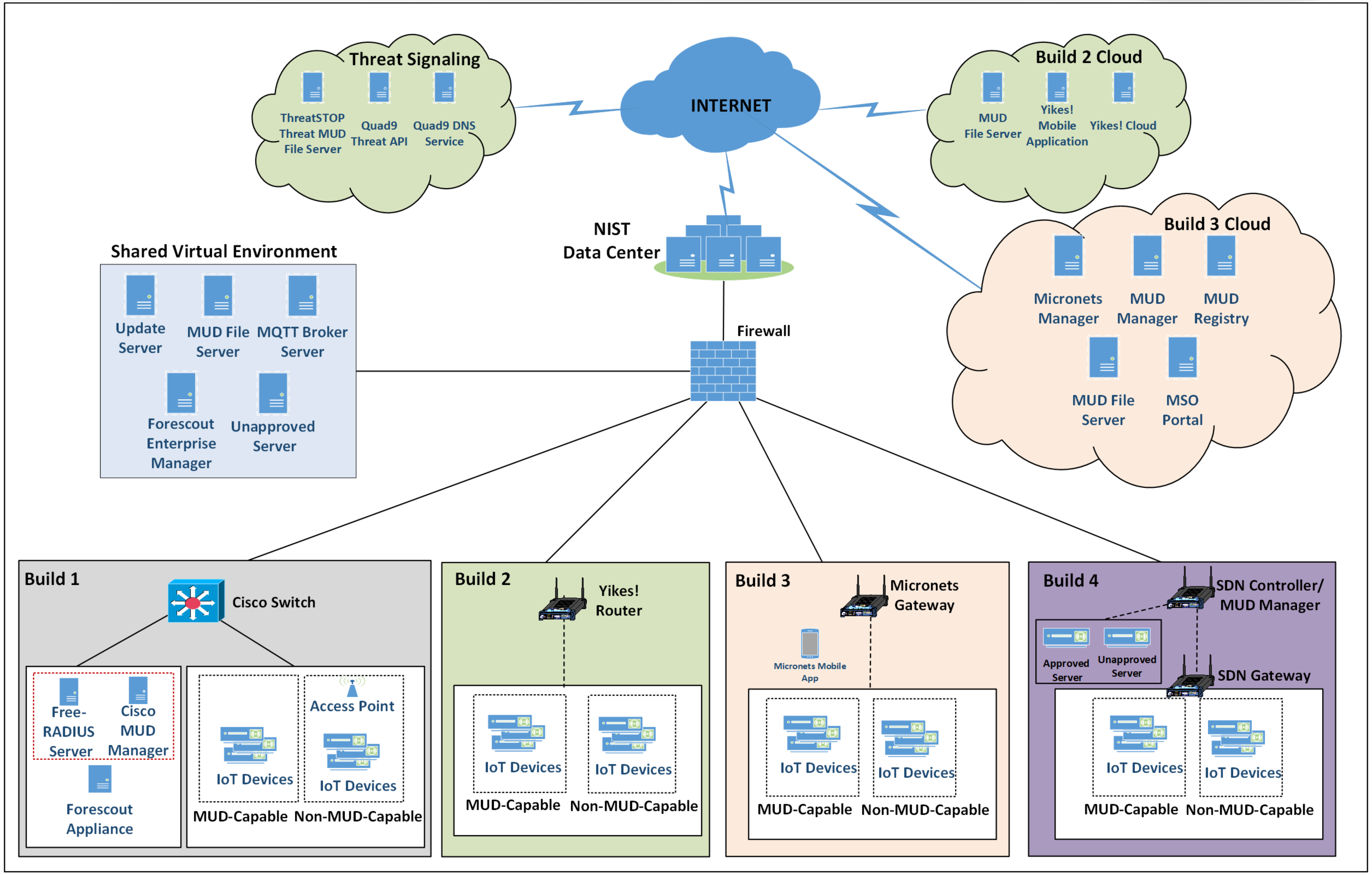
Architecture Overview

Reference Architecture


Home or Small Business Network
MUD Manager
Router or Switch
Devices
(2) MUD URL
(5) Device traffic filters
(1) MUD URL emitted via, e.g., DHCP, X.509, or LLDP
(3a) HTTPS get URL (MUD file)
(4a) HTTPS get URL (Signature file)
(3b) MUD file
(4b) Signature file
MUD File Server
Threat Signaling
Threat Signaling Server (w/ Intel Provided data)
Update Protocol
Update Server

Lab Architecture

RSAC Sandbox



NCCoE NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

Project Status

- Build 1, 2 and 4 Practice Guide SP 1800-15
  - Preliminary Draft Published in Nov. 2019

- Currently working on Build 3
  - Includes MUD and DPP (Device Provisioning Protocol)

https://www.nccoe.nist.gov/projects/building-blocks/mitigating-iot-based-ddos

Collaborators

arm

CableLabs

cisco

ctia

digicert

FORESCOUT

GLOBAL CYBER ALLIANCE

MasterPeace Solutions, Ltd.

molex
one company > a world of innovation

PATTON
Let's Connect!

Symantec.

Build 1 Demo Presentation

BUILD 1 – CISCO

- Cisco MUD Manager and FreeRadius
- Cisco Catalyst Switch
- NCCoE hosted MUD File Server
- MUD File
- DigiCert certificates
- MUD-capable IoT devices:
  - Molex PoE GW and Light Engine
  - Devkits
- Non-MUD-capable IoT devices
- NCCoE hosted Update Server
- NCCoE hosted Unapproved Server
- NCCoE hosted MQTT Broker Server
- Forescout and Forescout Enterprise Manager

PHASE 1

BUILD 2 – MASTERPEACE & GCA

- Yikes! Router including MUD Manager
- Yikes! Cloud & Yikes! Mobile App
- MasterPeace hosted MUD File Server
- MUD File
- DigiCert certificates
- MUD-capable IoT devices - Devkits
- Non-MUD-capable IoT devices
- NCCoE hosted Update Server
- NCCoE hosted Unapproved Server
- GCA Quad9 Threat Agent integrated into Yikes! Router
- GCA Quad9 Threat Signaling MUD Manager integrated into Yikes! Router
- GCA Quad9 Threat-Signaling DNS Services
- GCA Quad9 Threat API
- ThreatSTOP Threat MUD File Server
- ThreatSTOP Threat MUD File

BUILD 4 - NIST

- OpenDaylight SDN Controller including MUD Manager
- NCCoE hosted MUD File Server
- MUD File
- Wireless SDN Switch
- DigiCert certificates
- MUD-capable IoT devices - Devkits
- Non-MUD-capable IoT devices
- NCCoE hosted Unapproved Server
- Approved Server

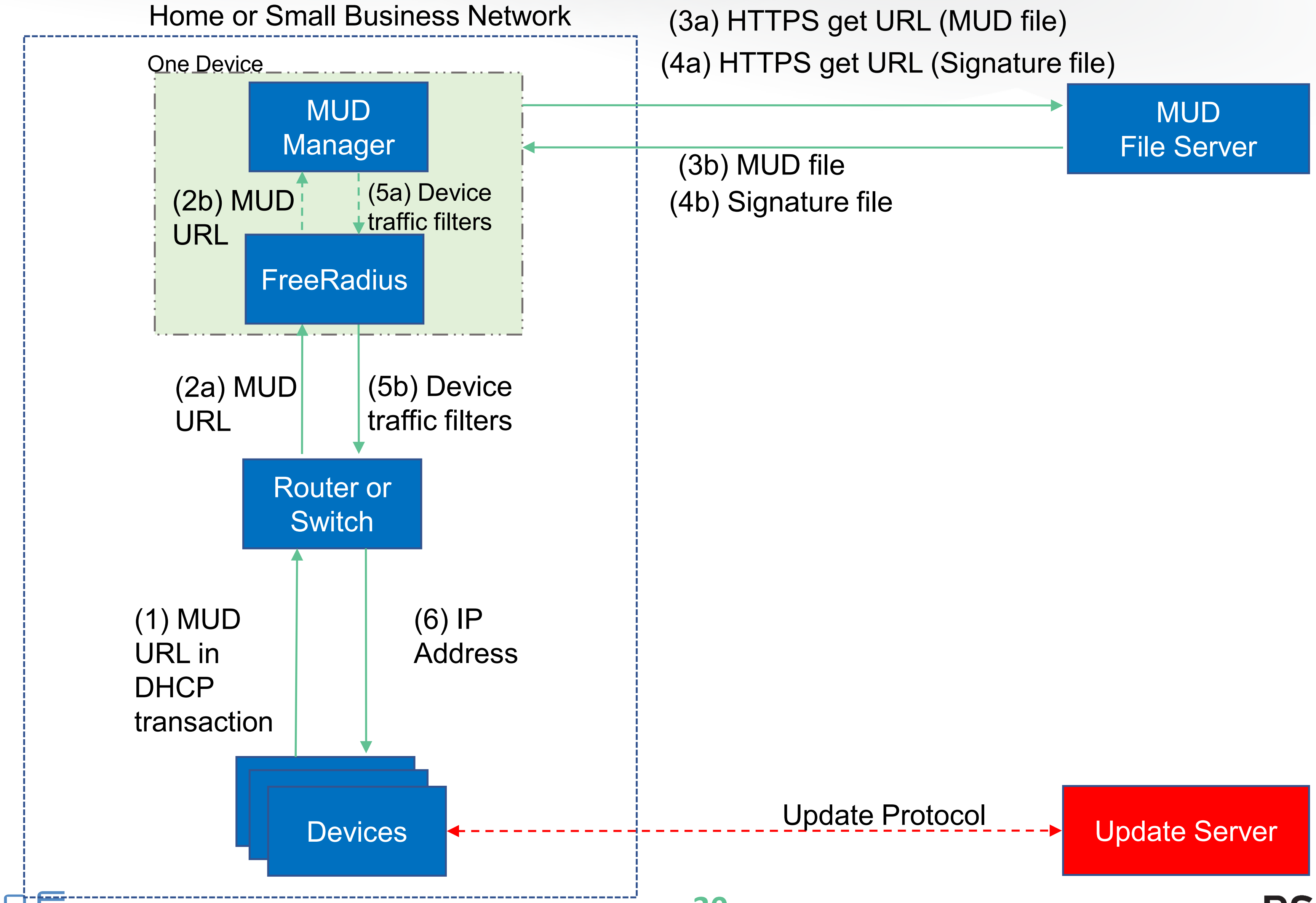
PHASE 2

BUILD 3 - CABLELABS

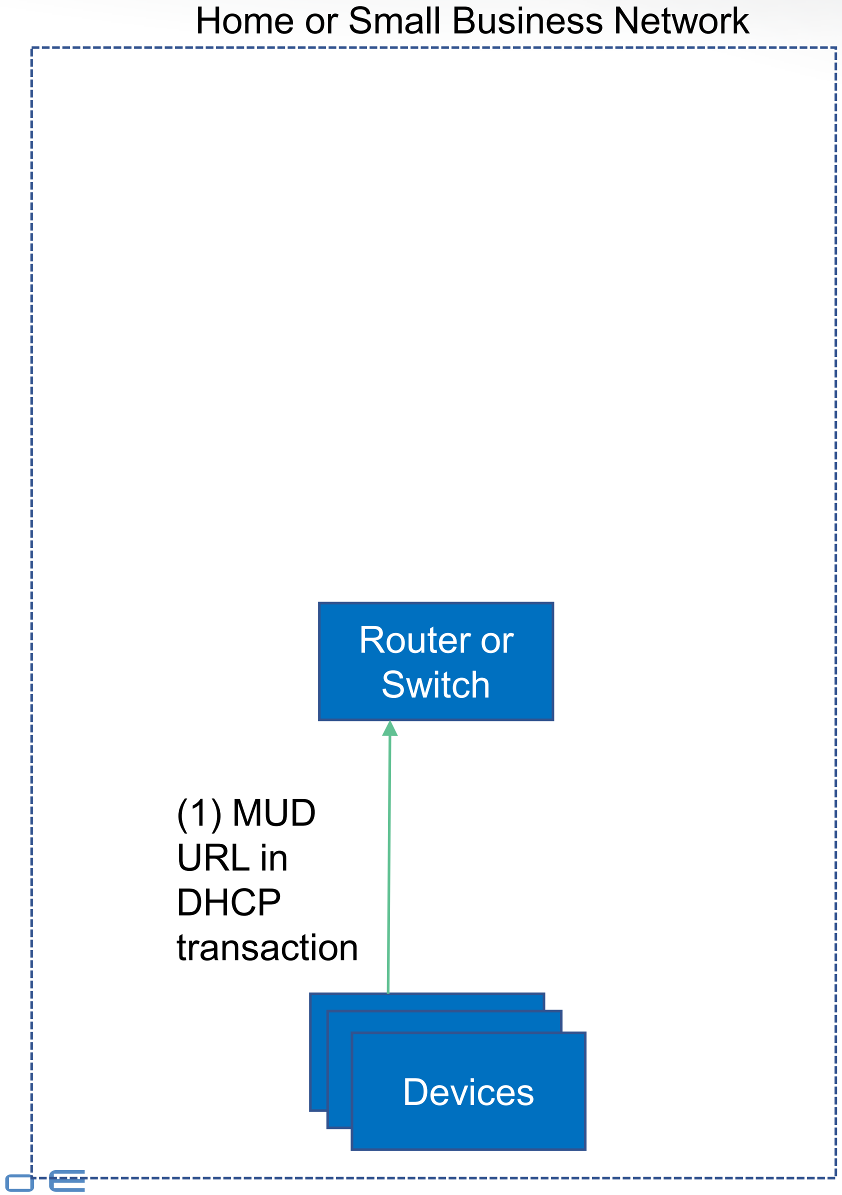
- Micronets Gateway
- Micronets Manager
- MUD Manager
- MUD Registry
- MSO Portal
- MUD File Server
- MUD File
- DigiCert certificates
- MUD-capable IoT devices - Devkits
- Non-MUD-capable IoT devices
- Update Server
- Unapproved Server
- Micronets Mobile App

PHASE 3

Logical Architecture – Build 1


Home or Small Business Network
One Device
MUD Manager
(2b) MUD URL
(5a) Device traffic filters
FreeRadius
(2a) MUD URL
(5b) Device traffic filters
Router or Switch
(1) MUD URL in DHCP transaction
(6) IP Address
Devices
(3a) HTTPS get URL (MUD file)
(4a) HTTPS get URL (Signature file)
MUD File Server
(3b) MUD file
(4b) Signature file
Update Protocol
Update Server

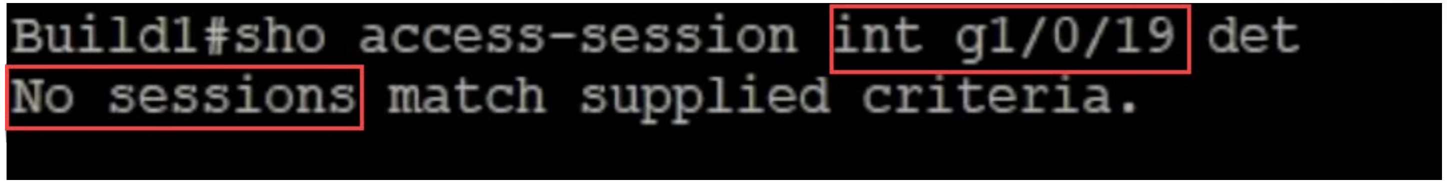
Step 1: Connect Device


Home or Small Business Network
Router or Switch
(1) MUD URL in DHCP transaction
Devices

Step 1: Connect Device

1. No session on interface

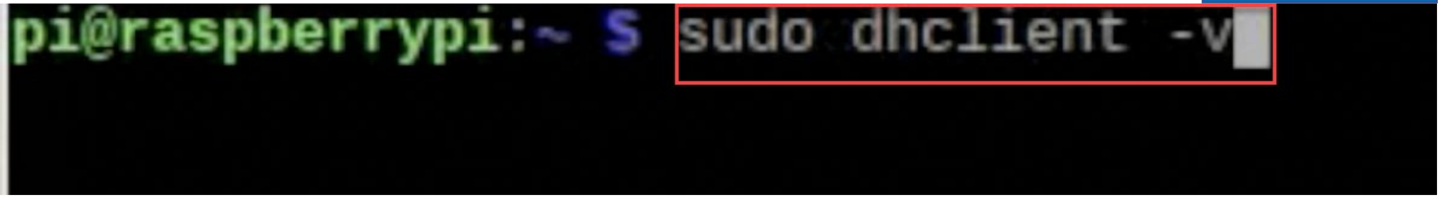
Router or Switch



```
Build1#sho access-session int g1/0/19 det
No sessions match supplied criteria.
```

2. Connect MUD enabled IoT Device

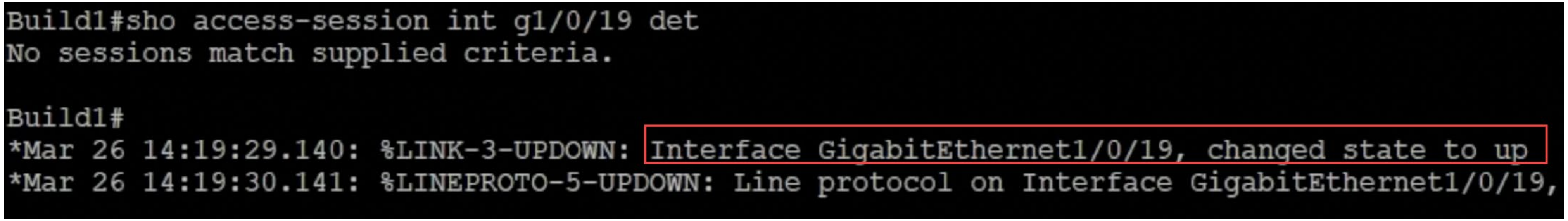
Devices



```
pi@raspberrypi:~ $ sudo dhclient -v
```

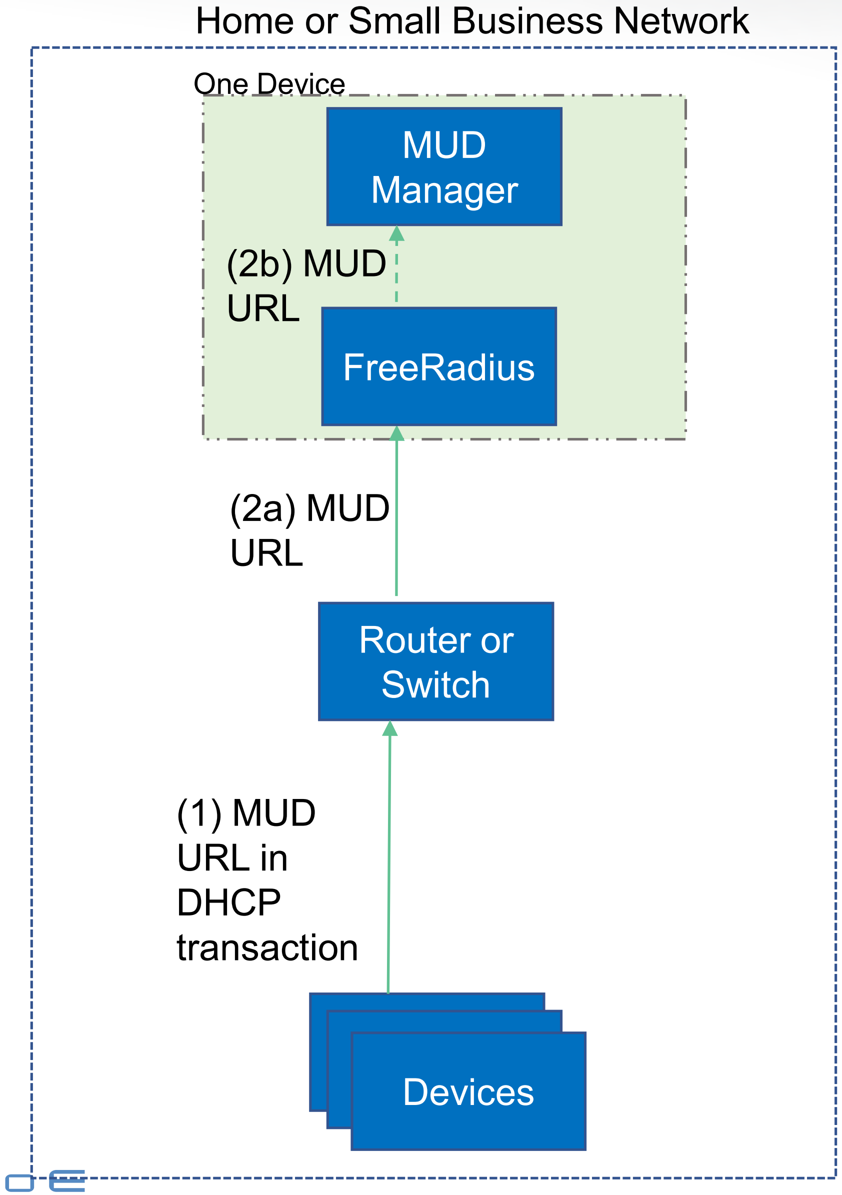
3. Interface state changed to up

Router or Switch



```
Build1#sho access-session int g1/0/19 det
No sessions match supplied criteria.

Build1#
*Mar 26 14:19:29.140: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/19, changed state to up
*Mar 26 14:19:30.141: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet1/0/19,
```

Step 2a/2b: Send MUD URL to MUD Manager



Home or Small Business Network

One Device

MUD Manager

(2b) MUD URL

FreeRadius

(2a) MUD URL

Router or Switch

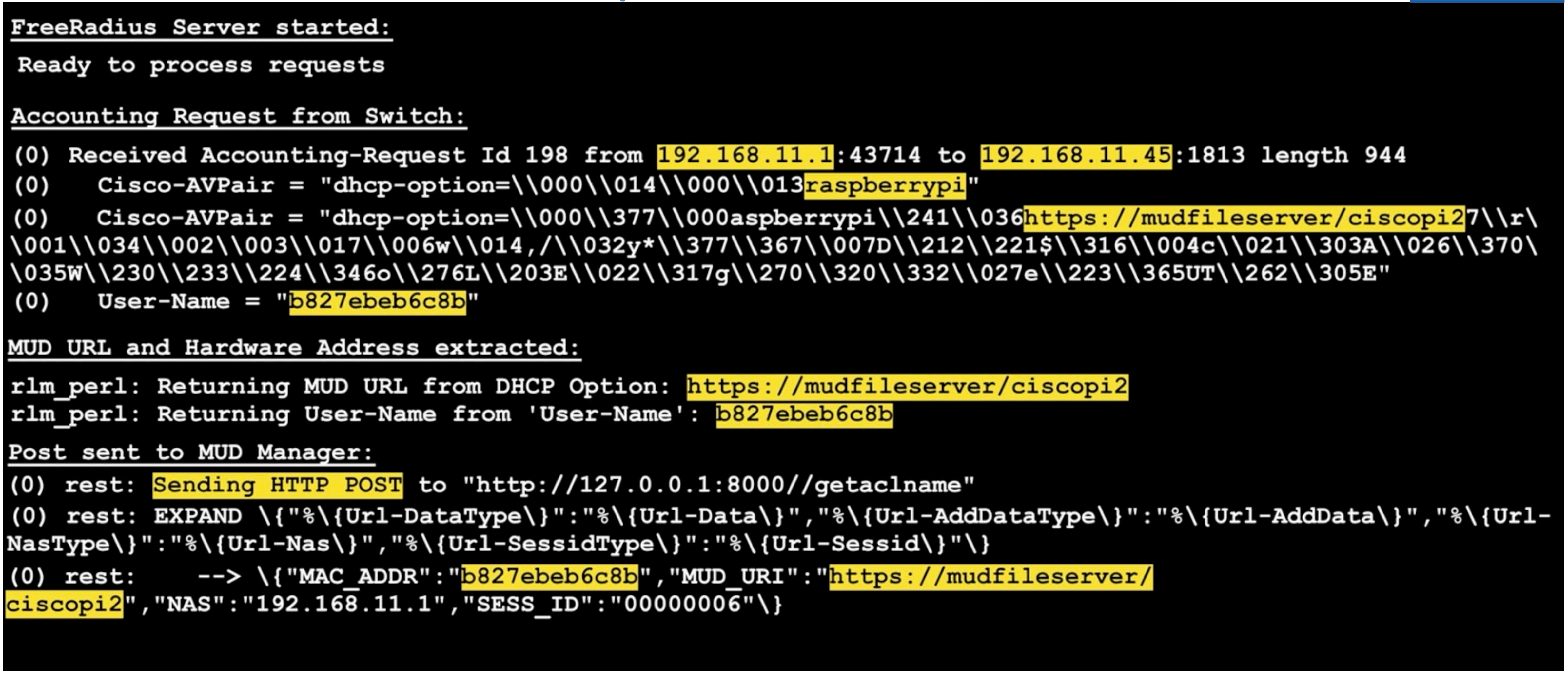
(1) MUD URL in DHCP transaction

Devices

Step 2a/2b: Send MUD URL to MUD Manager

FreeRadius

1. FreeRadius service receives and passes MUD URL

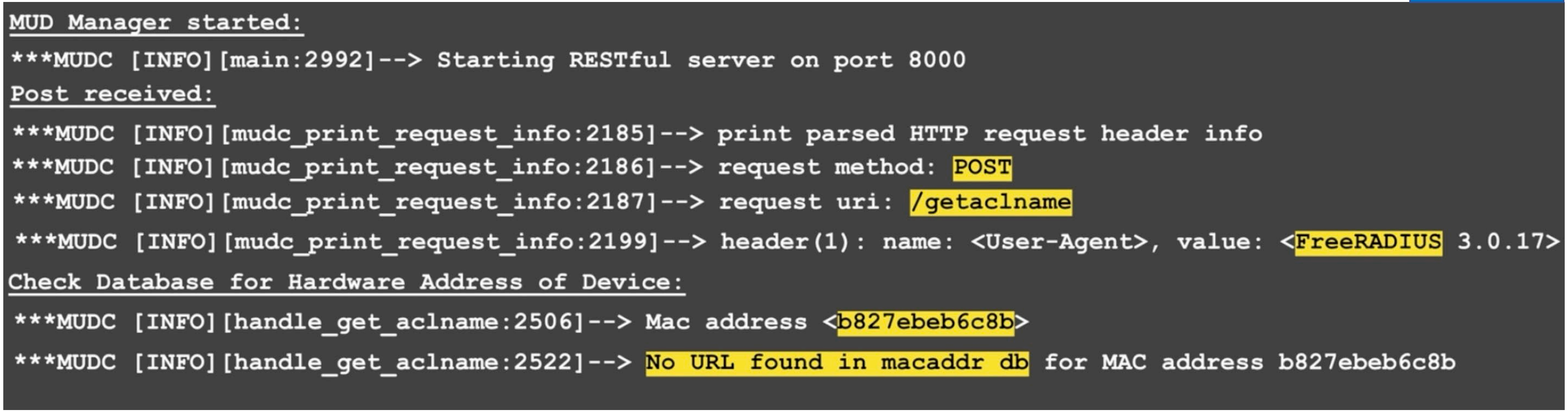


```
FreeRadius Server started:
 Ready to process requests

Accounting Request from Switch:
(0) Received Accounting-Request Id 198 from 192.168.11.1:43714 to 192.168.11.45:1813 length 944
(0)    Cisco-AVPair = "dhcp-option=\\000\\014\\000\\013raspberrypi"
(0)    Cisco-AVPair = "dhcp-option=\\000\\377\\000aspberrypi\\241\\036https://mudfileserver/ciscopi27\\r\
\001\\034\\002\\003\\017\\006w\\014,/\\032y*\\377\\367\\007D\\212\\221$\\316\\004c\\021\\303A\\026\\370\
\035W\\230\\233\\224\\346o\\276L\\203E\\022\\317g\\270\\320\\332\\027e\\223\\365UT\\262\\305E"
(0)    User-Name = "b827ebeb6c8b"

MUD URL and Hardware Address extracted:
rlm_perl: Returning MUD URL from DHCP Option: https://mudfileserver/ciscopi2
rlm_perl: Returning User-Name from 'User-Name': b827ebeb6c8b
Post sent to MUD Manager:
(0) rest: Sending HTTP POST to "http://127.0.0.1:8000//getaclname"
(0) rest: EXPAND \{"%\{Url-DataType\}":"%\{Url-Data\}","%\{Url-AddDataType\}":"%\{Url-AddData\}","%\{Url-
NasType\}":"%\{Url-Nas\}","%\{Url-SessidType\}":"%\{Url-Sessid\}"\}
(0) rest:     --> \{"MAC_ADDR":"b827ebeb6c8b","MUD_URI":"https://mudfileserver/
ciscopi2","NAS":"192.168.11.1","SESS_ID":"00000006"\}
```

Step 2b: Send MUD URL to MUD Manager

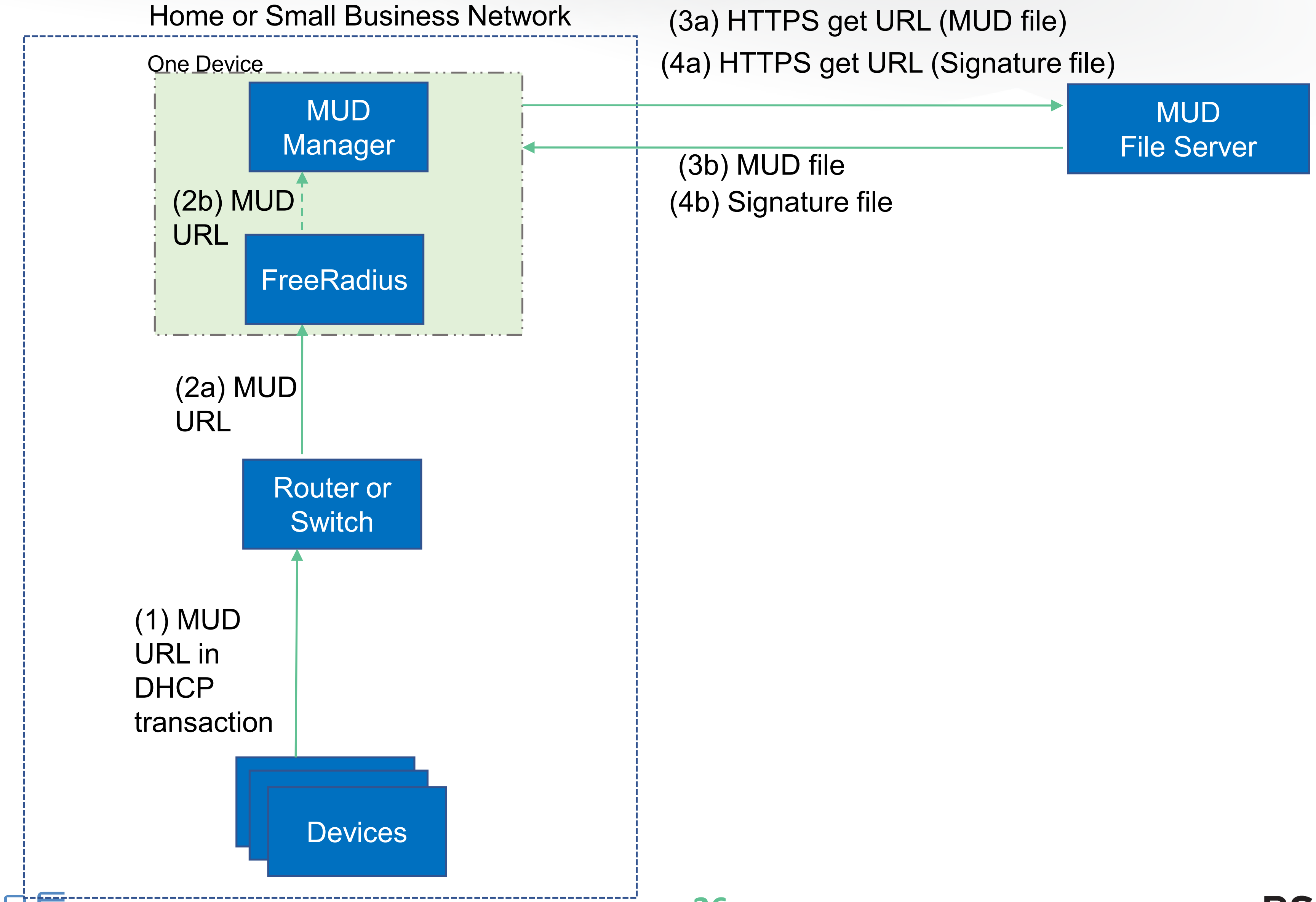
MUD Manager

2. MUD Manager receives MUD enabled IoT Device information from FreeRadius Service



```
MUD Manager started:
***MUDC [INFO][main:2992]--> Starting RESTful server on port 8000
Post received:
***MUDC [INFO][mudc_print_request_info:2185]--> print parsed HTTP request header info
***MUDC [INFO][mudc_print_request_info:2186]--> request method: POST
***MUDC [INFO][mudc_print_request_info:2187]--> request uri: /getaclname
***MUDC [INFO][mudc_print_request_info:2199]--> header(1): name: <User-Agent>, value: <FreeRADIUS 3.0.17>
Check Database for Hardware Address of Device:
***MUDC [INFO][handle_get_aclname:2506]--> Mac address <b827ebeb6c8b>
***MUDC [INFO][handle_get_aclname:2522]--> No URL found in macaddr db for MAC address b827ebeb6c8b
```

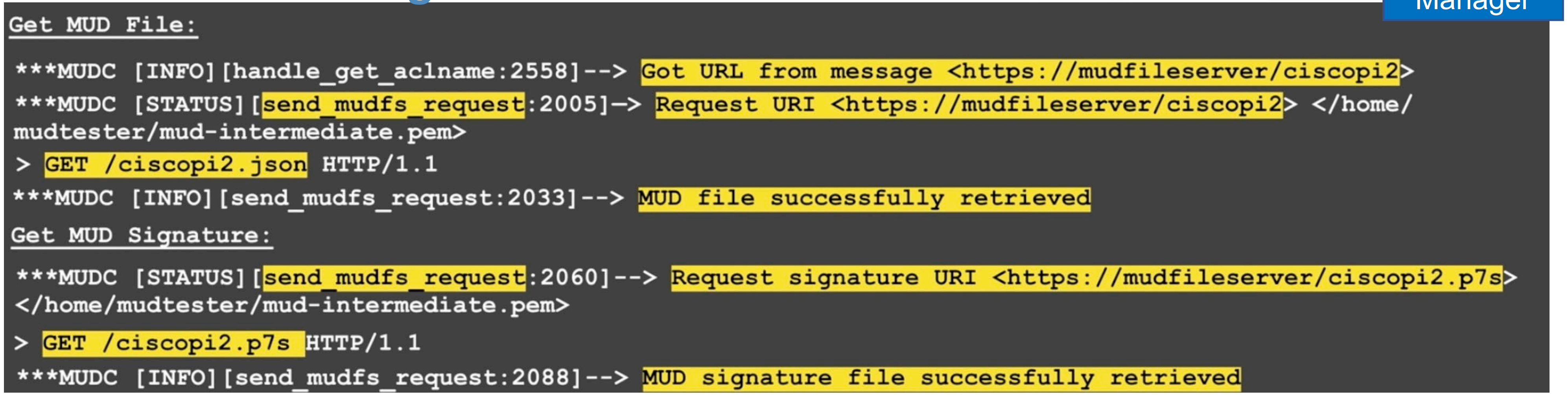
Step 3/4: Get MUD and Signature File


Home or Small Business Network
One Device
MUD Manager
(2b) MUD URL
FreeRadius
(2a) MUD URL
Router or Switch
(1) MUD URL in DHCP transaction
Devices
(3a) HTTPS get URL (MUD file)
(4a) HTTPS get URL (Signature file)
MUD File Server
(3b) MUD file
(4b) Signature file

Step 3/4: Get MUD and Signature File

1. Get MUD and Signature file

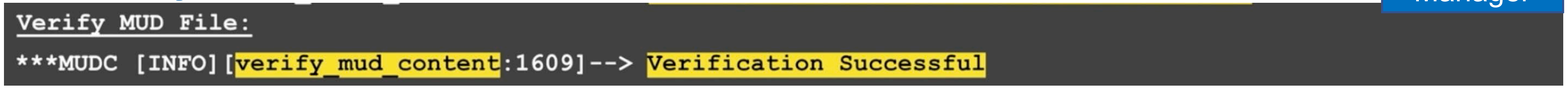
MUD Manager



```
Get MUD File:

***MUDC [INFO][handle_get_aclname:2558]--> Got URL from message <https://mudfileserver/ciscopi2>
***MUDC [STATUS][send_mudfs_request:2005]—> Request URI <https://mudfileserver/ciscopi2> </home/
mudtester/mud-intermediate.pem>
> GET /ciscopi2.json HTTP/1.1
***MUDC [INFO][send_mudfs_request:2033]--> MUD file successfully retrieved
Get MUD Signature:
***MUDC [STATUS][send_mudfs_request:2060]--> Request signature URI <https://mudfileserver/ciscopi2.p7s>
</home/mudtester/mud-intermediate.pem>
> GET /ciscopi2.p7s HTTP/1.1
***MUDC [INFO][send_mudfs_request:2088]--> MUD signature file successfully retrieved
```

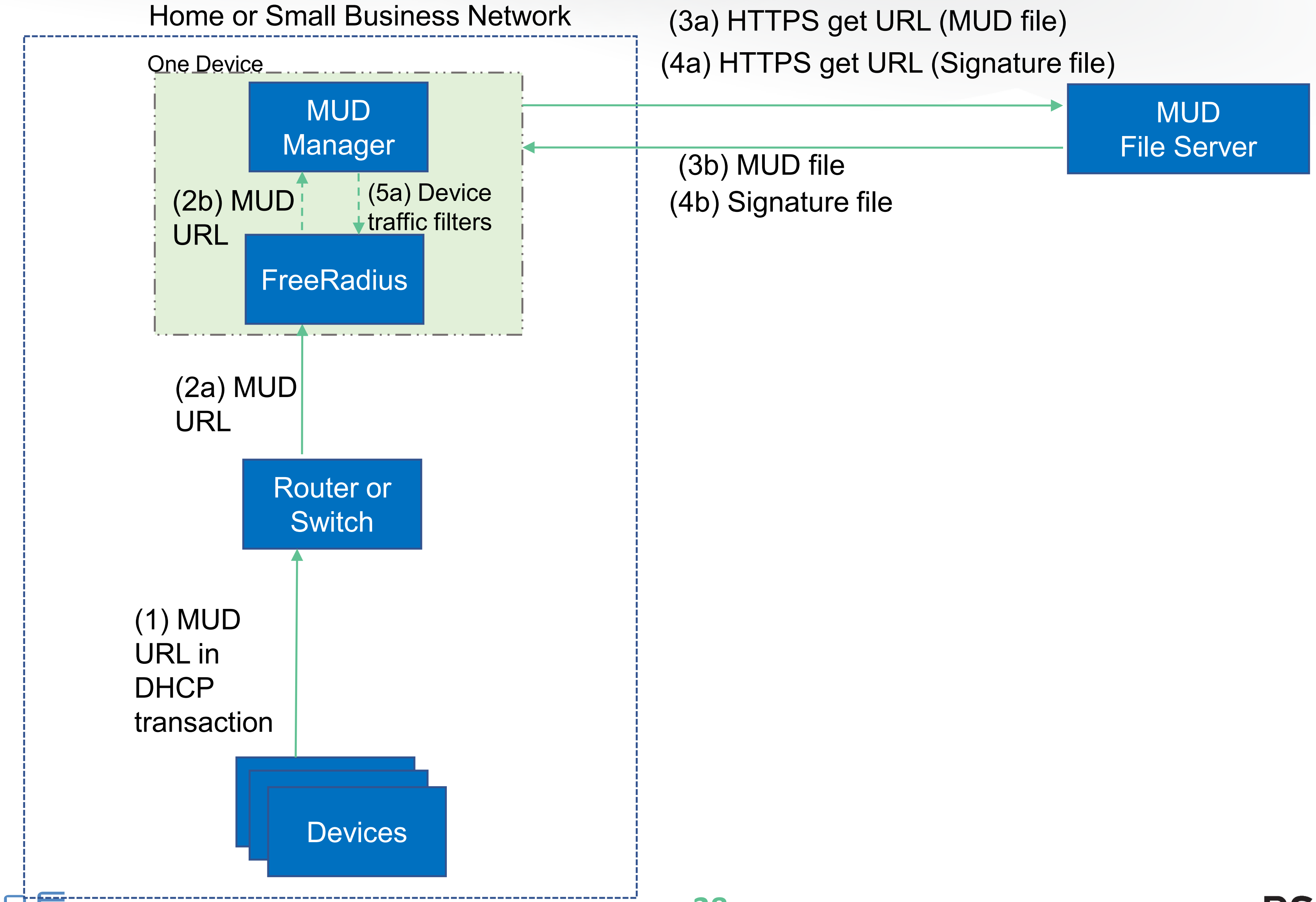
2. Verify MUD file

MUD Manager



```
Verify MUD File:
***MUDC [INFO][verify_mud_content:1609]--> Verification Successful
```

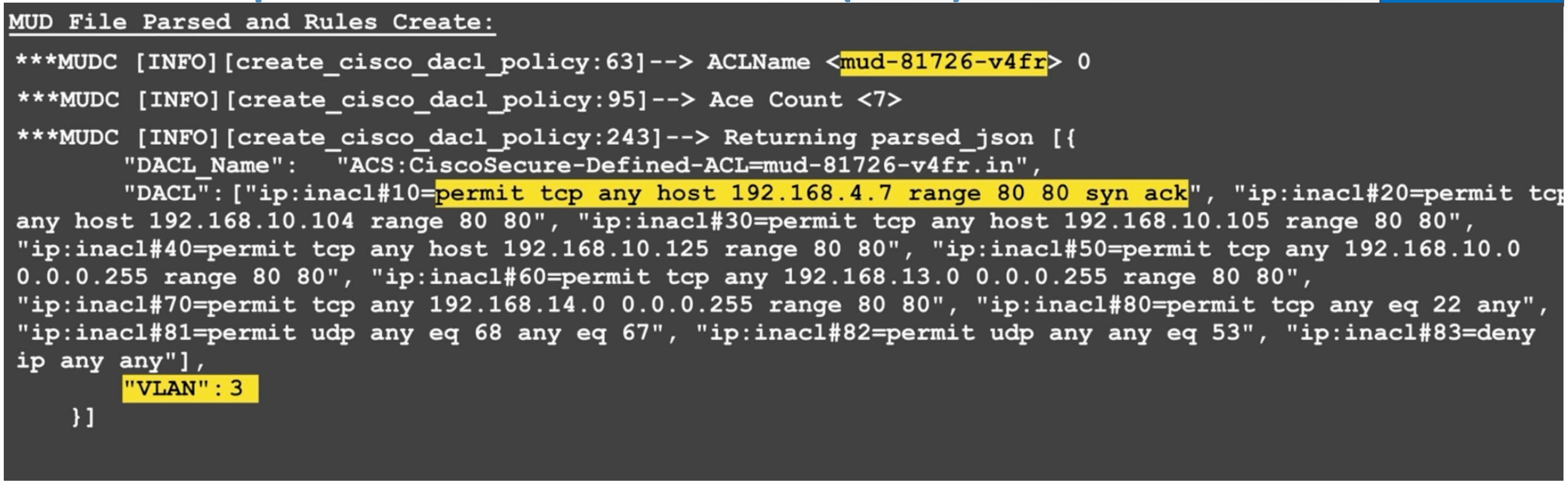
Step 5a: Send Device Traffic Filters


Home or Small Business Network
One Device
MUD Manager
(2b) MUD URL
(5a) Device traffic filters
FreeRadius
(2a) MUD URL
Router or Switch
(1) MUD URL in DHCP transaction
Devices
(3a) HTTPS get URL (MUD file)
(4a) HTTPS get URL (Signature file)
MUD File Server
(3b) MUD file
(4b) Signature file

Step 5a: Send Device Traffic Filters

MUD Manager

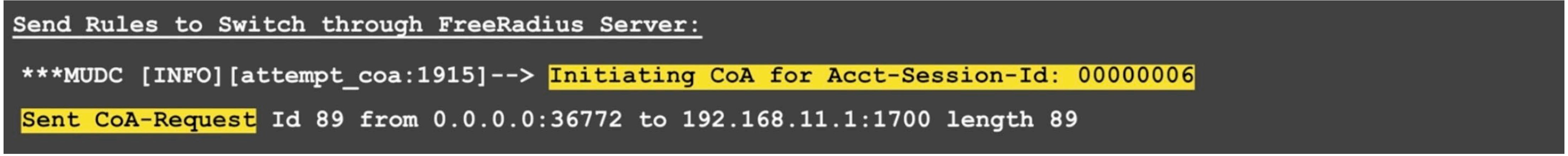
1. MUD File parsed and translated to ACL (rules)



```
MUD File Parsed and Rules Create:

***MUDC [INFO][create_cisco_dacl_policy:63]--> ACLName <mud-81726-v4fr> 0

***MUDC [INFO][create_cisco_dacl_policy:95]--> Ace Count <7>

***MUDC [INFO][create_cisco_dacl_policy:243]--> Returning parsed_json [{
       "DACL_Name":  "ACS:CiscoSecure-Defined-ACL=mud-81726-v4fr.in",
       "DACL":["ip:inacl#10=permit tcp any host 192.168.4.7 range 80 80 syn ack", "ip:inacl#20=permit tcp
any host 192.168.10.104 range 80 80", "ip:inacl#30=permit tcp any host 192.168.10.105 range 80 80",
"ip:inacl#40=permit tcp any host 192.168.10.125 range 80 80", "ip:inacl#50=permit tcp any 192.168.10.0
0.0.0.255 range 80 80", "ip:inacl#60=permit tcp any 192.168.13.0 0.0.0.255 range 80 80",
"ip:inacl#70=permit tcp any 192.168.14.0 0.0.0.255 range 80 80", "ip:inacl#80=permit tcp any eq 22 any",
"ip:inacl#81=permit udp any eq 68 any eq 67", "ip:inacl#82=permit udp any any eq 53", "ip:inacl#83=deny
ip any any"],
       "VLAN":3
    }]
```

MUD Manager

2. MUD Manager sends ACL

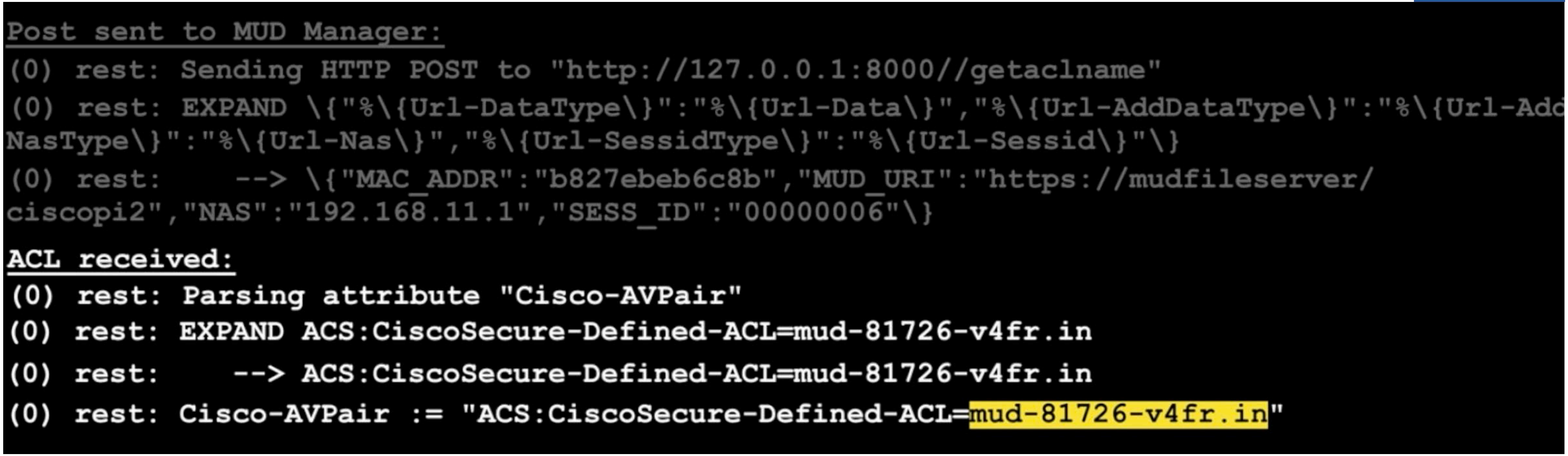


```
Send Rules to Switch through FreeRadius Server:

 ***MUDC [INFO][attempt_coa:1915]--> Initiating CoA for Acct-Session-Id: 00000006

 Sent CoA-Request Id 89 from 0.0.0.0:36772 to 192.168.11.1:1700 length 89
```

Step 5a: Send Device Traffic Filters

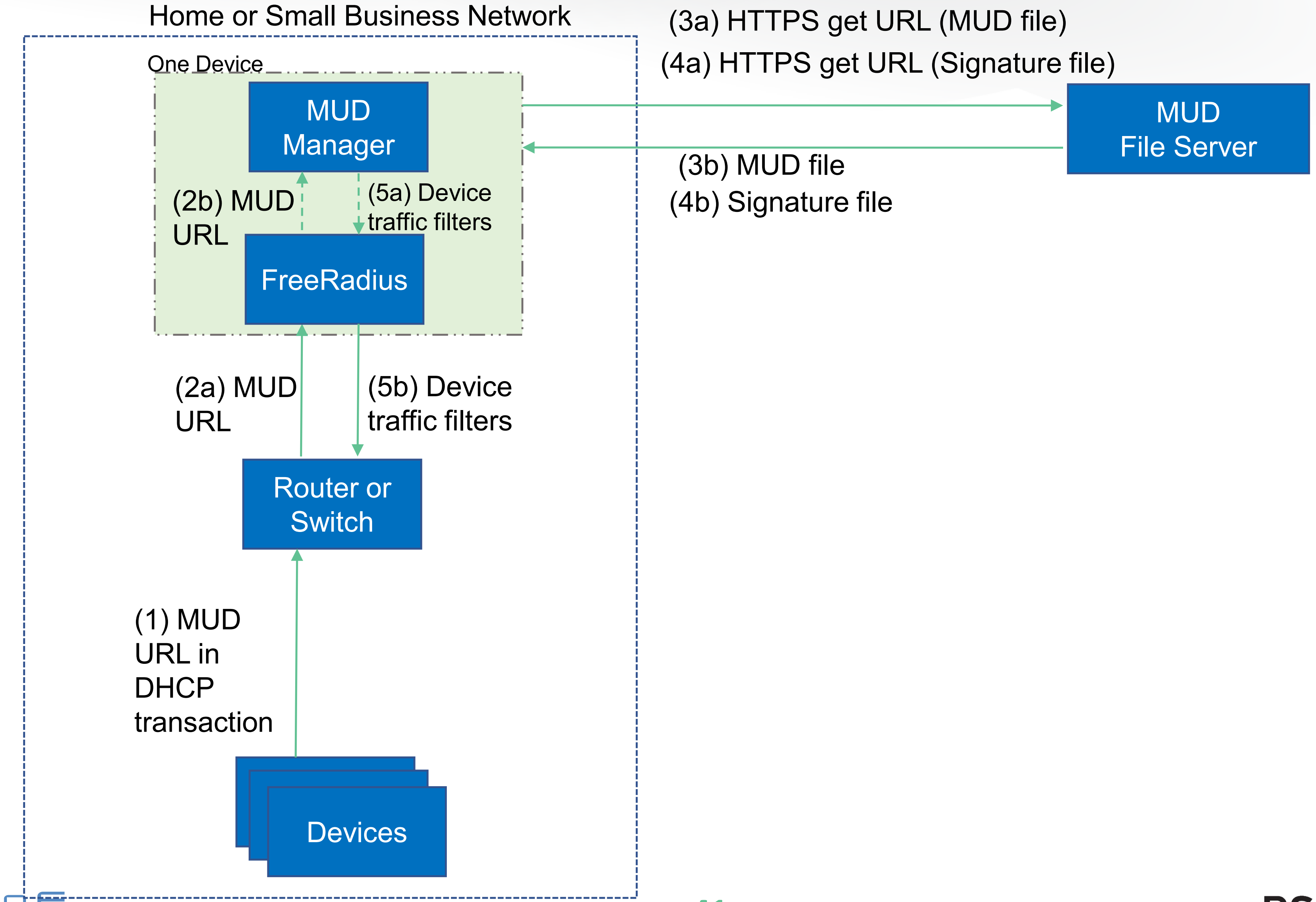
FreeRadius

3. FreeRadius receives ACL from MUD Manager



```
Post sent to MUD Manager:
(0) rest: Sending HTTP POST to "http://127.0.0.1:8000//getaclname"
(0) rest: EXPAND \{"%\{Url-DataType\}":"%\{Url-Data\}","%\{Url-AddDataType\}":"%\{Url-Add
NasType\}":"%\{Url-Nas\}","%\{Url-SessidType\}":"%\{Url-Sessid\}"\}
(0) rest:    --> \{"MAC_ADDR":"b827ebeb6c8b","MUD_URI":"https://mudfileserver/
ciscopi2","NAS":"192.168.11.1","SESS_ID":"00000006"\}

ACL received:
(0) rest: Parsing attribute "Cisco-AVPair"
(0) rest: EXPAND ACS:CiscoSecure-Defined-ACL=mud-81726-v4fr.in
(0) rest:    --> ACS:CiscoSecure-Defined-ACL=mud-81726-v4fr.in
(0) rest: Cisco-AVPair := "ACS:CiscoSecure-Defined-ACL=mud-81726-v4fr.in"
```

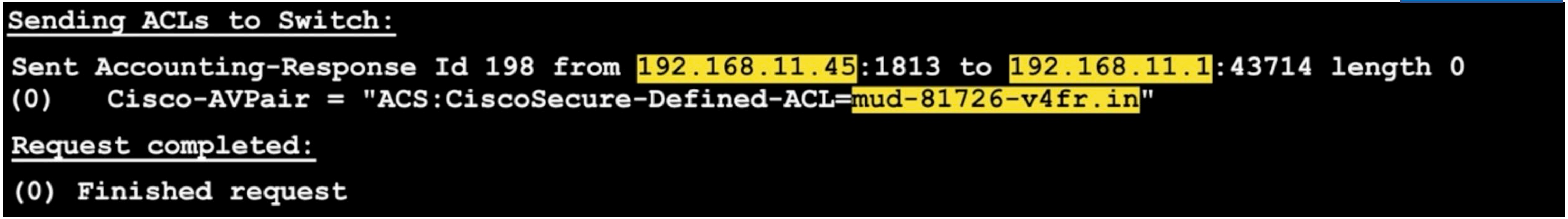
Step 5b: Send Device Traffic Filters


Home or Small Business Network
One Device
MUD Manager
(2b) MUD URL
(5a) Device traffic filters
FreeRadius
(2a) MUD URL
(5b) Device traffic filters
Router or Switch
(1) MUD URL in DHCP transaction
Devices
(3a) HTTPS get URL (MUD file)
(4a) HTTPS get URL (Signature file)
MUD File Server
(3b) MUD file
(4b) Signature file

Step 5b: Send Device Traffic Filters

FreeRadius

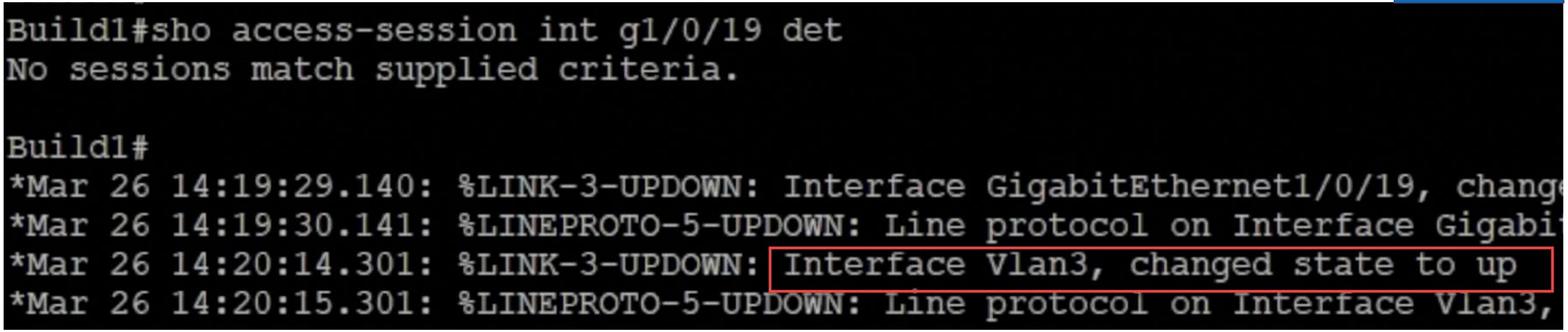
1. FreeRadius sends ACL to switch



```
Sending ACLs to Switch:

Sent Accounting-Response Id 198 from 192.168.11.45:1813 to 192.168.11.1:43714 length 0
(0)    Cisco-AVPair = "ACS:CiscoSecure-Defined-ACL=mud-81726-v4fr.in"

Request completed:

(0)  Finished request
```

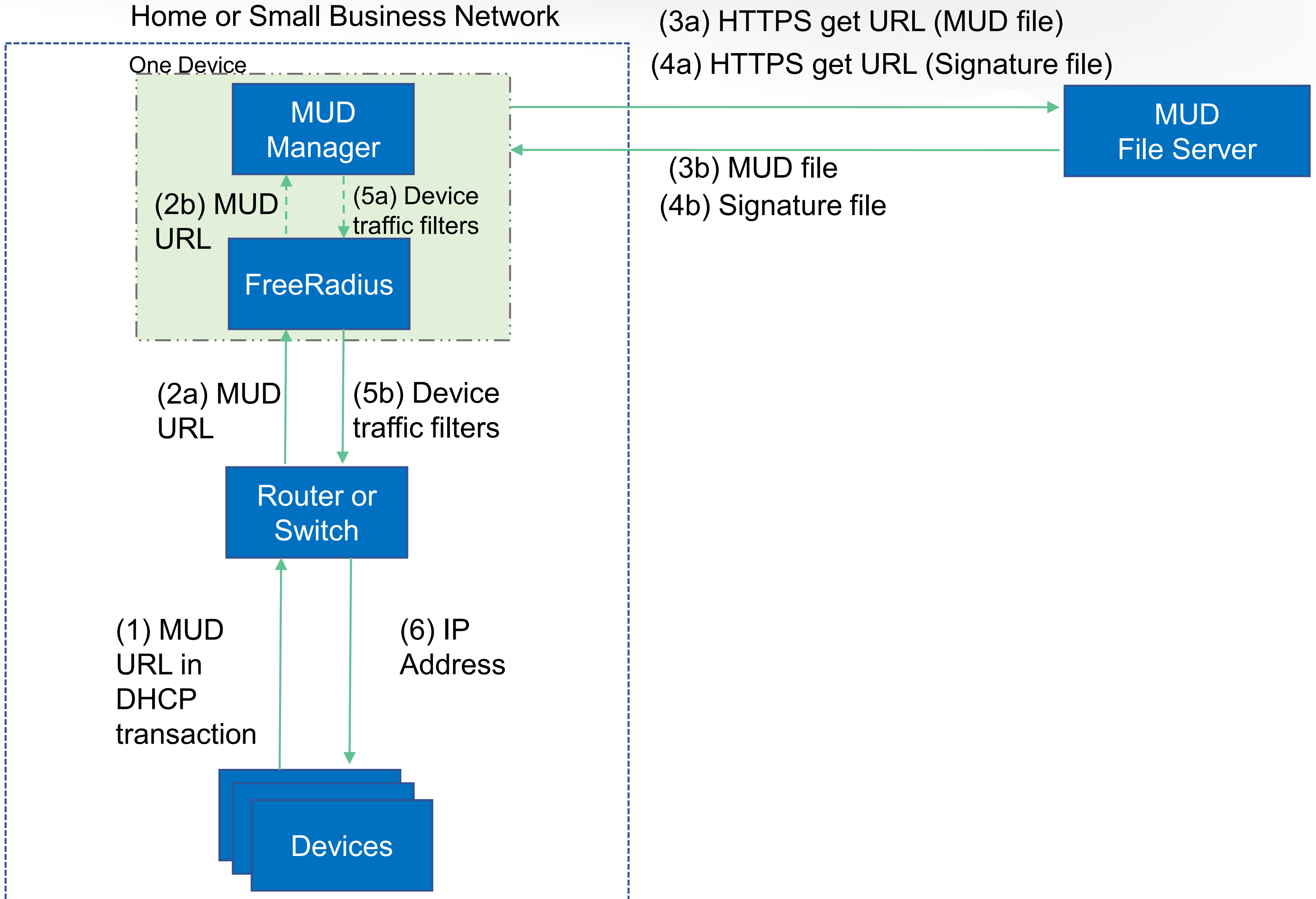
Router or Switch

2. ACL received, and configurations applied



```
Build1#sho access-session int g1/0/19 det
No sessions match supplied criteria.

Build1#
*Mar 26 14:19:29.140: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/19, change
*Mar 26 14:19:30.141: %LINEPROTO-5-UPDOWN: Line protocol on Interface Gigabi
*Mar 26 14:20:14.301: %LINK-3-UPDOWN: Interface Vlan3, changed state to up
*Mar 26 14:20:15.301: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan3,
```

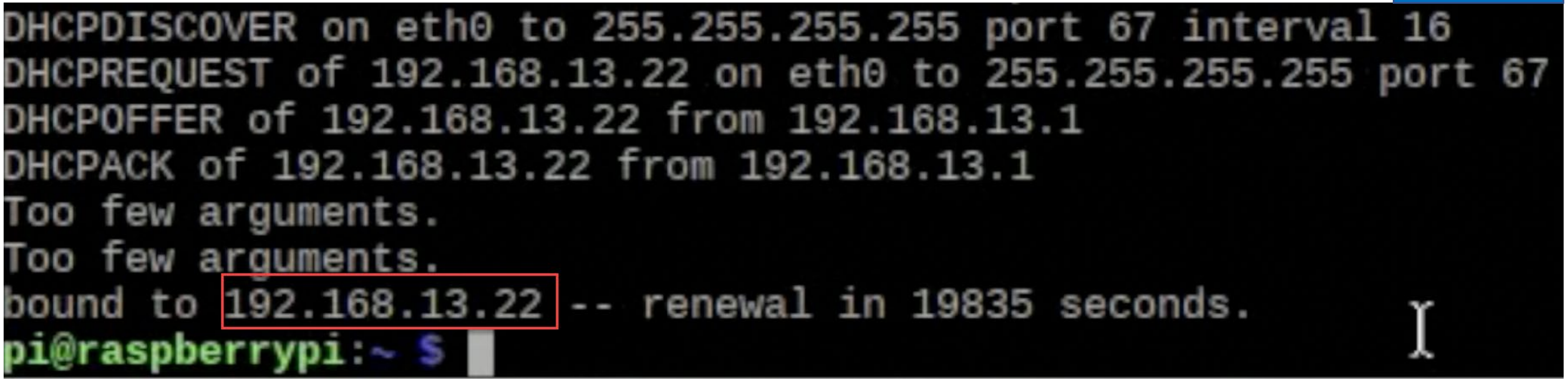
Step 6: IP Address Assigned


Home or Small Business Network
One Device
MUD Manager
(2b) MUD URL
(5a) Device traffic filters
FreeRadius
(2a) MUD URL
(5b) Device traffic filters
Router or Switch
(1) MUD URL in DHCP transaction
(6) IP Address
Devices
(3a) HTTPS get URL (MUD file)
(4a) HTTPS get URL (Signature file)
MUD File Server
(3b) MUD file
(4b) Signature file

Step 6: IP address assigned

Devices

1. IoT Device receives IP address

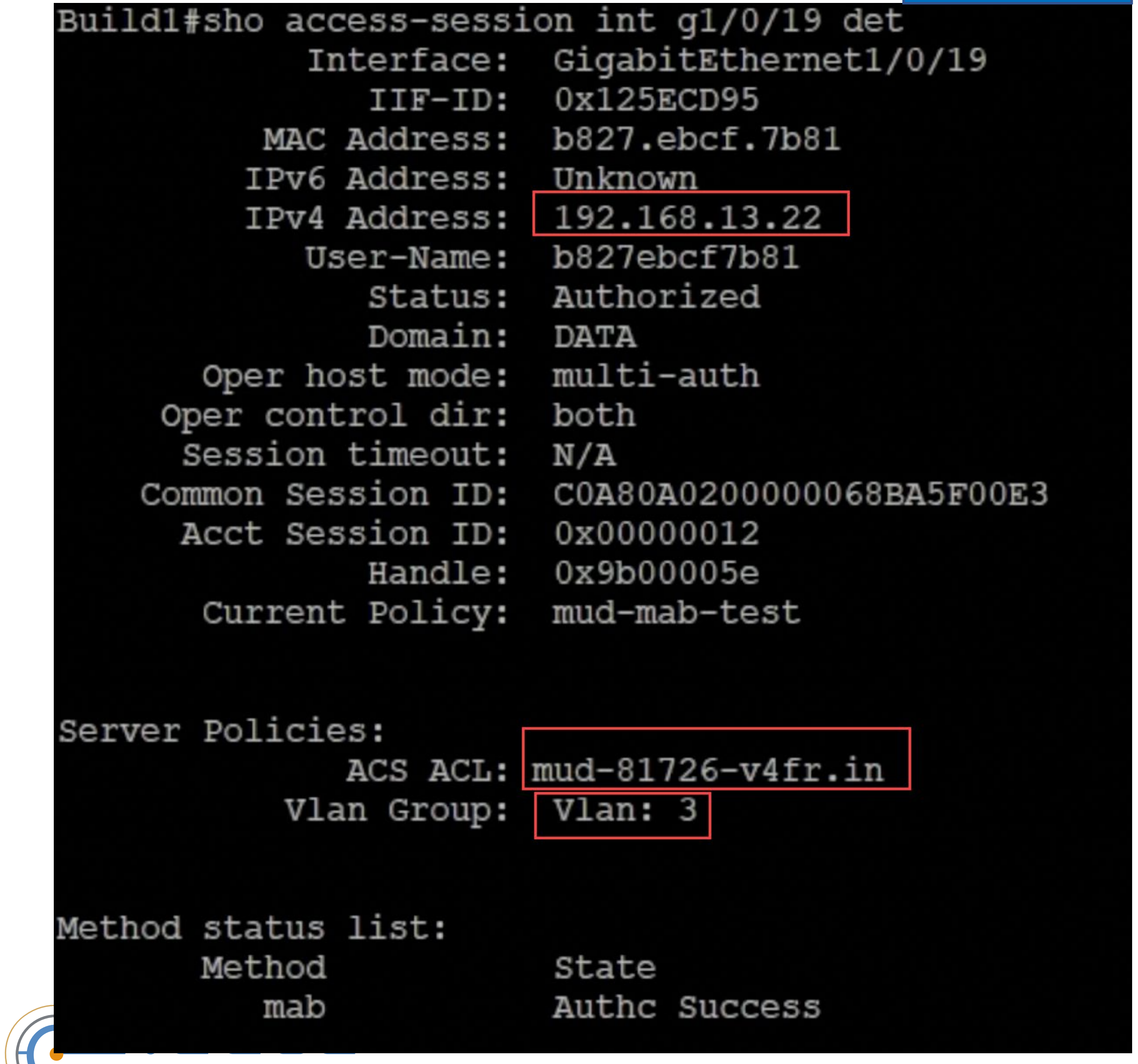


```
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 16
DHCPREQUEST of 192.168.13.22 on eth0 to 255.255.255.255 port 67
DHCPOFFER of 192.168.13.22 from 192.168.13.1
DHCPACK of 192.168.13.22 from 192.168.13.1
Too few arguments.
Too few arguments.
bound to 192.168.13.22 -- renewal in 19835 seconds.
pi@raspberrypi:~ $
```

Step 6: IP address assigned

1. Show access-session

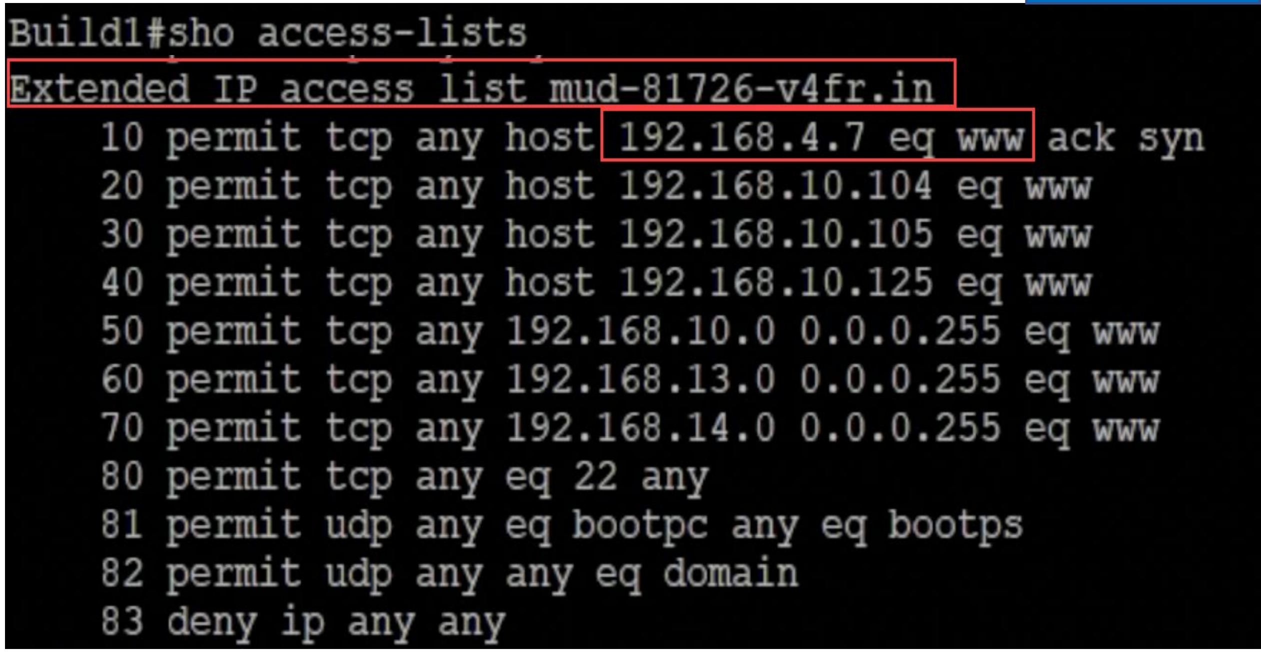
Router or Switch



```
Build1#sho access-session int g1/0/19 det
            Interface:  GigabitEthernet1/0/19
               IIF-ID:  0x125ECD95
          MAC Address:  b827.ebcf.7b81
         IPv6 Address:  Unknown
         IPv4 Address:  192.168.13.22
            User-Name:  b827ebcf7b81
               Status:  Authorized
               Domain:  DATA
       Oper host mode:  multi-auth
     Oper control dir:  both
      Session timeout:  N/A
    Common Session ID:  C0A80A0200000068BA5F00E3
      Acct Session ID:  0x00000012
               Handle:  0x9b00005e
       Current Policy:  mud-mab-test


Server Policies:
              ACS ACL: mud-81726-v4fr.in
           Vlan Group:  Vlan: 3


Method status list:
       Method           State
          mab           Authc Success
```

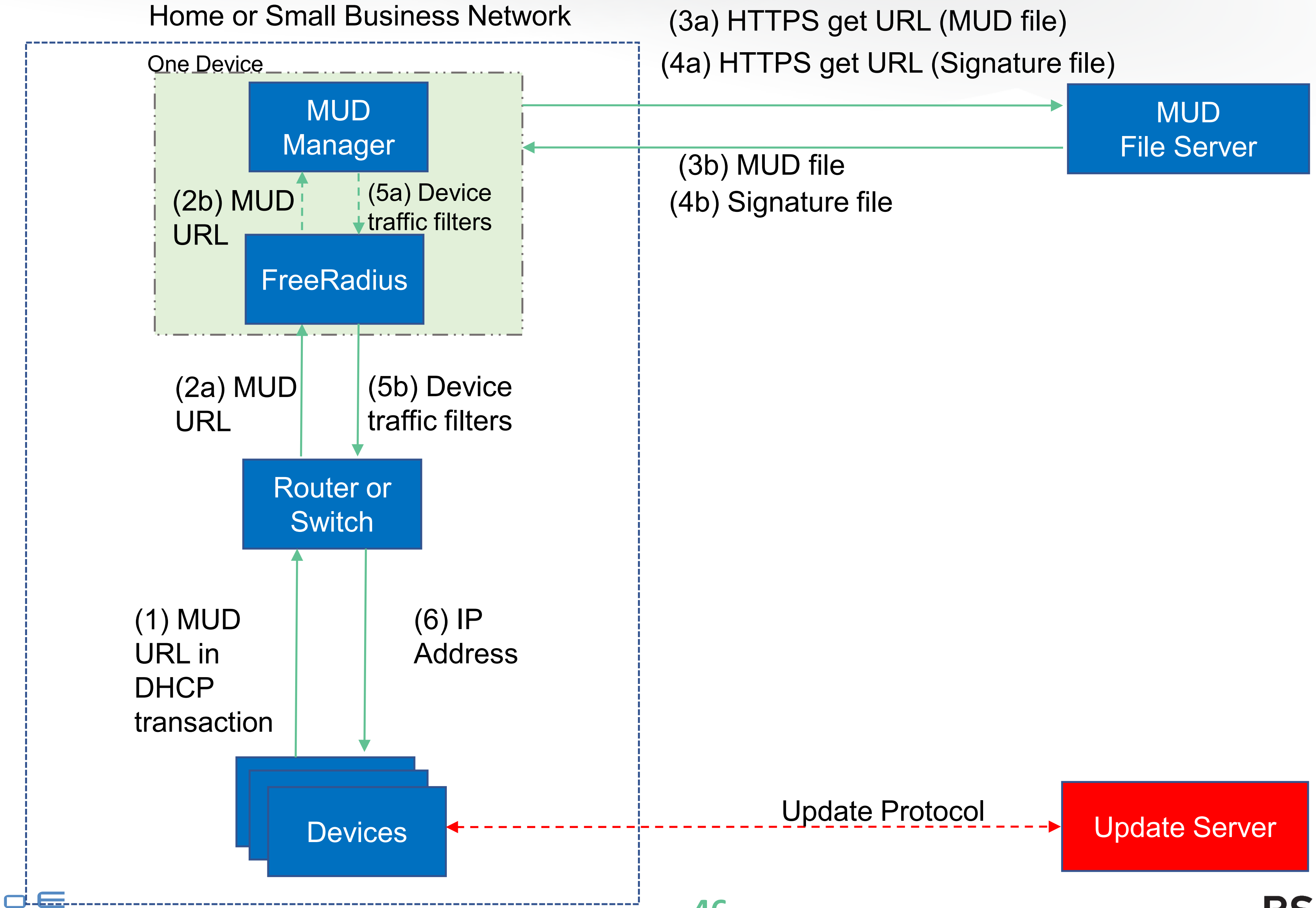
2. Show access-lists

Router or Switch



```
Build1#sho access-lists
Extended IP access list mud-81726-v4fr.in
    10 permit tcp any host 192.168.4.7 eq www ack syn
    20 permit tcp any host 192.168.10.104 eq www
    30 permit tcp any host 192.168.10.105 eq www
    40 permit tcp any host 192.168.10.125 eq www
    50 permit tcp any 192.168.10.0 0.0.0.255 eq www
    60 permit tcp any 192.168.13.0 0.0.0.255 eq www
    70 permit tcp any 192.168.14.0 0.0.0.255 eq www
    80 permit tcp any eq 22 any
    81 permit udp any eq bootpc any eq bootps
    82 permit udp any any eq domain
    83 deny ip any any
```

Step 7: Test communication

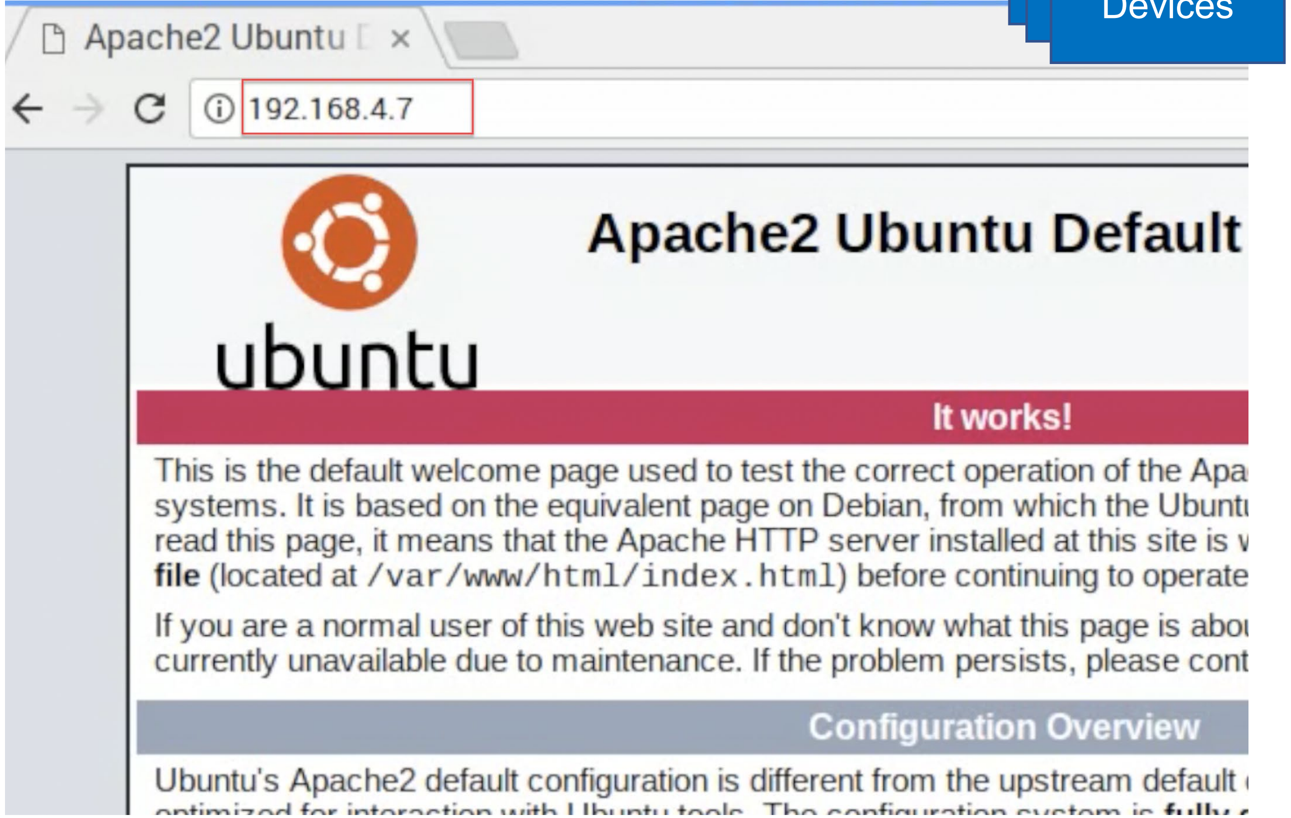

Home or Small Business Network
One Device
MUD Manager
(2b) MUD URL
(5a) Device traffic filters
FreeRadius
(2a) MUD URL
(5b) Device traffic filters
Router or Switch
(1) MUD URL in DHCP transaction
(6) IP Address
Devices
(3a) HTTPS get URL (MUD file)
(4a) HTTPS get URL (Signature file)
MUD File Server
(3b) MUD file
(4b) Signature file
Update Protocol
Update Server

Step 7: Test communication

RSAC Sandbox

1. Test browsing to “Update Server”

Devices



Apache2 Ubuntu [×

192.168.4.7

Apache2 Ubuntu Default

ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apa systems. It is based on the equivalent page on Debian, from which the Ubunt read this page, it means that the Apache HTTP server installed at this site is v **file** (located at `/var/www/html/index.html`) before continuing to operate

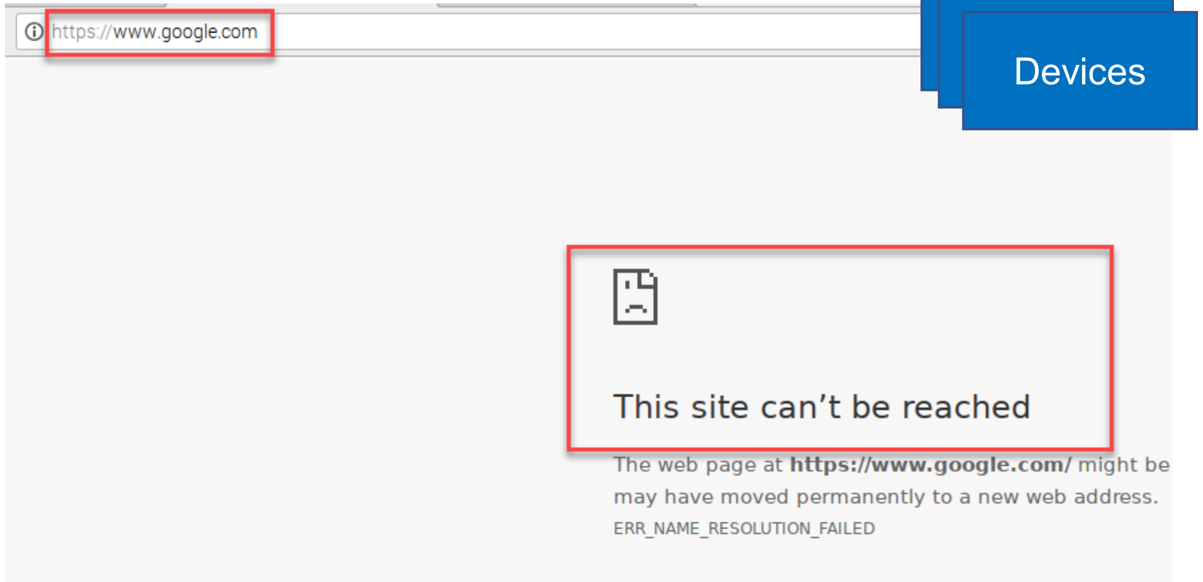
If you are a normal user of this web site and don't know what this page is abou currently unavailable due to maintenance. If the problem persists, please cont

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default optimized for interaction with Ubuntu tools. The configuration system is fully

2. Test browsing to unapproved server

Devices



https://www.google.com

This site can’t be reached

The web page at **https://www.google.com/** might be may have moved permanently to a new web address.

ERR_NAME_RESOLUTION_FAILED

Build 2 Demo Presentation

BUILD 1 – CISCO

- Cisco MUD Manager and FreeRadius
- Cisco Catalyst Switch
- NCCoE hosted MUD File Server
- MUD File
- DigiCert certificates
- MUD-capable IoT devices:
  - Molex PoE GW and Light Engine
  - Devkits
- Non-MUD-capable IoT devices
- NCCoE hosted Update Server
- NCCoE hosted Unapproved Server
- NCCoE hosted MQTT Broker Server
- Forescout and Forescout Enterprise Manager

PHASE 1

BUILD 2 – MASTERPEACE & GCA

- Yikes! Router including MUD Manager
- Yikes! Cloud & Yikes! Mobile App
- MasterPeace hosted MUD File Server
- MUD File
- DigiCert certificates
- MUD-capable IoT devices - Devkits
- Non-MUD-capable IoT devices
- NCCoE hosted Update Server
- NCCoE hosted Unapproved Server
- GCA Quad9 Threat Agent integrated into Yikes! Router
- GCA Quad9 Threat Signaling MUD Manager integrated into Yikes! Router
- GCA Quad9 Threat-Signaling DNS Services
- GCA Quad9 Threat API
- ThreatSTOP Threat MUD File Server
- ThreatSTOP Threat MUD File

BUILD 4 - NIST

- OpenDaylight SDN Controller including MUD Manager
- NCCoE hosted MUD File Server
- MUD File
- Wireless SDN Switch
- DigiCert certificates
- MUD-capable IoT devices - Devkits
- Non-MUD-capable IoT devices
- NCCoE hosted Unapproved Server
- Approved Server

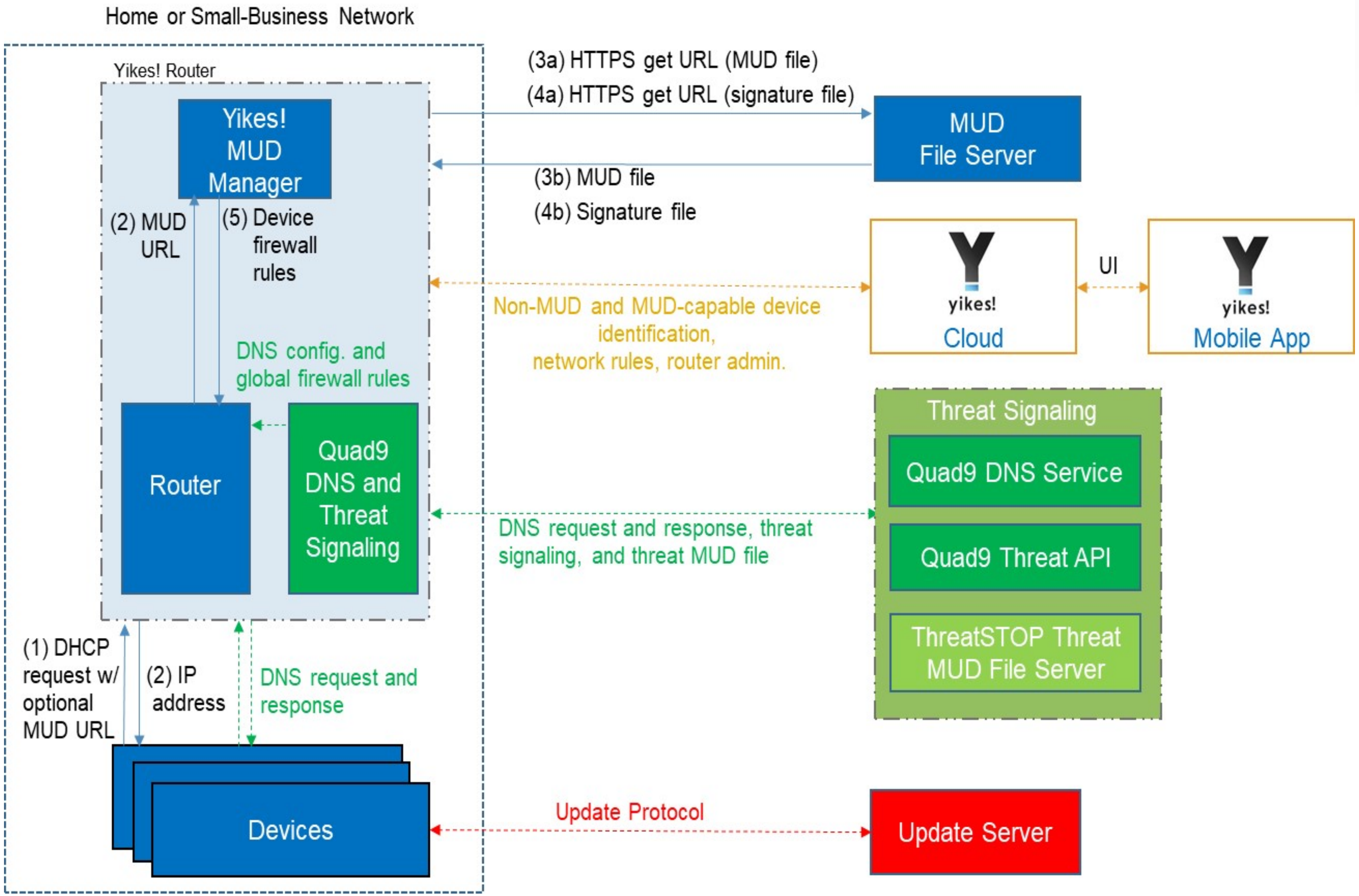
PHASE 2

BUILD 3 - CABLELABS

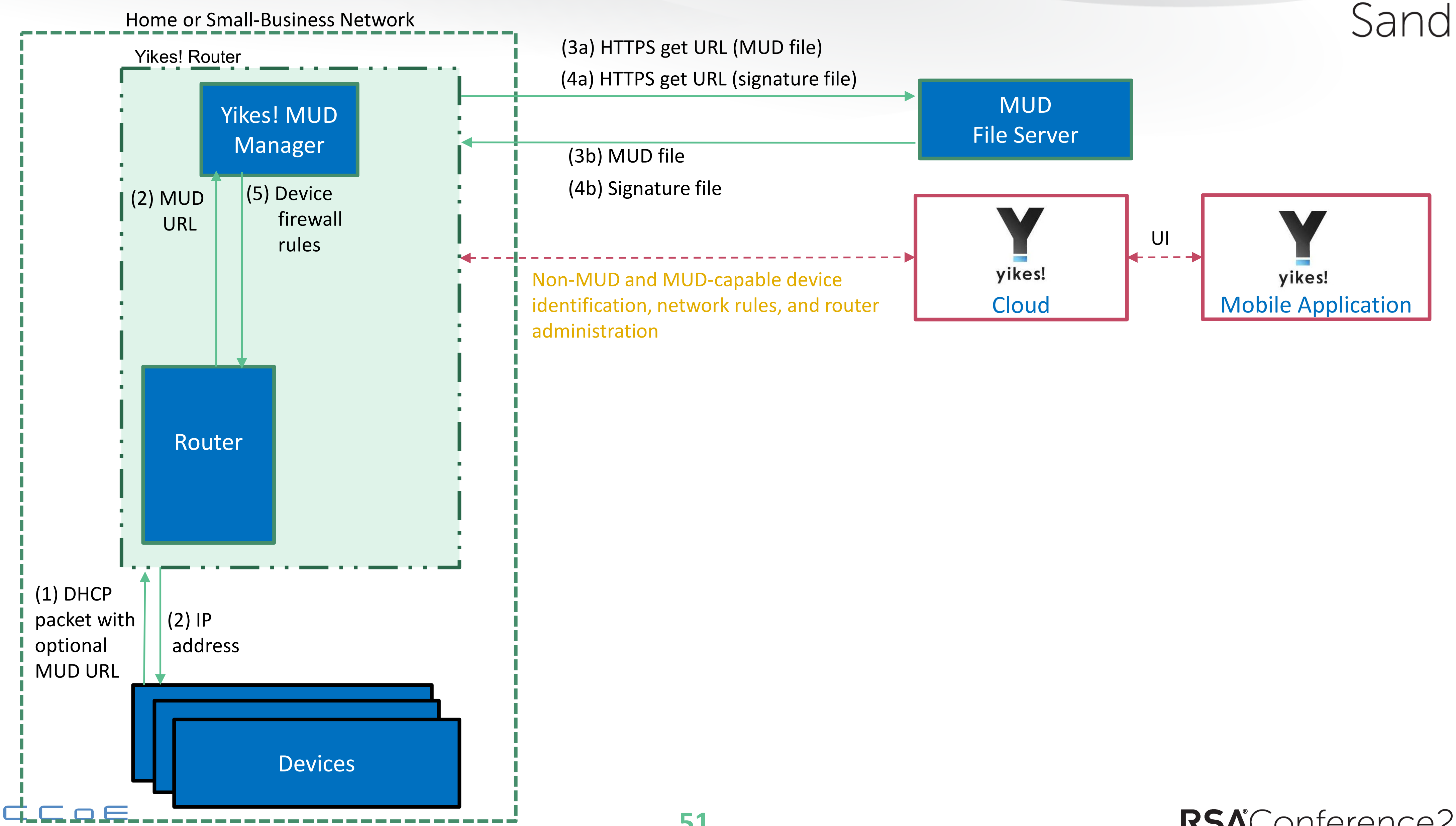
- Micronets Gateway
- Micronets Manager
- MUD Manager
- MUD Registry
- MSO Portal
- MUD File Server
- MUD File
- DigiCert certificates
- MUD-capable IoT devices - Devkits
- Non-MUD-capable IoT devices
- Update Server
- Unapproved Server
- Micronets Mobile App

PHASE 3

Logical Architecture – Build 2


Home or Small-Business Network
Yikes! Router
Yikes! MUD Manager
(2) MUD URL
(5) Device firewall rules
DNS config. and global firewall rules
Router
Quad9 DNS and Threat Signaling
(1) DHCP request w/ optional MUD URL
(2) IP address
DNS request and response
Devices
(3a) HTTPS get URL (MUD file)
(4a) HTTPS get URL (signature file)
MUD File Server
(3b) MUD file
(4b) Signature file
Non-MUD and MUD-capable device identification, network rules, router admin.
yikes! Cloud
UI
yikes! Mobile App
Threat Signaling
Quad9 DNS Service
Quad9 Threat API
ThreatSTOP Threat MUD File Server
DNS request and response, threat signaling, and threat MUD file
Update Protocol
Update Server

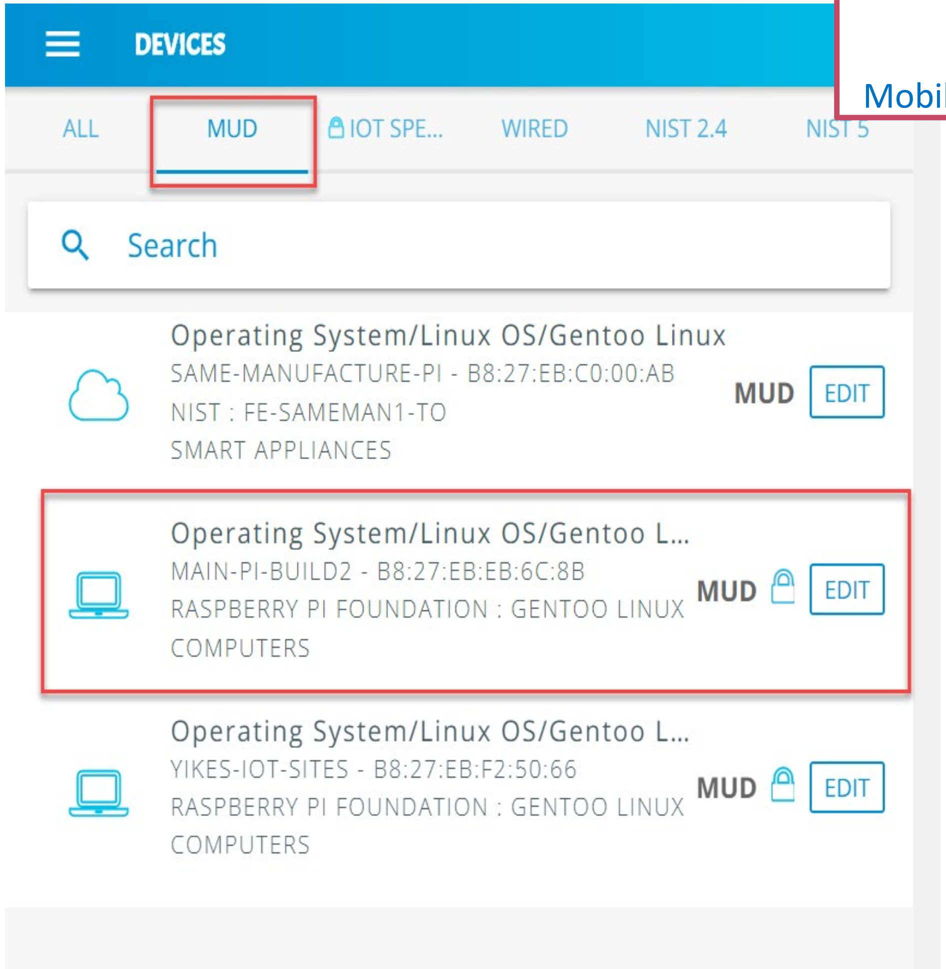
Step 1-5: Processing, applying, and viewing MUD File Rules


Home or Small-Business Network
Yikes! Router
Yikes! MUD Manager
(2) MUD URL
(5) Device firewall rules
Router
(1) DHCP packet with optional MUD URL
(2) IP address
Devices
(3a) HTTPS get URL (MUD file)
(4a) HTTPS get URL (signature file)
MUD File Server
(3b) MUD file
(4b) Signature file
Non-MUD and MUD-capable device identification, network rules, and router administration
yikes!
Cloud
UI
yikes!
Mobile Application

Step 1-5: Processing, applying, and viewing MUD File Rules

1. View all MUD devices on network

yikes!
Mobile Application



DEVICES

ALL | MUD | IOT SPE... | WIRED | NIST 2.4 | NIST 5

Search

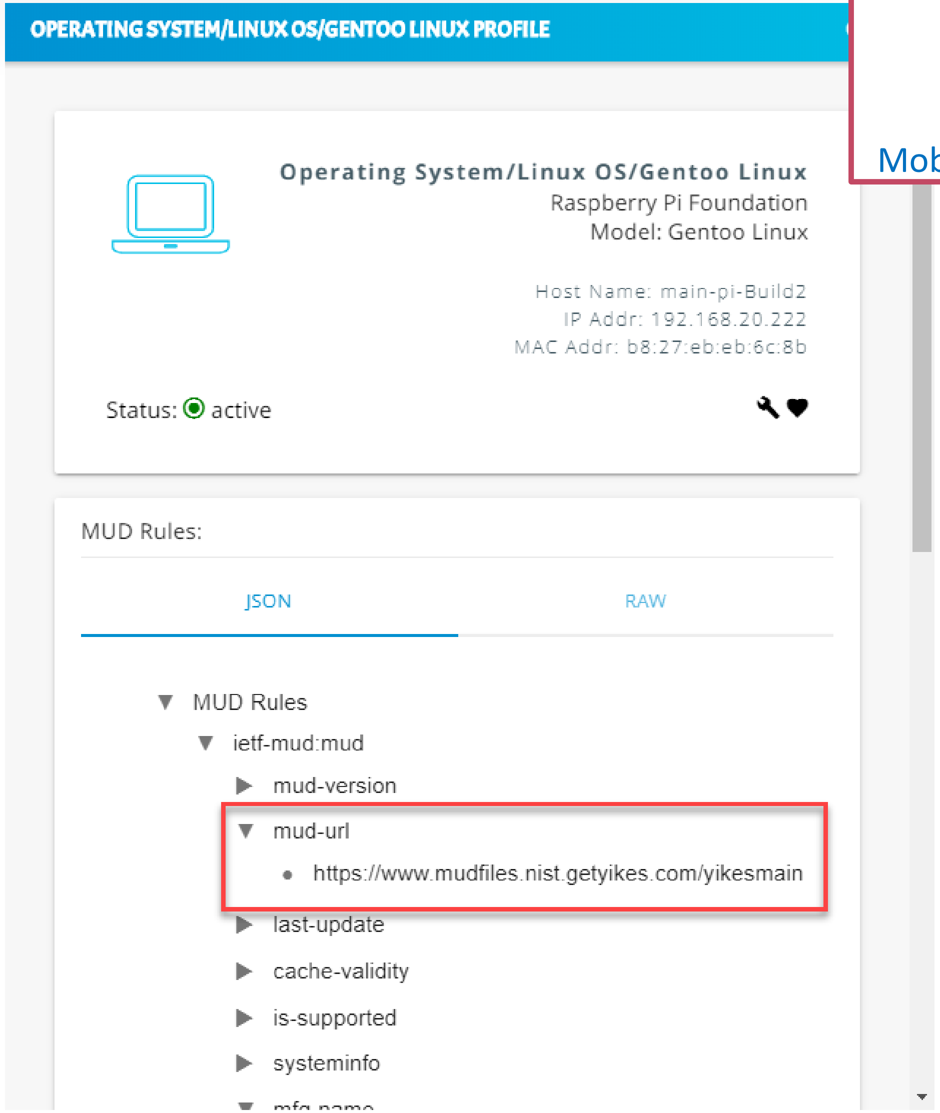
Operating System/Linux OS/Gentoo Linux
SAME-MANUFACTURE-PI - B8:27:EB:C0:00:AB
NIST : FE-SAMEMAN1-TO
SMART APPLIANCES
MUD EDIT

Operating System/Linux OS/Gentoo L...
MAIN-PI-BUILD2 - B8:27:EB:EB:6C:8B
RASPBERRY PI FOUNDATION : GENTOO LINUX
COMPUTERS
MUD EDIT

Operating System/Linux OS/Gentoo L...
YIKES-IOT-SITES - B8:27:EB:F2:50:66
RASPBERRY PI FOUNDATION : GENTOO LINUX
COMPUTERS
MUD EDIT

2. Expand device's profile

yikes!
Mobile Application



OPERATING SYSTEM/LINUX OS/GENTOO LINUX PROFILE

Operating System/Linux OS/Gentoo Linux
Raspberry Pi Foundation
Model: Gentoo Linux

Host Name: main-pi-Build2
IP Addr: 192.168.20.222
MAC Addr: b8:27:eb:eb:6c:8b

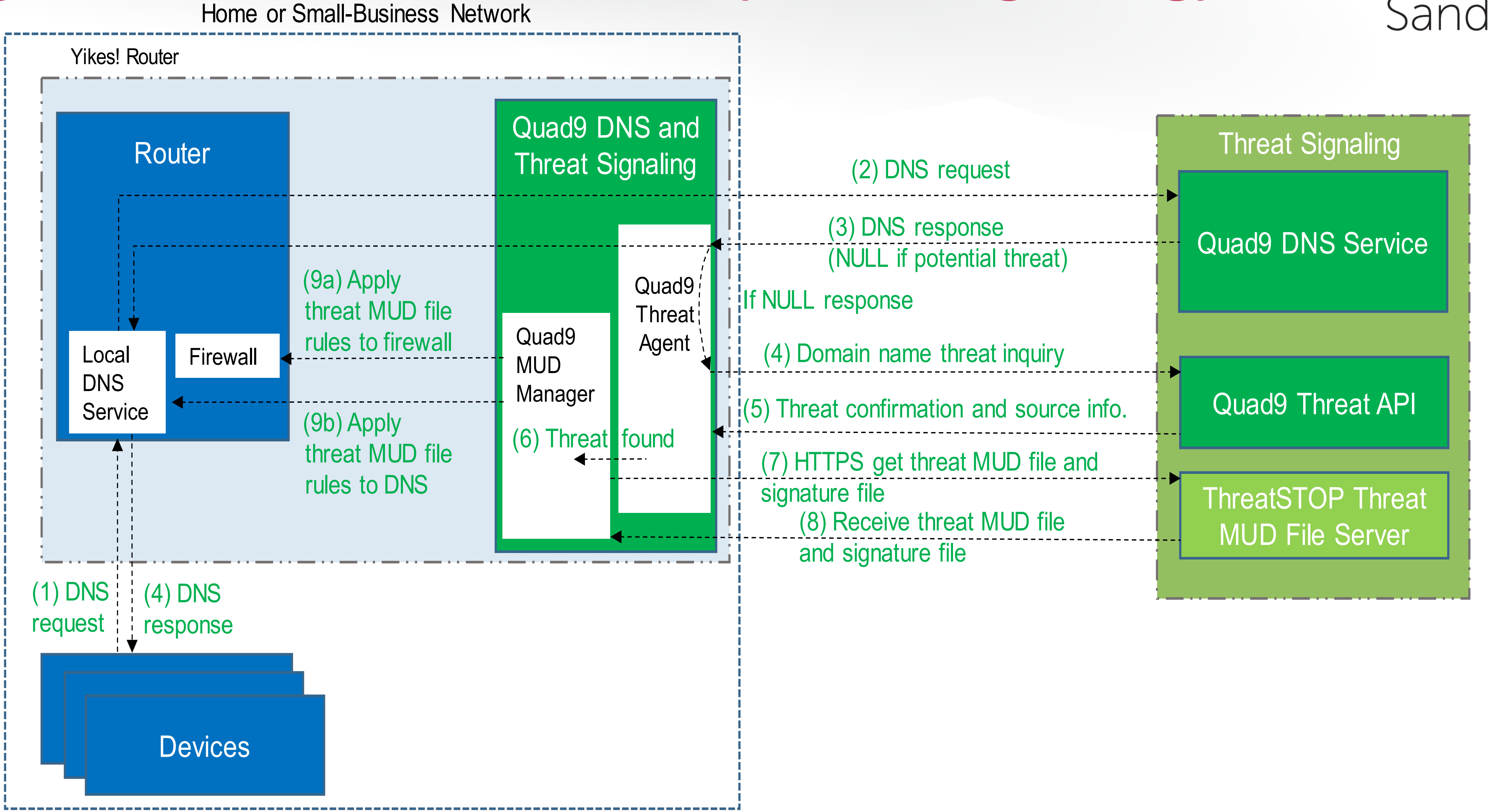
Status: ● active

MUD Rules:

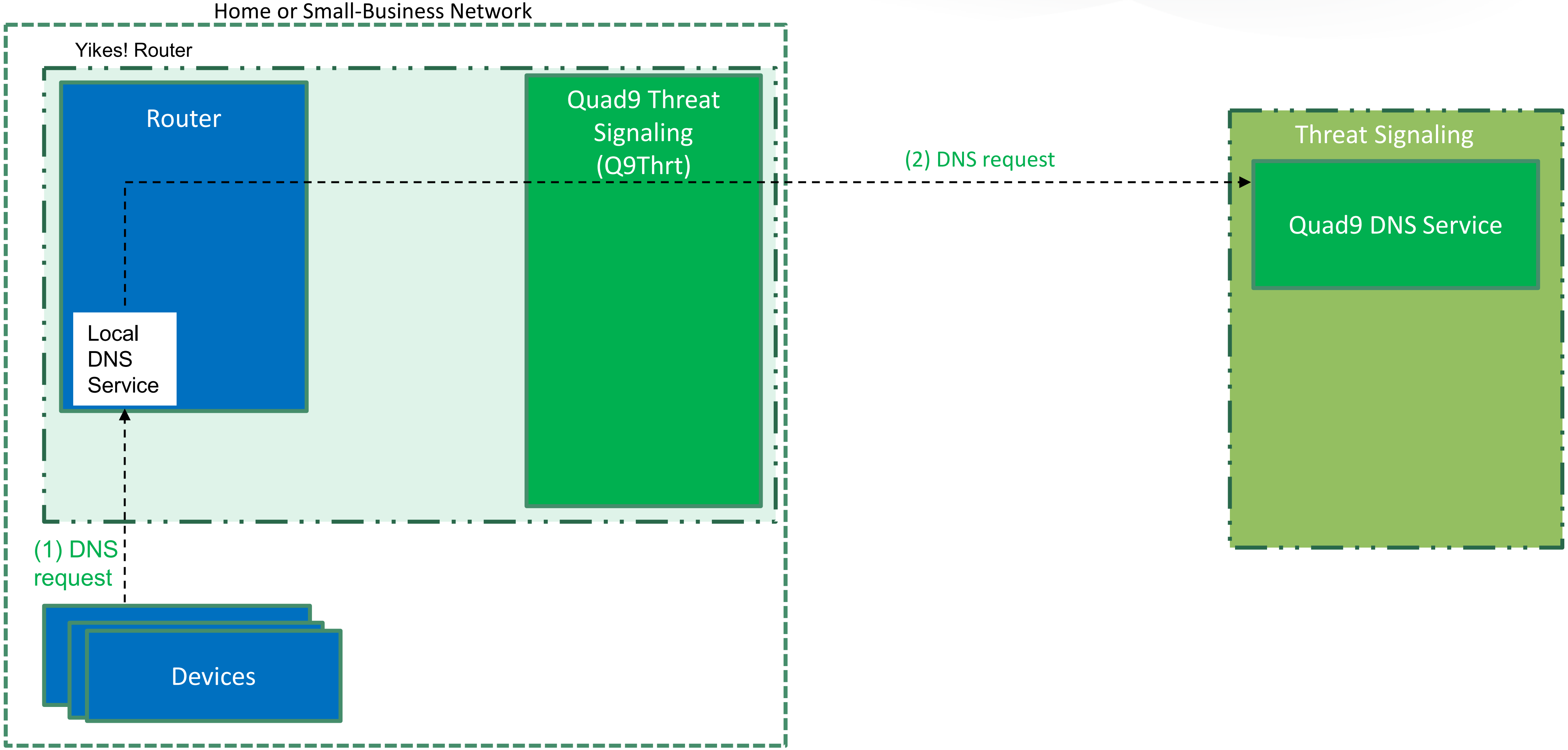
JSON | RAW

- ▼ MUD Rules
  - ▼ ietf-mud:mud
    - ▶ mud-version
    - ▼ mud-url
      - https://www.mudfiles.nist.getyikes.com/yikesmain
    - ▶ last-update
    - ▶ cache-validity
    - ▶ is-supported
    - ▶ systeminfo
    - ▼ mfg-name

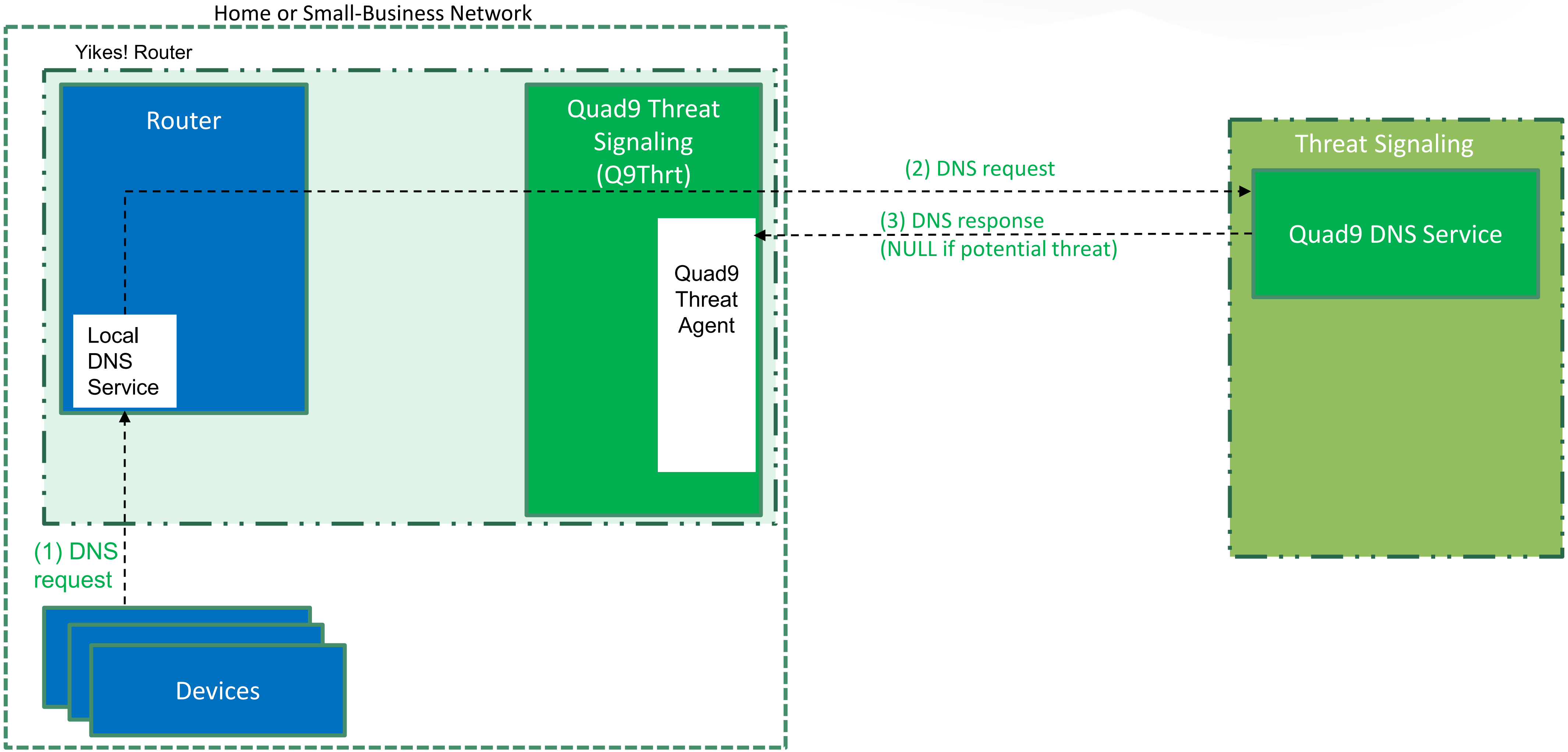
Logical Architecture – Build 2 (Threat Signaling)


Home or Small-Business Network
Yikes! Router
Router
Local DNS Service
Firewall
Quad9 DNS and Threat Signaling
Quad9 Threat Agent
Quad9 MUD Manager
(6) Threat found
(9a) Apply threat MUD file rules to firewall
(9b) Apply threat MUD file rules to DNS
Threat Signaling
Quad9 DNS Service
Quad9 Threat API
ThreatSTOP Threat MUD File Server
(2) DNS request
(3) DNS response (NULL if potential threat)
If NULL response
(4) Domain name threat inquiry
(5) Threat confirmation and source info.
(7) HTTPS get threat MUD file and signature file
(8) Receive threat MUD file and signature file
(1) DNS request
(4) DNS response
Devices

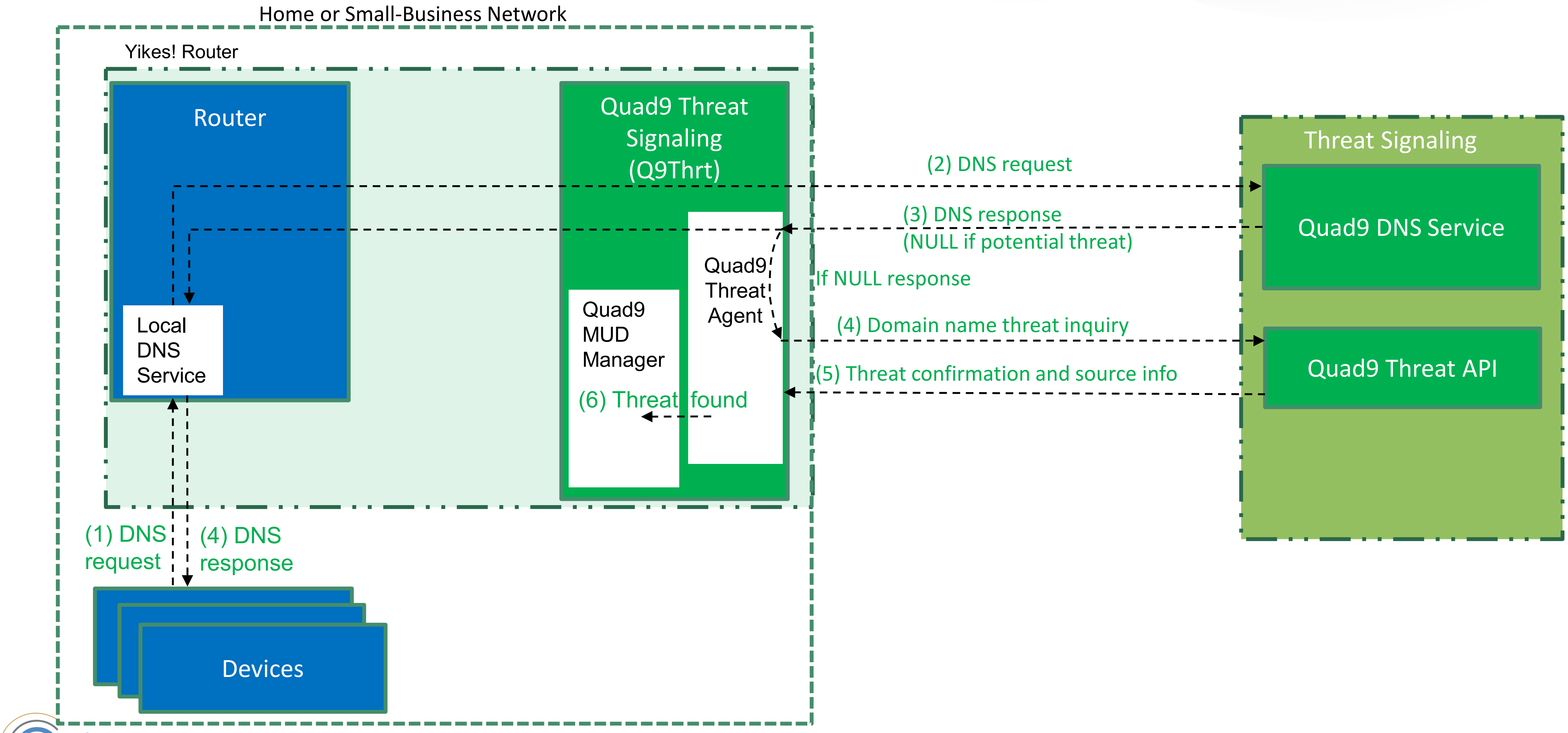
Step 1-2: Device attempts to communicate with compromised site



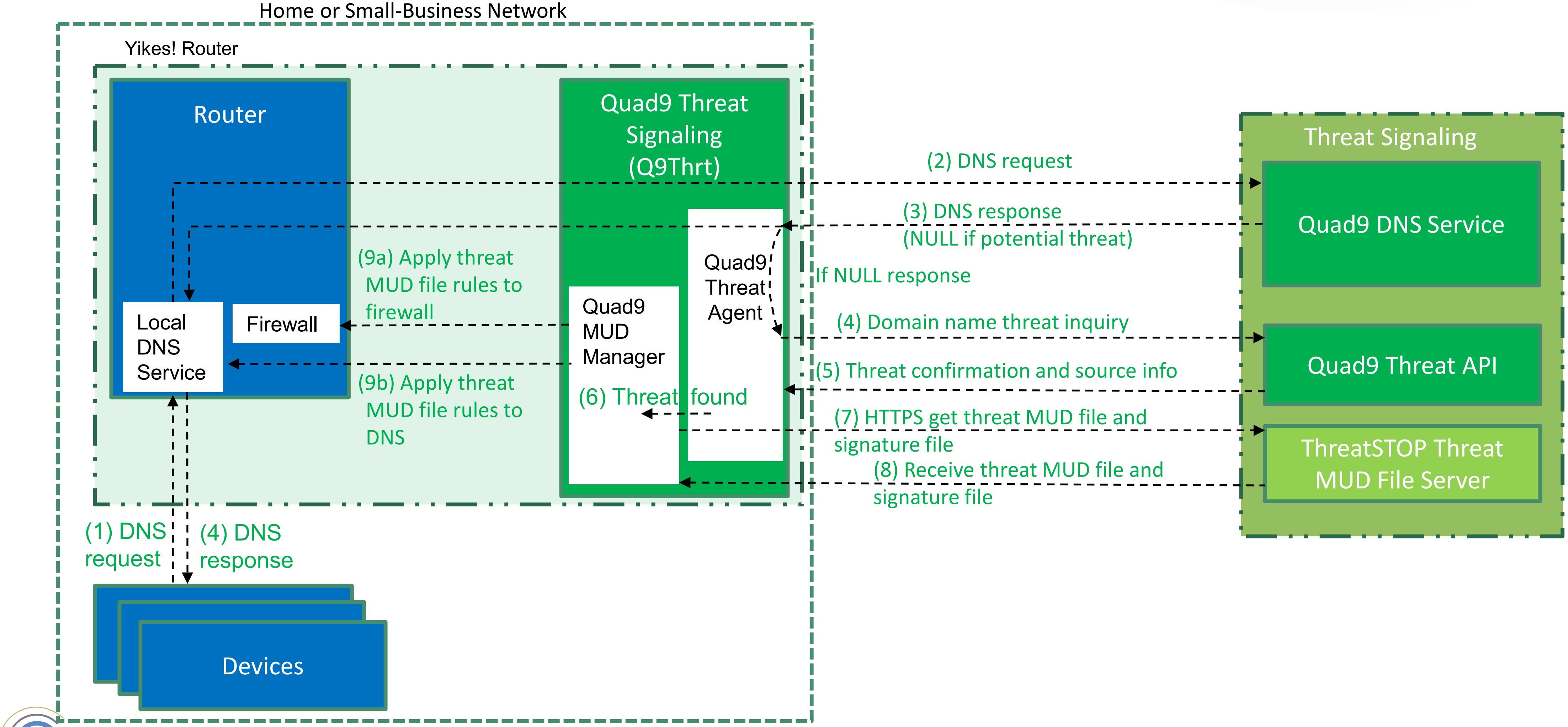
Step 3: Router receives DNS response from Quad9 DNS Service



Step 4-6: Threat found and local Quad9 MUD Manager notified



Step 7-9: Threat MUD file and signature file requested, verified, and applied on router



Home or Small-Business Network
Yikes! Router
Router
Local DNS Service
Firewall
Quad9 Threat Signaling (Q9Thrt)
Quad9 Threat Agent
Quad9 MUD Manager
(6) Threat found
(9a) Apply threat MUD file rules to firewall
(9b) Apply threat MUD file rules to DNS
(1) DNS request
(4) DNS response
Devices
Threat Signaling
Quad9 DNS Service
Quad9 Threat API
ThreatSTOP Threat MUD File Server
(2) DNS request
(3) DNS response (NULL if potential threat)
If NULL response
(4) Domain name threat inquiry
(5) Threat confirmation and source info
(7) HTTPS get threat MUD file and signature file
(8) Receive threat MUD file and signature file

NCCoE NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

Apply What You Have Learned Today

- Short term:
  - Review published Practice Guide for more details
  - Join NCCoE IoT MUD Community of Interest - mitigating-iot-ddos-nccoe@nist.gov
- Long term:
  - IoT device manufacturers, and network equipment manufacturers could implement MUD to improve the security of their products and of their customers' networks
  - IoT device users could purchase MUD-capable devices, when available, to protect their IoT devices from network-based attacks

https://www.nccoe.nist.gov/projects/building-blocks/mitigating-iot-based-ddos

Acknowledgements

- **NIST:**
  - Donna Dodson, Douglas Montgomery, Tim Polk, Mudumbai Ranganathan, Murugiah Souppaya
- **Arm:**
  - Allaukik Abhishek
- **CableLabs:**
  - Steve Johnson, Ashwini Kadam, Craig Pratt, Darshak Thakore, Mark Walker, Tao Wan
- **Cisco:**
  - Russ Gyurek, Eliot Lear, Peter Romness, Brian Weis
- **CTIA:**
  - Rob Cantu
- **Dakota Consulting:**
  - William Barker
- **DigiCert:**
  - Dean Coclin, Avesta Hojjati, Clint Wilson
- **Forescout**:
  - Katherine Gronberg, Tim Jones
- **Global Cyber Alliance:**
  - Adnan Baykal
- **MasterPeace Solutions:**
  - Drew Cohen, Nate Lesser, Kevin Yeich
- **The MITRE Corporation:**
  - Yemi Fashina, Joshua Harrington, Joshua Klosterman, Mary Raguso, Susan Symington, Paul Watrobski
- **Molex:**
  - Mo Alhroub, Jaideep Singh
- **Patton Electronics:**
  - Bryan Dubois, Stephen Ochs
- **Symantec:**
  - Matt Boucher, Bruce McCorkendale, Susanta Nanda, Yun Shen
- **Vigil Security:**
  - Russ Housley

Contact Information

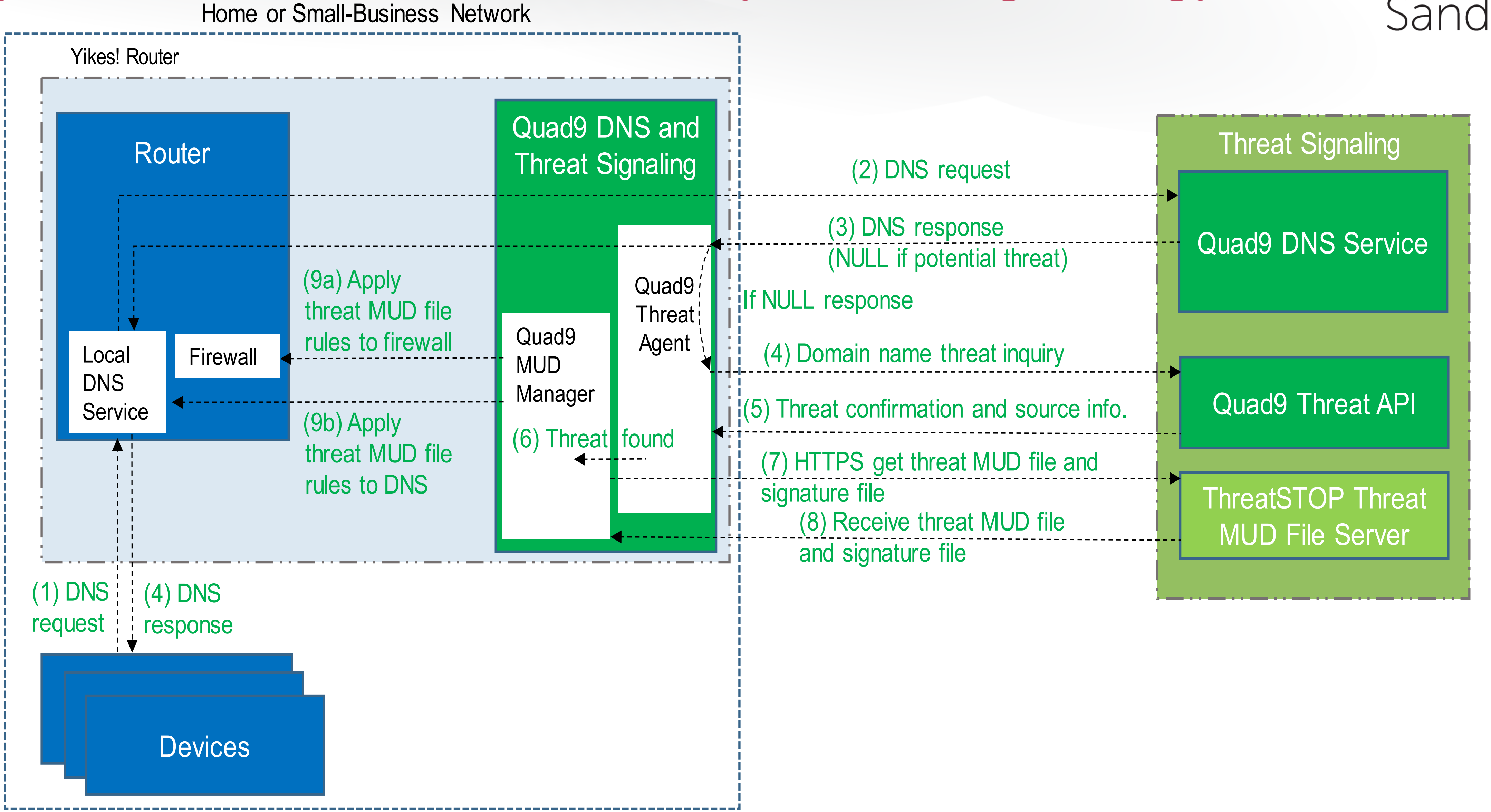
https://www.nccoe.nist.gov/projects/building-blocks/mitigating-iot-based-ddos

mitigating-iot-ddos-nccoe@nist.gov

Backups

Logical Architecture – Build 2 (Threat Signaling)

RSAC Sandbox



Home or Small-Business Network

Yikes! Router

Router

Local DNS Service

Firewall

Quad9 DNS and Threat Signaling

Quad9 MUD Manager

Quad9 Threat Agent

Devices

Threat Signaling

Quad9 DNS Service

Quad9 Threat API

ThreatSTOP Threat MUD File Server

(1) DNS request

(2) DNS request

(3) DNS response (NULL if potential threat)

If NULL response

(4) Domain name threat inquiry

(4) DNS response

(5) Threat confirmation and source info.

(6) Threat found

(7) HTTPS get threat MUD file and signature file

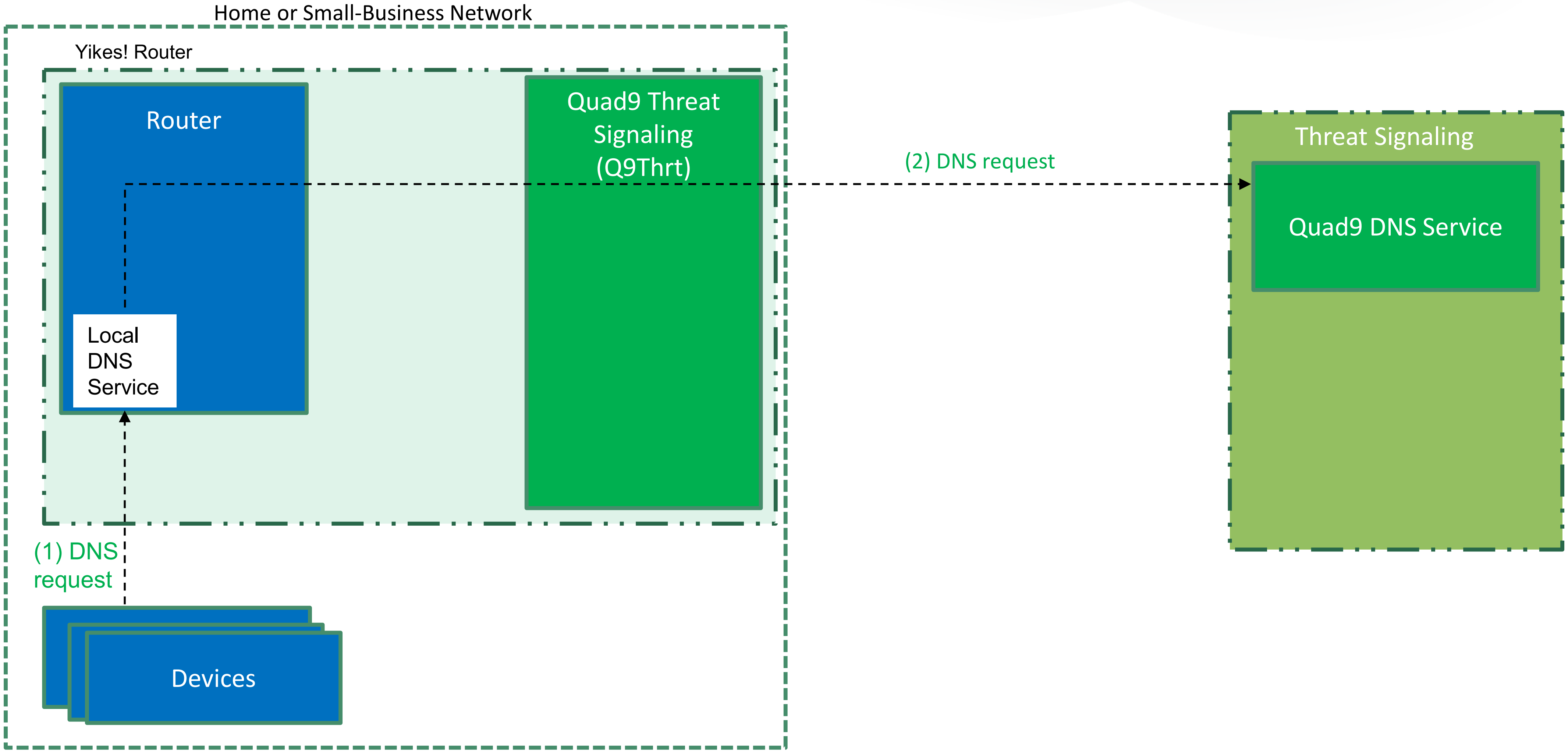
(8) Receive threat MUD file and signature file

(9a) Apply threat MUD file rules to firewall

(9b) Apply threat MUD file rules to DNS

NCCoE NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

Step 1-2: Device attempts to communicate with compromised site

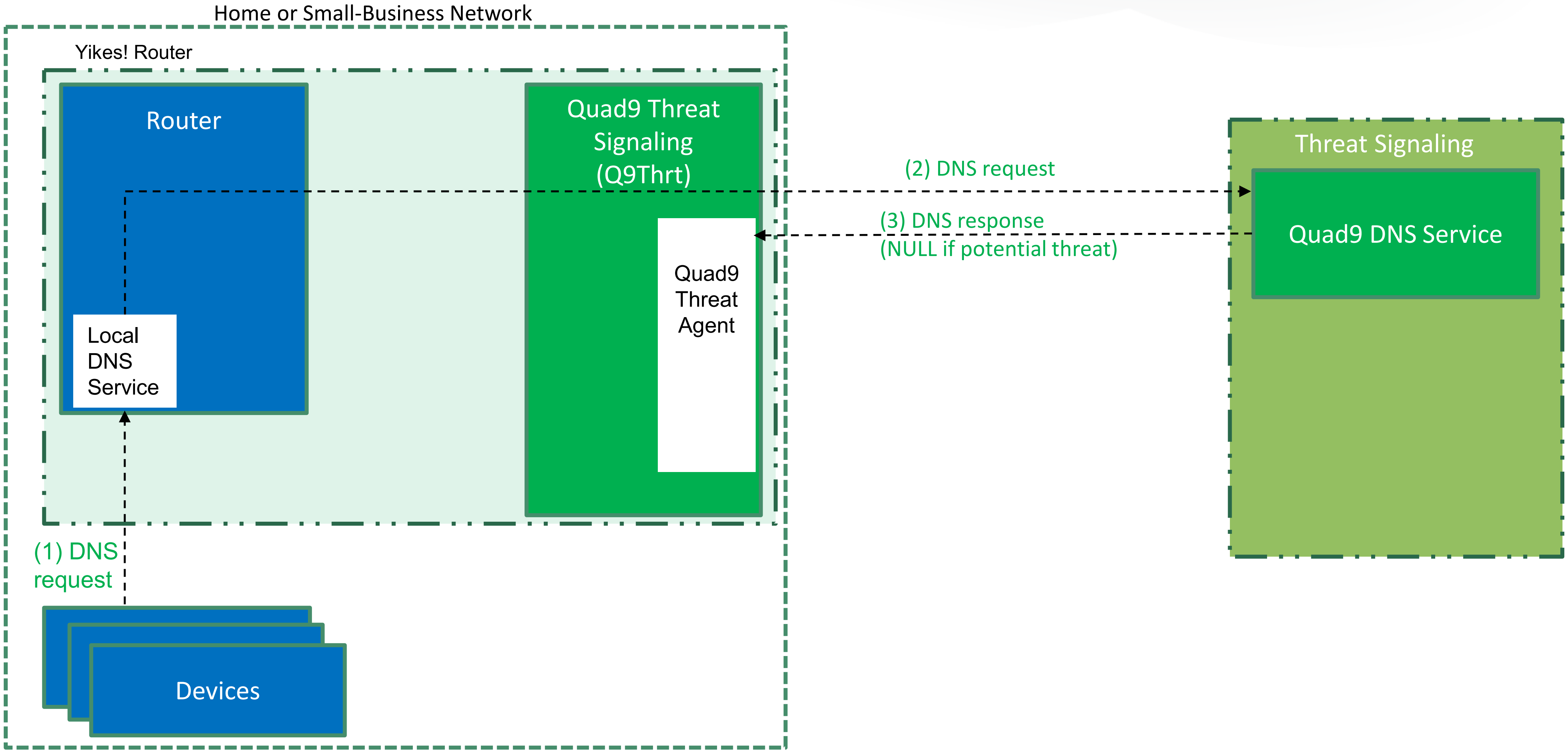


Step 1-2: Device attempts to communicate with compromised site

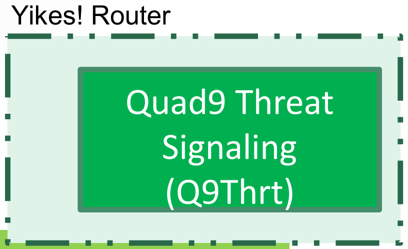
1. Device Pings known malicious host

```
$ ping www.dangerousSite.org
ping: cannot resolve www.dangerousSite.org: Unknown host
```

Step 3: Router receives DNS response from Quad9 DNS Service



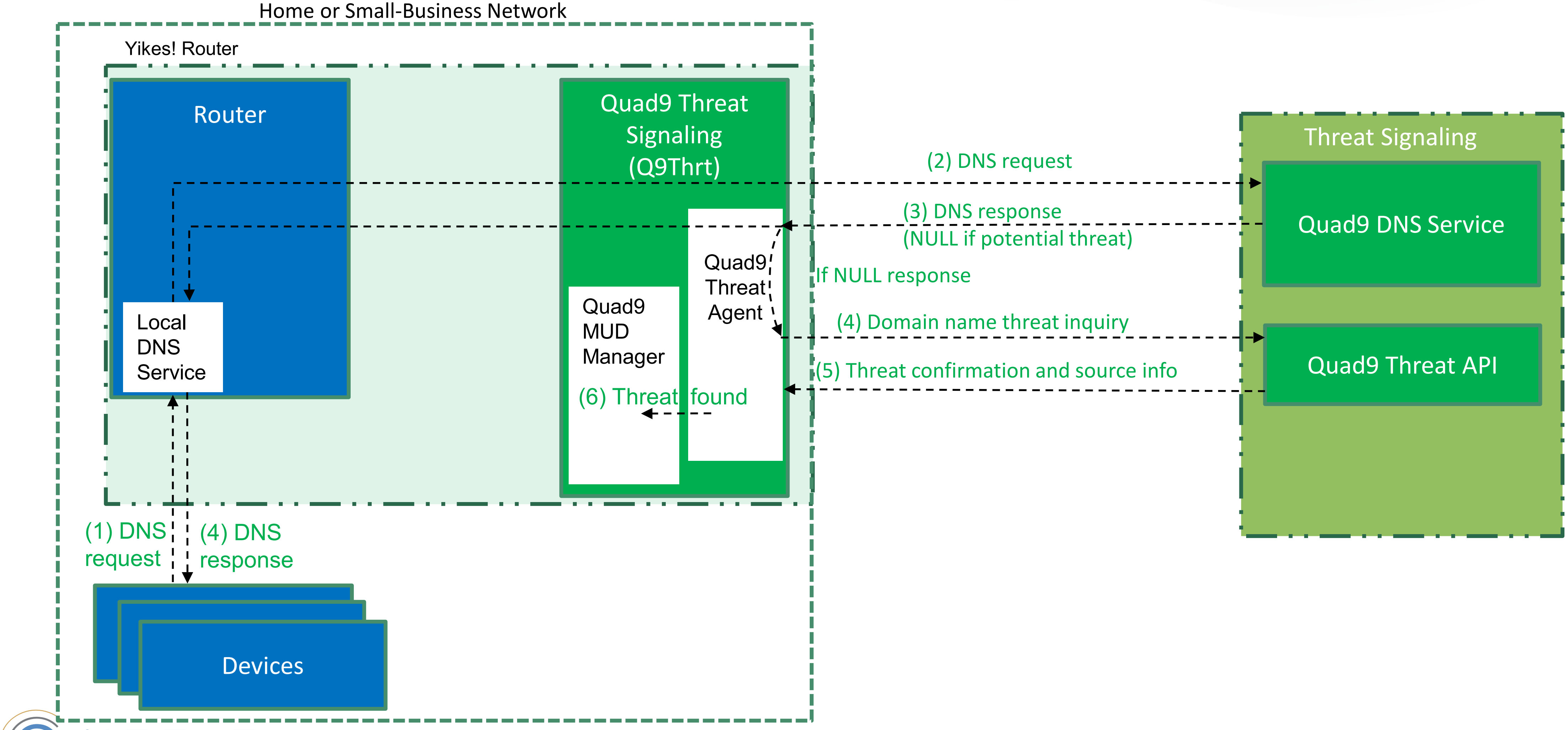
Step 3: Router receives DNS response from Quad9 DNS Service



1. Quad9 Agent receives DNS response

```
9.9.9.9.53 > 192.168.5.2.17847: [udp sum ok] 26864 NXDomain- q: A?
dangerousSite.org. 0/0/0 (29)
A? - dangerousSite.org - https://api.quad9.net/search
```

Step 4-6: Threat found and local Quad9 MUD Manager notified



Step 4-6: Threat found and local Quad9 MUD Manager notified

RSAC Sandbox

Yikes! Router

Quad9 Threat Signaling (Q9Thrt)

1. Quad9 Agent queries Quad9 API to confirm potential threat

```
DEBUG: runQuad9(): Calling Quad9 on:curl -s  -o /tmp/dangerousSite.org.q9
https://api.quad9.net/search/dangerousSite.org
DEBUG: runQuad9(): Search returned
{"domain":“dangerousSitre.org","blocked":true,"blocked_by":["threatstop"],"meta":[{"name":"ThreatSTOP
"}]} stored in /tmp/dangerousSite.org.q9
```

Yikes! Router

Quad9 Threat Signaling (Q9Thrt)

2. Quad9 Agent parses threat query response from Quad9 API and validates that site is blocked

```
DEBUG: runQuad9(): Download success via Quad9 threat API.
DEBUG: isQuad9Blocked(): Calling: jq  .blocked /tmp/dangerousSite.org.q9
https://api.quad9.net/search/dangerousSite.org
DEBUG: isQuad9Blocked(): Command result: true
DEBUG: isBlockedByProvider(): Calling: jq  -c .blocked_by /tmp/dangerousSite.org.q9
https://api.quad9.net/search/dangerousSite.org
DEUBG: isBlockedByProvider(): ["threatstop"] ---===--- threatstop
WARN: isBlockedByProvider(): Threat WAS FOUND TO BE BAD by threatstop
```

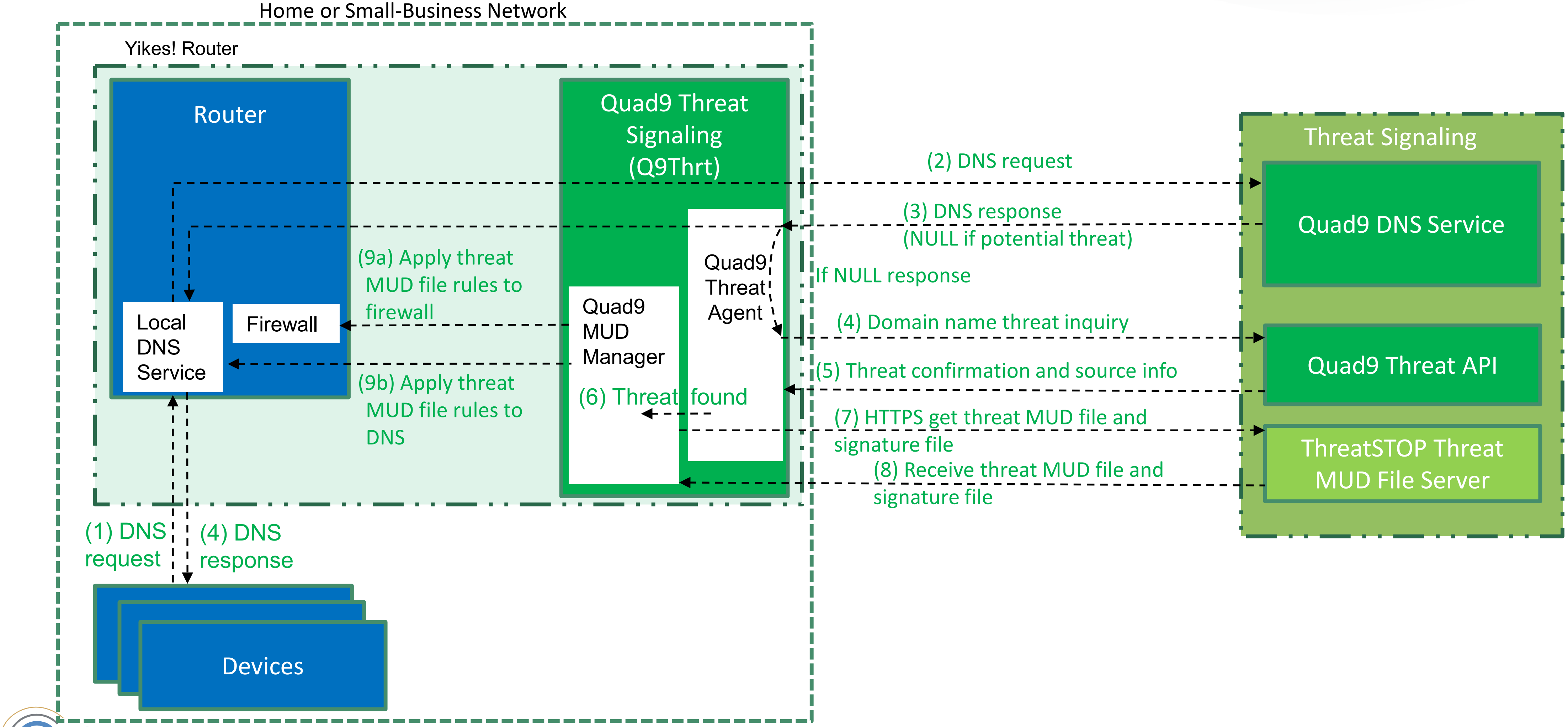
Yikes! Router

Quad9 Threat Signaling (Q9Thrt)

3. Quad9 Agent notifies Quad9 MUD Manager that threat has been found

```
DEBUG: runQuad9(): Threat provider threatstop: They found  to be bad. Call them now for more
detailed threat response information.
```

Step 7-9: Threat MUD file and signature file requested, verified, and applied on router

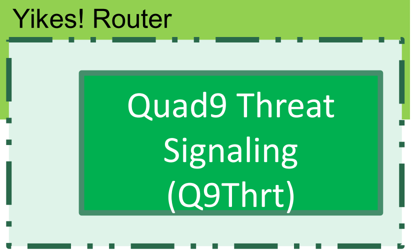


Home or Small-Business Network
Yikes! Router
Router
Local DNS Service
Firewall
Quad9 Threat Signaling (Q9Thrt)
Quad9 Threat Agent
Quad9 MUD Manager
(6) Threat found
(9a) Apply threat MUD file rules to firewall
(9b) Apply threat MUD file rules to DNS
(1) DNS request
(4) DNS response
Devices
Threat Signaling
Quad9 DNS Service
Quad9 Threat API
ThreatSTOP Threat MUD File Server
(2) DNS request
(3) DNS response (NULL if potential threat)
If NULL response
(4) Domain name threat inquiry
(5) Threat confirmation and source info
(7) HTTPS get threat MUD file and signature file
(8) Receive threat MUD file and signature file

Step 7-9: Threat MUD file and signature file requested, verified, and applied on router

1. Quad9 MUD Manager requests Threat MUD and Signature file and validates respectively

```
DEBUG: retrieveThreatProviderFile(): Calling: curl -s  -o /etc/q9thrt/state/mudfiles/dangerousSite.org.json
https://mud.threatstop.com/dangerousSite.org.json
INFO: retrieveThreatProviderFile(): MUD FILE RETRIEVED
DEBUG: retrieveThreatProviderFile(): Calling: curl -s  -o /etc/q9thrt/state/mudfiles/dangerousSite.org.p7s
https://mud.threatstop.com/dangerousSite.org.p7s
INFO: retrieveThreatProviderFile(): SIGNATURE FILE RETRIEVED
DEBUG: testMudFile(): Calling: jq  -r '.["ietf-mud:mud"]["mud-version"]'
/etc/q9thrt/state/mudfiles/dangerousSite.org.json
DEBUG: testMudFile(): Valid Mud file: MudVersion = 1
DEBUG: testMudFileSignature(): Calling: openssl asn1parse -in /etc/q9thrt/state/mudfiles/dangerousSite.org.p7s
-inform der | grep -i error | wc -l
DEBUG: testMudFileSignature(): Valid Mud file signature.
DEBUG: validateThreatMudFile(): Both the MUD file and MUD p7s signature files are valid. Now test signature.
DEBUG: validateThreatMudFile(): Calling: openssl cms -verify -in
/etc/q9thrt/state/mudfiles/dangerousSite.org.p7s -inform DER -content
/etc/q9thrt/state/mudfiles/dangerousSite.org.json > /dev/null
INFO: validateThreatMudFile(): MUD FILE SIGNATURE PASSED
```



Step 7-9: Threat MUD file and signature file requested, verified, and applied on router

2. Quad9 MUD Manager builds and applies policies to local DNS and Firewall

Yikes! Router

Quad9 Threat Signaling (Q9Thrt)

```
DEBUG: runQuad9(): Installing valid mud file:
/etc/q9thrt/state/mudfiles/dangerousSite.org.json
DEBUG: installMudFile(): Calling:
/etc/q9thrt/build_policies.sh -e dangerousSite.org -m
/etc/q9thrt/state/mudfiles/dangerousSite.org.json -s lan -d
wan -k /etc/q9thrt/state/rules
INFO: installMudFile(): MUD FILE INSTALLED
DEBUG: installMudFile(): Calling:
/etc/q9thrt/build_policies.sh -e dangerousSite.org -m
/etc/q9thrt/state/mudfiles/dangerousSite.org.json -s wan -d
lan -k /etc/q9thrt/state/rules
INFO: installMudFile(): MUD FILE INSTALLED
DEBUG: commitThreatConfiguration(): Calling:
/etc/q9thrt/commit_threat_rules.sh -d
/etc/q9thrt/state/rules -t /tmp/q9thrt_tmp_dir
```

3. Quad9 Firewall Rules

Yikes! Router

Router

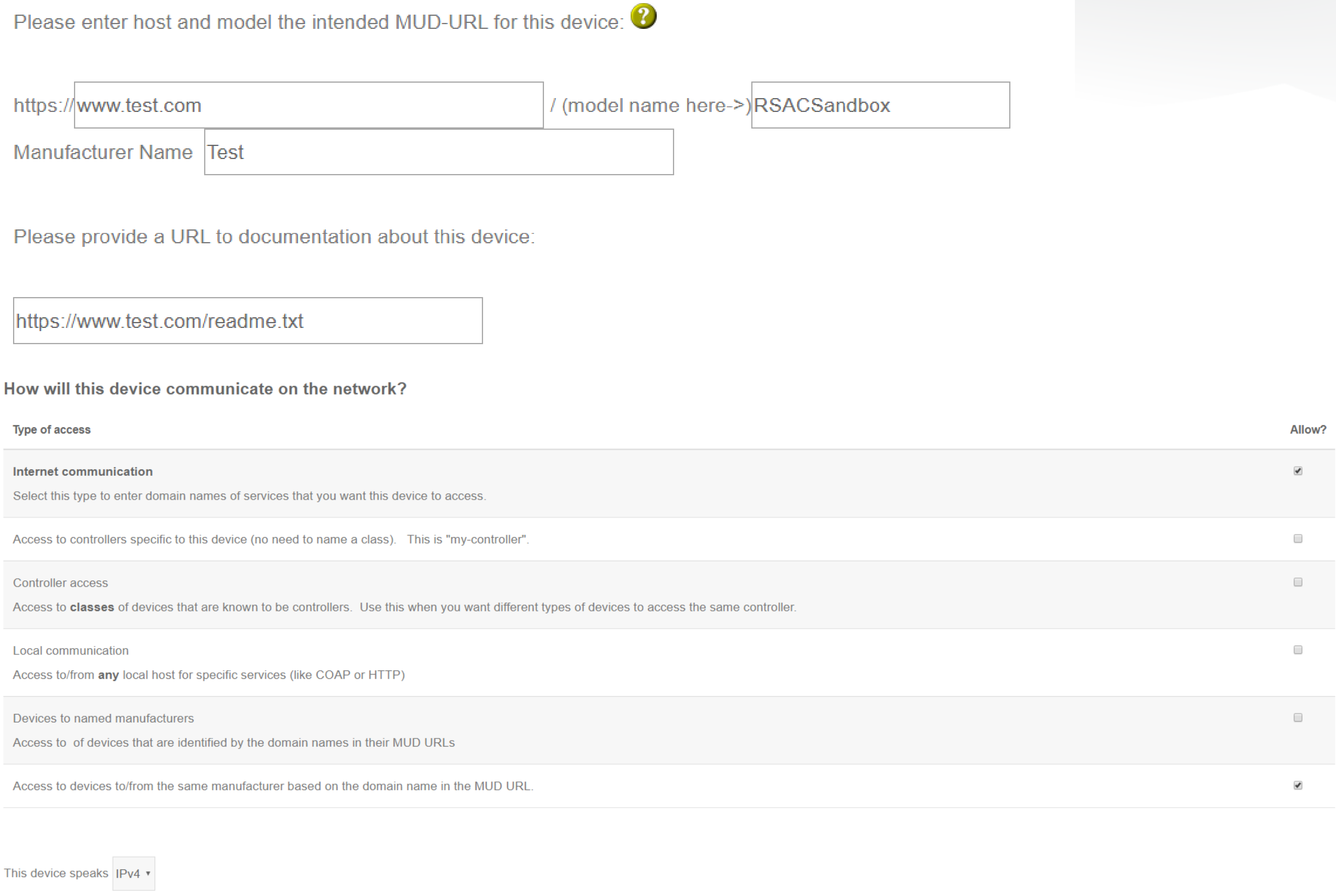
```
# Q9THREATRULES start
config ipset
    option enabled 1
    option name Q9TS-dangerousSite_orgFD
    option match dest_ip
    option storage hash
    option family ipv4
    option external Q9TS-dangerousSite_orgFD
config ipset
    option enabled 1
    option name Q9TS-dangerousSite_orgTD
    option match src_ip
    option storage hash
    option family ipv4
    option external Q9TS-dangerousSite_orgTD
 config rule
        option enabled   '1'
        option name      'Q9TS-dangerousSite_orgFD'
        option target    REJECT
        option src       lan
        option dest      wan
        option proto     all
        option family    ipv4
        option ipset     Q9TS-dangerousSite_orgFD
        option src_ip    any
 config rule
        option enabled   '1'
        option name      'Q9TS-dangerousSite_orgTD'
        option target    REJECT
        option src       wan
        option dest      lan
        option proto     all
        option family    ipv4
        option ipset     Q9TS-dangerousSite_orgTD
        option dest_ip    any
# Q9THREATRULES end
```

Step 7-9: Threat MUD file and signature file requested, verified, and applied on router

4. Device attempts to communicate with malicious host after rules are applied – DNS now resolves dangerousSite.org to IoT device loopback address

```
$ ping www.dangerousSite.org
PING www.dangerousSite.org(127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.049 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.073 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.082 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.139 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.079 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.072 ms
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.123 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.073 ms
ç64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.066 ms
^C
--- www.dangerousSite.org ping statistics ---
9 packets transmitted, 9 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.049/0.084/0.139/0.027 ms
```

MUD File maker - MUDMaker.org



Please enter host and model the intended MUD-URL for this device:

https://www.test.com / (model name here->) RSACSandbox

Manufacturer Name: Test

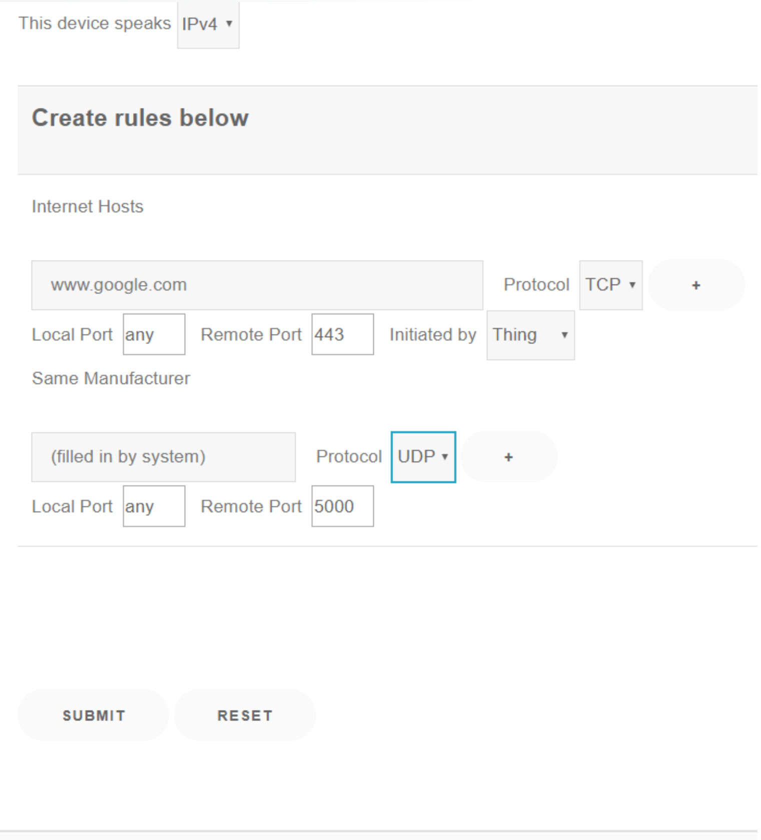
Please provide a URL to documentation about this device:

https://www.test.com/readme.txt

How will this device communicate on the network?

Type of access	Allow?
Internet communication Select this type to enter domain names of services that you want this device to access.	☑
Access to controllers specific to this device (no need to name a class). This is "my-controller".	□
Controller access Access to **classes** of devices that are known to be controllers. Use this when you want different types of devices to access the same controller.	□
Local communication Access to/from **any** local host for specific services (like COAP or HTTP)	□
Devices to named manufacturers Access to of devices that are identified by the domain names in their MUD URLs	□
Access to devices to/from the same manufacturer based on the domain name in the MUD URL.	☑

This device speaks: IPv4



This device speaks: IPv4

Create rules below

Internet Hosts

www.google.com — Protocol: TCP

Local Port: any — Remote Port: 443 — Initiated by: Thing

Same Manufacturer

(filled in by system) — Protocol: UDP

Local Port: any — Remote Port: 5000

Sample MUD File

```
{
 "ietf-mud:mud": {
  "mud-version": 1,
  "mud-url": "https://www.test.com/RSACSandbox",
  "last-update": "2020-01-14T19:45:00+00:00",
  "cache-validity": 48,
  "is-supported": true,
  "systeminfo": "Test MUD File for RSAC 2020",
  "mfg-name": "Test",
  "documentation": "https://www.test.com/readme.txt",
  "model-name": "RSACSandbox",
  "from-device-policy": {
   "access-lists": {
    "access-list": [
     {
      "name": "mud-33577-v4fr"
     }
    ]
   }
  },
  "to-device-policy": {
   "access-lists": {
    "access-list": [
     {
      "name": "mud-33577-v4to"
  [......]
```

```
[.....]
 "ietf-access-control-list:acls": {
  "acl": [
   {
    "name": "mud-33577-v4to",
    "type": "ipv4-acl-type",
    "aces": {
     "ace": [
      {
       "name": "cl0-todev",
       "matches": {
        "ipv4": {
         "ietf-acldns:src-dnsname": "www.google.com",
         "protocol": 6
        },
        "tcp": {
         "ietf-mud:direction-initiated": "from-device",
         "source-port": {
          "operator": "eq",
          "port": 443
         }
        }
       },
       "actions": {
        "forwarding": "accept"
       }
      },

      [....]
```