Automated Dynamic Firmware Analysis at Scale: A Case Study on Embedded Web Interfaces

Andrei Costin
andrei@firmware.re
@costinandrei
Embedded Devices: Smarted, More Complex

Connect

by Wilgengebroed on Flickr [CC-BY-2.0]
Embedded Devices: More Interconnected, More WWW

API

The Internet of Things

{JSON}

RESTful API
GET PUT POST DELETE

<?xml?>

CONNECT THE WORLD

WWW

by Wilgengebroed on Flickr [CC-BY-2.0]
By 2014, there were hundred thousands firmware packages (Costin et al., USENIX Security 2014)

By 2014, there were 14 billion Internet connected objects (Cisco, Internet of Things Connections Counter, 2014)

By 2020, there will be between 20 and 50 billion interconnected IoT/embedded devices (Cisco, The Internet of Everything in Motion, 2013)
- Large number of devices
- Large number of firmware files
- Highly heterogeneous systems
- Increasingly “smart”, “connected”
- Highly unstructured firmware data
- Vulnerable devices exposed
● Large number of devices → Analysis without devices
● Large number of firmware files → Scalable architectures
● Highly heterogeneous systems → Generic techniques
● Increasingly “smart”, “connected” → Focus on web interfaces & APIs
● Highly unstructured firmware data → Large dataset classification
● Vulnerable devices exposed → Technology-independent device fingerprinting
Scalable Framework: Dynamic Firmware Analysis
Scalable Framework: Dynamic Firmware Analysis
Scalable Framework: Dynamic Firmware Analysis

- Unpacked Firmware Sources
- Firmware Selection
- File Systems Preparation
- Scalable Cloud VM Infrastructure
  - Dynamic Analysis
  - QEMU/Chroot Analysis Tools
  - Static Analysis
  - Doc Root Analysis
Scalable Framework: Dynamic Firmware Analysis

Unpacked Firmware Sources → Firmware Selection → File Systems Preparation

Scalable Cloud VM Infrastructure
- Dynamic Analysis
- QEMU/Chroot → Analysis Tools
- Static Analysis
- Doc Root Analysis

Results Collection and Analysis
Scalable Framework: Dynamic Firmware Analysis

Unpacked Firmware Sources → Firmware Selection → File Systems Preparation → Scalable Cloud VM Infrastructure

Feedback for Improving Analysis:
- Dynamic Analysis
  - QEMU/Chroot
  - Analysis Tools
- Static Analysis
  - Doc Root Analysis

Results Collection and Analysis → Results Exploitation
Scalable Framework: Dynamic Firmware Analysis

Unpacked Firmware Sources ➔ Firmware Selection ➔ File Systems Preparation ➔ Feedback for Improving Analysis ➔ Scalable Cloud VM ➔ Results Collection and Analysis ➔ Results Exploitation

- Dynamic Analysis
  - QEMU/Chroot
  - Static Analysis
  - Doc Root Analysis
Embedded Devices Emulation: Mind the Scalability/Heterogeneity

- Ideal emulator: "Perfect" emulation
- Generic system emulator: Original FW, original kernel
- Userland emulator: Original FW with chroot, generic Kernel
- No emulator: Original FW with architectural chroot

Emulation accuracy

Complexity

Speed
Embedded Devices Emulation: Mind the Scalability/Heterogeneity

- Ideal emulator
  - "Perfect" emulation
- Generic system emulator
  - Original FW, original kernel
- Userland emulator
  - Original FW with chroot, generic Kernel
- No emulator
  - Original FW with architectural chroot
  - Hosted web application

<table>
<thead>
<tr>
<th>Emulation accuracy</th>
<th>Complexity</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Embedded Devices Emulation: Mind the Scalability/Heterogeneity

### Emulation Accuracy
- Ideal emulator
- Generic system emulator
- Userland emulator
- No emulator

### Features
- "Perfect" emulation
- Original FW, original kernel
- Original FW with chroot, generic Kernel
- Original FW with architectural chroot
- Hosted web application

### Metrics
- Complexity
- Speed
Embedded Devices Emulation: Mind the Scalability/Heterogeneity

- Ideal emulator
  - "Perfect" emulation
- Generic system emulator
  - Original FW, original kernel
  - Original FW with chroot, generic Kernel
- Userland emulator
  - Original FW with architectural chroot
- No emulator
  - Hosted web application

<table>
<thead>
<tr>
<th>Emulation accuracy</th>
<th>Complexity</th>
<th>Speed</th>
</tr>
</thead>
</table>
Embedded Devices Emulation: Mind the Scalability/Heterogeneity

- Ideal emulator
  - "Perfect" emulation

- Generic system emulator
  - Original FW, original kernel
  - Original FW with chroot, generic Kernel

- Userland emulator
  - Original FW with architectural chroot

- No emulator
  - Hosted web application

Emulation accuracy
Complexity
Speed
Embedded Devices Emulation: Some modes are challenging

- Ideal emulator: Perfect emulation, original FW, original kernel
- Generic system emulator: Original FW, original kernel
- Userland emulator: Original FW with chroot, generic Kernel
- No emulator: Original FW with architectural chroot, Hosted web application

Emulation accuracy

Complexity

Speed
Embedded Devices Emulation: Some modes are challenging

<table>
<thead>
<tr>
<th>Ideal emulator</th>
<th>Generic system emulator</th>
<th>Userland emulator</th>
<th>No emulator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perfect emulation</td>
<td>Original FW, original kernel</td>
<td>Original FW with chroot, generic Kernel</td>
<td>Original FW with architectural chroot</td>
</tr>
</tbody>
</table>

- Emulation accuracy
- Complexity
- Speed
Embedded Devices Emulation: Some modes are challenging
Embedded Devices Emulation: Some modes are challenging.
QEMU System Emulation:
Original FW, Generic kernel, Chroot

Ubuntu 14 VM

QEMU (Debian Squeeze armel)

Debian Squeeze armel Linux 2.6 Kernel

Linux X86_64 Kernel
QEMU System Emulation: Original FW, Generic kernel, Chroot

- Ubuntu 14 VM
  - QEMU (Debian Squeeze armel)
    - Debian Squeeze UserSpace
      - Firmware (userspace)
    - Debian Squeeze armel Linux 2.6 Kernel
  - Linux X86_64 Kernel
QEMU System Emulation:
Original FW, Generic kernel, Chroot
QEMU System Emulation: Original FW, Generic kernel, Chroot
QEMU System Emulation: Original FW, Generic kernel, Chroot

Ubuntu 14 VM

QEMU (Debian Squeeze armel)

Chrooted Firmware (userspace)

rc scripts

Other services

Debian Squeeze Userspace

Init

Web Server

Perl

HTML

PHP

File System

Shell

Utilities

Native CGI

Debian Squeeze armel Linux 2.6 Kernel

Linux X86_64 Kernel

Scripting Framework

Arachni

ZAP

w3af

Tcpdump

...
<table>
<thead>
<tr>
<th>Dataset phase</th>
<th># of FWs (unique)</th>
<th># of root FS</th>
<th># of vendors (unique)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original dataset</td>
<td>1925</td>
<td>–</td>
<td>54</td>
</tr>
<tr>
<td>Candidates for chroot and web interface emulation</td>
<td>1580</td>
<td>1754</td>
<td>49</td>
</tr>
<tr>
<td>Improved by heuristics</td>
<td>1580</td>
<td>1982</td>
<td>49</td>
</tr>
<tr>
<td>Chroot OK</td>
<td>488</td>
<td>–</td>
<td>17</td>
</tr>
<tr>
<td>Web server OK</td>
<td>246</td>
<td>–</td>
<td>11</td>
</tr>
<tr>
<td>High impact vulnerabilities (static + dynamic)</td>
<td>185</td>
<td>–</td>
<td>13</td>
</tr>
</tbody>
</table>
- Emulation failures limit the FW test coverage
  - “chroot failed” failures for 69% (or 1092) FWs
  - “webserver failed” failures for 50% (or 242) FWs
  - Failure analysis, random sampling
    - 95% confidence level and a ± 10% confidence interval for the accuracy of estimations
  - Fixing “chroot failed” should be relatively easy for 70.4% of the failures
  - Fixing “webserver failed” – should be relatively easy for 34.8% of the failures
<table>
<thead>
<tr>
<th>Arch.</th>
<th>QEMU support</th>
<th>Original firmware</th>
<th>Chroot OK</th>
<th>Web server OK</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM</td>
<td>mainline</td>
<td>35%</td>
<td>53%</td>
<td>55%</td>
</tr>
<tr>
<td>MIPS</td>
<td>mainline</td>
<td>19%</td>
<td>21%</td>
<td>17%</td>
</tr>
<tr>
<td>MIPSel</td>
<td>mainline</td>
<td>17%</td>
<td>26%</td>
<td>28%</td>
</tr>
<tr>
<td>Axis CRIS</td>
<td>patch [53, 54]</td>
<td>16%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>bFLT</td>
<td>mainline</td>
<td>5%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>PowerPC</td>
<td>mainline</td>
<td>3%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Intel 80386</td>
<td>mainline</td>
<td>2%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>DLink Specific</td>
<td>no</td>
<td>≲ 1%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Unknown</td>
<td>no</td>
<td>≲ 1%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Altera Nios II</td>
<td>patch [83]</td>
<td>≲ 1%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>ARC Tagent-A5</td>
<td>no</td>
<td>≲ 1%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>–</td>
<td><strong>1925</strong></td>
<td><strong>488</strong></td>
<td><strong>246</strong></td>
</tr>
<tr>
<td>Web server</td>
<td>% among started web servers</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>minihttpd</td>
<td>37%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lighttpd</td>
<td>30%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boa</td>
<td>4%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>thttpd</td>
<td>3%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>empty banner</td>
<td>26%</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Network services – Fuzz 'em all!

TABLE VIII: Distribution of network services opened by 207 firmware instances out of 488 successfully emulated ones. The last entry summarizes the 16 unusual port numbers opened by services such as web, telnetd, ftp or upnp servers.

<table>
<thead>
<tr>
<th>Port type</th>
<th>Port number</th>
<th>Service name</th>
<th># of FWs</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>554</td>
<td>RTSP</td>
<td>91</td>
</tr>
<tr>
<td>TCP</td>
<td>555</td>
<td>RTSP</td>
<td>84</td>
</tr>
<tr>
<td>TCP</td>
<td>23</td>
<td>Telnet</td>
<td>60</td>
</tr>
<tr>
<td>TCP</td>
<td>53</td>
<td>DNS</td>
<td>23</td>
</tr>
<tr>
<td>TCP</td>
<td>22</td>
<td>SSH</td>
<td>15</td>
</tr>
<tr>
<td>TCP</td>
<td>Others</td>
<td>Others</td>
<td>58</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td>207 (unique)</td>
</tr>
</tbody>
</table>
## Results: Static Analysis

<table>
<thead>
<tr>
<th>Vulnerability type</th>
<th># of issues</th>
<th># of affected FWs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross-site scripting</td>
<td>5000</td>
<td>143</td>
</tr>
<tr>
<td>File manipulation</td>
<td>1129</td>
<td>98</td>
</tr>
<tr>
<td>Command execution</td>
<td>938</td>
<td>41</td>
</tr>
<tr>
<td>File inclusion</td>
<td>513</td>
<td>40</td>
</tr>
<tr>
<td>File disclosure</td>
<td>461</td>
<td>87</td>
</tr>
<tr>
<td>SQL injection</td>
<td>442</td>
<td>10</td>
</tr>
<tr>
<td>Possible flow control</td>
<td>171</td>
<td>56</td>
</tr>
<tr>
<td>Code execution</td>
<td>141</td>
<td>21</td>
</tr>
<tr>
<td>HTTP response splitting</td>
<td>127</td>
<td>27</td>
</tr>
<tr>
<td>Unserialize</td>
<td>119</td>
<td>15</td>
</tr>
<tr>
<td>POP gadgets</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>HTTP header injection</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>9046</strong></td>
<td><strong>145 (unique)</strong></td>
</tr>
</tbody>
</table>
## Dynamic Analysis

<table>
<thead>
<tr>
<th>Vulnerability type</th>
<th># of issues</th>
<th># of affected FWs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command execution</td>
<td>51</td>
<td>21</td>
</tr>
<tr>
<td>Cross-site scripting</td>
<td>90</td>
<td>32</td>
</tr>
<tr>
<td>CSRF</td>
<td>84</td>
<td>37</td>
</tr>
<tr>
<td><strong>Sub-total HIGH impact</strong></td>
<td><strong>225</strong></td>
<td><strong>45 (unique)</strong></td>
</tr>
<tr>
<td>Cookies w/o HttpOnly †</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>No X-Content-Type-Options †</td>
<td>2938</td>
<td>23</td>
</tr>
<tr>
<td>No X-Frame-Options †</td>
<td>2893</td>
<td>23</td>
</tr>
<tr>
<td>Backup files †</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Application error info †</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>Sub-total low impact †</strong></td>
<td><strong>5843</strong></td>
<td><strong>23 (unique)</strong></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>6068</strong></td>
<td><strong>58 (unique)</strong></td>
</tr>
</tbody>
</table>
• CVE-2011-1674
  • (Pre-Auth) Web Privilege Escalation to **admin**
    • *The NetGear ProSafe WNAP210 with firmware 2.0.12 allows remote attackers to **bypass authentication** and obtain access to the configuration page by visiting recreate.php and then visiting index.php.*

• Affected Devices
  • NetGear WNAP210
  • Just WNAP210, really?

• Using our scalable dynamic analysis framework
  • Quickly verify other firmwares for existing CVEs
  • NetGear WG103
ACSA-2015-001
  - http://firmware.re/vulns/cve-2016-1555.php

(Pre-Auth) Command Injection and XSS

Affected Devices – NetGear
  - WG102, WG103
  - WN604
  - WNDAP350, WNDAP360
  - WNAP320
  - WNAP210
  - WNDAP620, WNDAP660
  - WNDAP380R, WNDAP380R(v2)
  - WN370
  - WND930
• Affected Modules (name)
  • boardData102.php (example below)
  • boardData103.php
  • boardDataNA.php
  • boardDataWW.php
  • boardDataJP.php
• Command Injection
  • http://NETGEAR-DEVICE-IP/boardData102.php?
    writeData=true&reginfo=0&macAddress=%20001122334455%20-c
    %200%20;cp%20/etc/passwd%20/tmp/passwd;%20echo%20#?
  • Independently discovered by Chen et. al as CVE-2016-1555
• XSS
  • http://NETGEAR-DEVICE-IP/boardData102.php?macAddress=
    %22%3E%3Cscript%3Ealert%281%29%3C/script%3E
• Affected Modules (sha256)
  • 03bd170b6b284f43168dcf9de905ed33ae2edd721554cebecc81894a8d5bcdeea5
  • 2311b6a83298833d2cf6f6d02f38b04c8f562f3a1b5eb0092476efd025fd4004
  • 325c7fe9555a62c6ed49358c27881b1f32c26a93f8b9b91214e8d70d595d89bb
  • 33a29622653ef3abc1f178d3f3670f55151137941275f187a7c03ec2acdb5caa
  • 35c60f56ffcc79f00bf1322830ecf65c9a8ca8e0f1d68692ee1b5b9df1bdef7c1
  • 40fbb495a60c5ae68d83d3ae69197ac03ac50a8201d2bccd23f296361b0040b9
  • 453658ac170bda80a6539dcb6d42451f30644c7b089308352a0b3422d21bdc01
  • 4679aca17917ab9b074d38217bb5302e33a725ad179f2e4aaf2e7233ec6bc842
  • 56714f750d6b8e2cf8c9c3a8f310ac226b5b0c6b2ab3f93175826a42ea0f4545
  • 70fe0274d6616126e758473b043da37c2635a871e295395e073fb782f955840e
  • 760bde74861b6e48dcbf3e5513aaa721583fbd2e69c93bccb246800e8b9bc1e6
  • 8bf836c5826a1017b339e23411162ef6f6acc343c3df02a8ee9e6df40abe681ff
  • 9f56e5656c137a5ce407eee25bf2405f56b56e69fa89c61cfd65f07bc6600ef
  • a5ef01368da8588fc4bc72d3faaa20b21c43c0eea6ef71866b7aa160e531a5b4
  • dcefcff36f2825333784c86212e0f1b73b25db9db78476d9c75035f51f135ef6
ACSA-2015-002
  - [Link](http://firmware.re/vulns/acsa-2015-002.php)
  - (Pre-Auth) Command Injection
  - Affected Devices – Netgear ProSafe
    - WC9500 (~5,500 USD)
    - WC7600 (~3,400 USD)
    - WC7520 (~1,200 USD)
    - WMS5316 (~1,000 USD) (*maybe vulnerable)
  - Affected Modules (name)
    - login_handler.php
    - Related: ExploitDB 38097 “login_handler.php” for NetGear WMS5316
  - Command Injection
• High-severity **vulnerability impact**
  • Command injection, XSS, CSRF
  • **Automated+scalable** static and dynamic analysis
  • **225 high-severity** vulnerabilities, many previously unknown
  • **185 firmware images** (~10% of original)
  • **13 vendors** (~25% of original)
• Total **alerts** from the tools
  • 6068 dynamic analysis alerts on 58 firmware images
  • 9046 static analysis alerts on 145 firmware images
  • Manual triage and confirmation is challenging
"Automated Dynamic Firmware Analysis at Scale: A Case Study on Embedded Web Interfaces" (ACM AsiaCCS 2016 to appear)

- [http://firmware.re/dynamicanalysis/](http://firmware.re/dynamicanalysis/)

"A Large-Scale Analysis of the Security of Embedded Firmwares" (Usenix Security 2014)

- [http://firmware.re/usenixsec14/](http://firmware.re/usenixsec14/)

• http://binwalk.org/
• http://www.binaryanalysis.org/
• http://rips-scanner.sourceforge.net/
• http://www.arachni-scanner.com/
• https://www.owasp.org/index.php/OWASP_Zed
• http://w3af.org/
• http://www.metasploit.com/
• http://www.tenable.com/products/nessus-vulnerability-scanner
• https://shodan.io
• https://zmap.io
• https://scans.io
• https://censys.io
• https://www.zoomeye.org/
Large scale firmware analysis is absolutely necessary, especially with the IoT hype.
- Large scale firmware analysis is absolutely necessary, especially with the IoT hype

- Scalable (dynamic) analysis of firmware is feasible and yields very good results
• Large scale firmware analysis is absolutely necessary, especially with the IoT hype

• Scalable (dynamic) analysis of firmware is feasible and yields very good results

• Many vendors do not perform proper/basic security testing and QA
• Dr. Jonas Zaddach
• Prof. Aurelien Francillon
• Prof. Davide Balzarotti
• Dr. Apostolos Zarras
• S3 SysSec research group
Your feedback is important!

Please fill the BH16ASIA feedback form for this talk

“AUTOMATED DYNAMIC FIRMWARE ANALYSIS AT SCALE: A CASE STUDY ON EMBEDDED WEB INTERFACES”

Thank you!
Thank you!
Questions?

andrei@firmware.re
@costinandre i