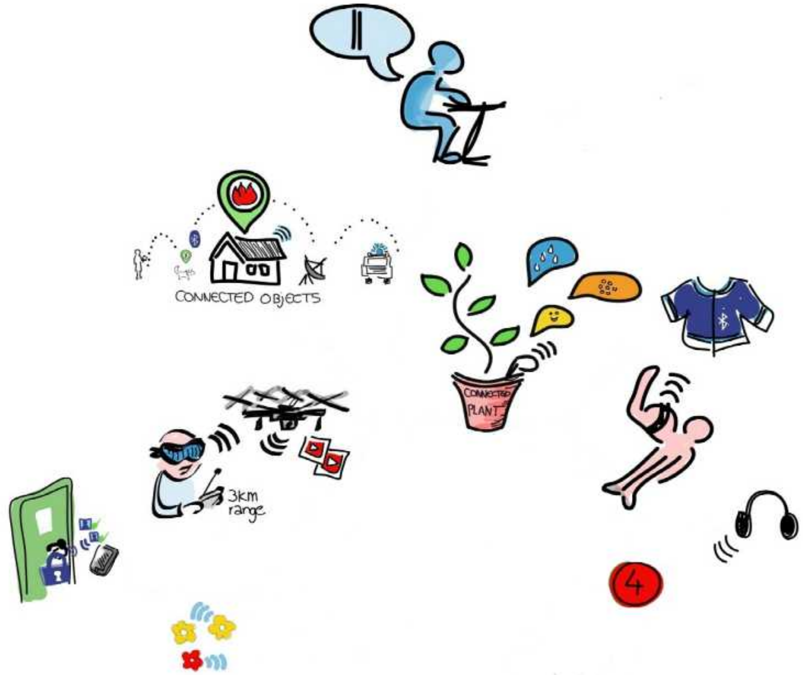
black hat ASIA 2016

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

Andrei Costin
andrei@firmware.re
@costinandrei

UBM

Embedded Devices: EVERYWHERE!

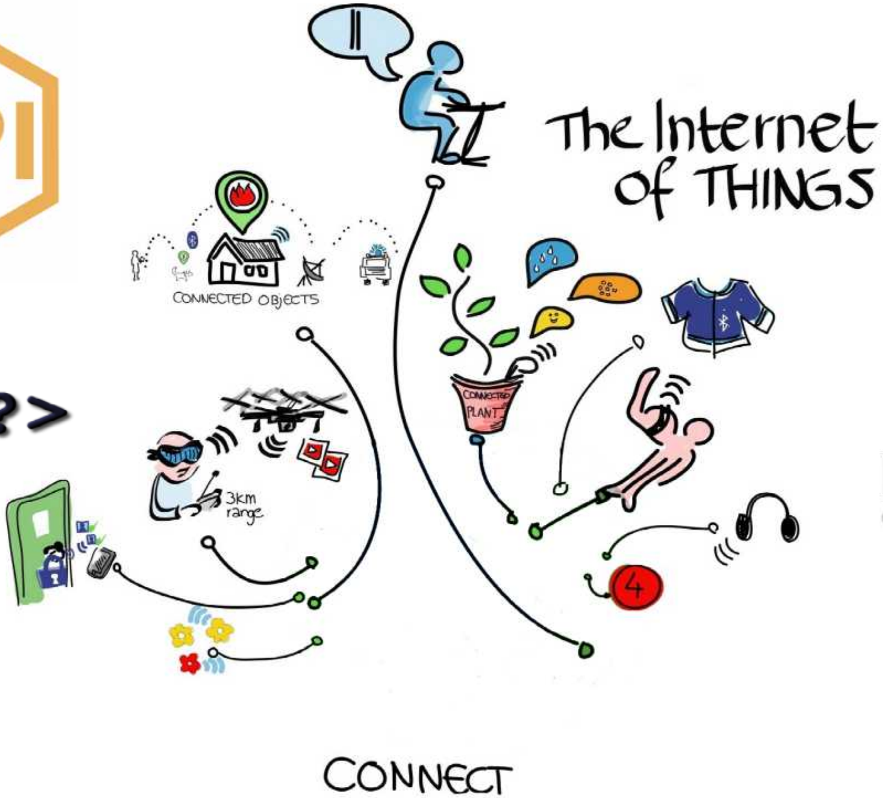


by Wilgengebroed on Flickr [CC-BY-2.0]

Embedded Devices: Smarted, More Complex

API

<?xml?>



{JSON}

RESTful API

GET PUT POST DELETE

by Wilgengebroed on Flickr [CC-BY-2.0]

black hat ASIA 2016

Embedded Devices: More Interconnected, More WWW

API

<?xml?>

by Wilgengebroed on Flickr [CC-BY-2.0]

{JSON}

RESTful API

GET PUT POST DELETE

Observations

- 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*)

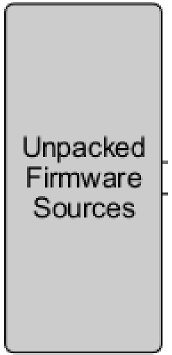
Challenges

- Large number of devices
- Large number of firmware files
- Highly heterogeneous systems
- Increasingly “smart”, “connected”
- Highly unstructured firmware data
- Vulnerable devices exposed

Challenges and Solutions

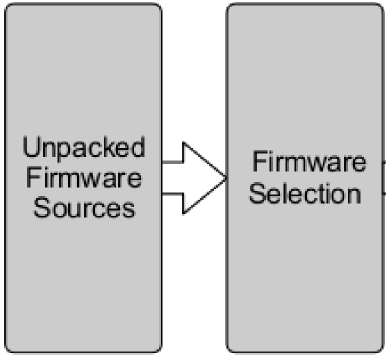
- Large number of devices → Analysis without devices
- Large number of firmware files → Scalable architectures
- Highly heterogeneous systems → Generic techniques
- Increasingly “smart”, “connected” → Focus on web interfaces & APIs
- Highly unstructured firmware data → Large dataset classification
- Vulnerable devices exposed → Technology-independent device fingerprinting

Scalable Framework: Dynamic Firmware Analysis

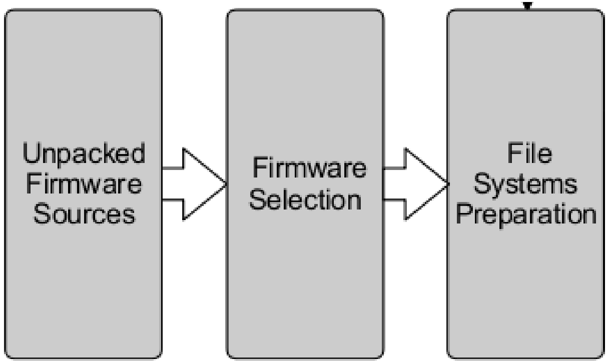


Unpacked Firmware Sources

Scalable Framework: Dynamic Firmware Analysis



Scalable Framework: Dynamic Firmware Analysis

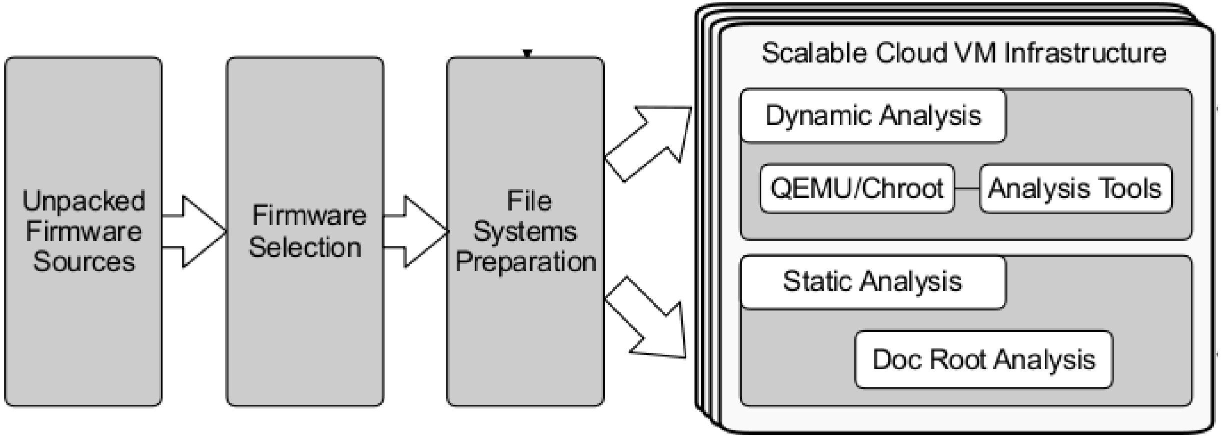


Unpacked Firmware Sources

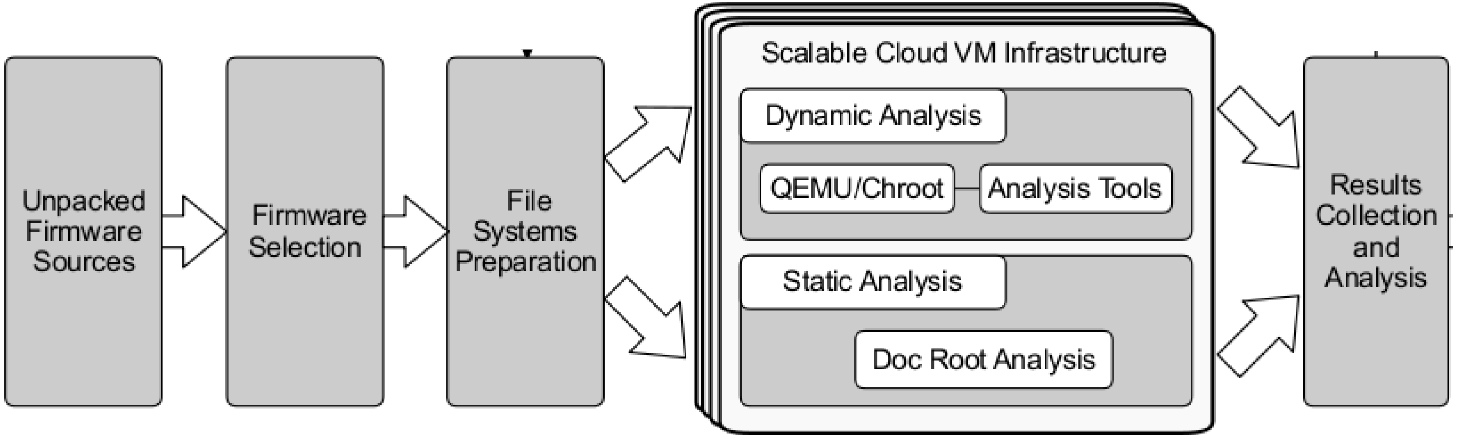
Firmware Selection

File Systems Preparation

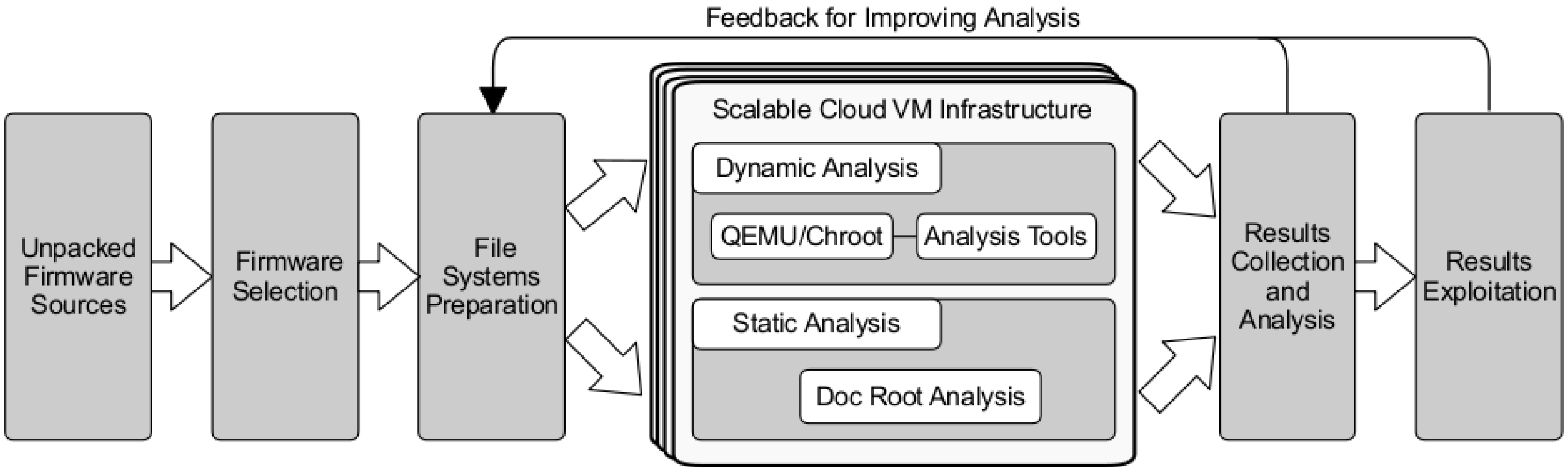
Scalable Framework: Dynamic Firmware Analysis



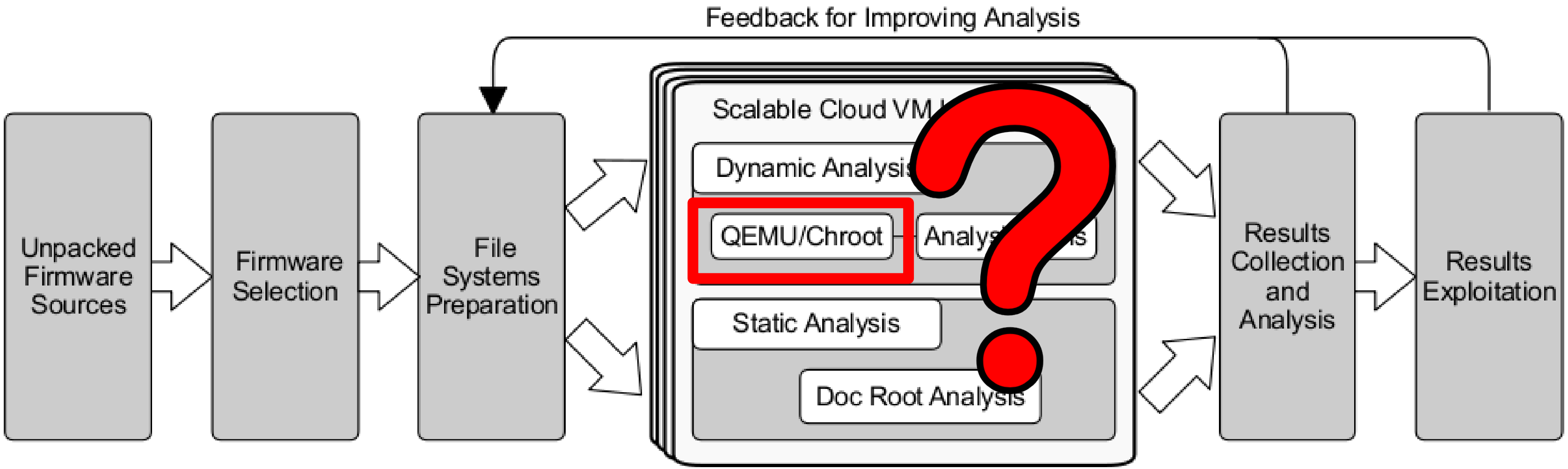
Scalable Framework: Dynamic Firmware Analysis



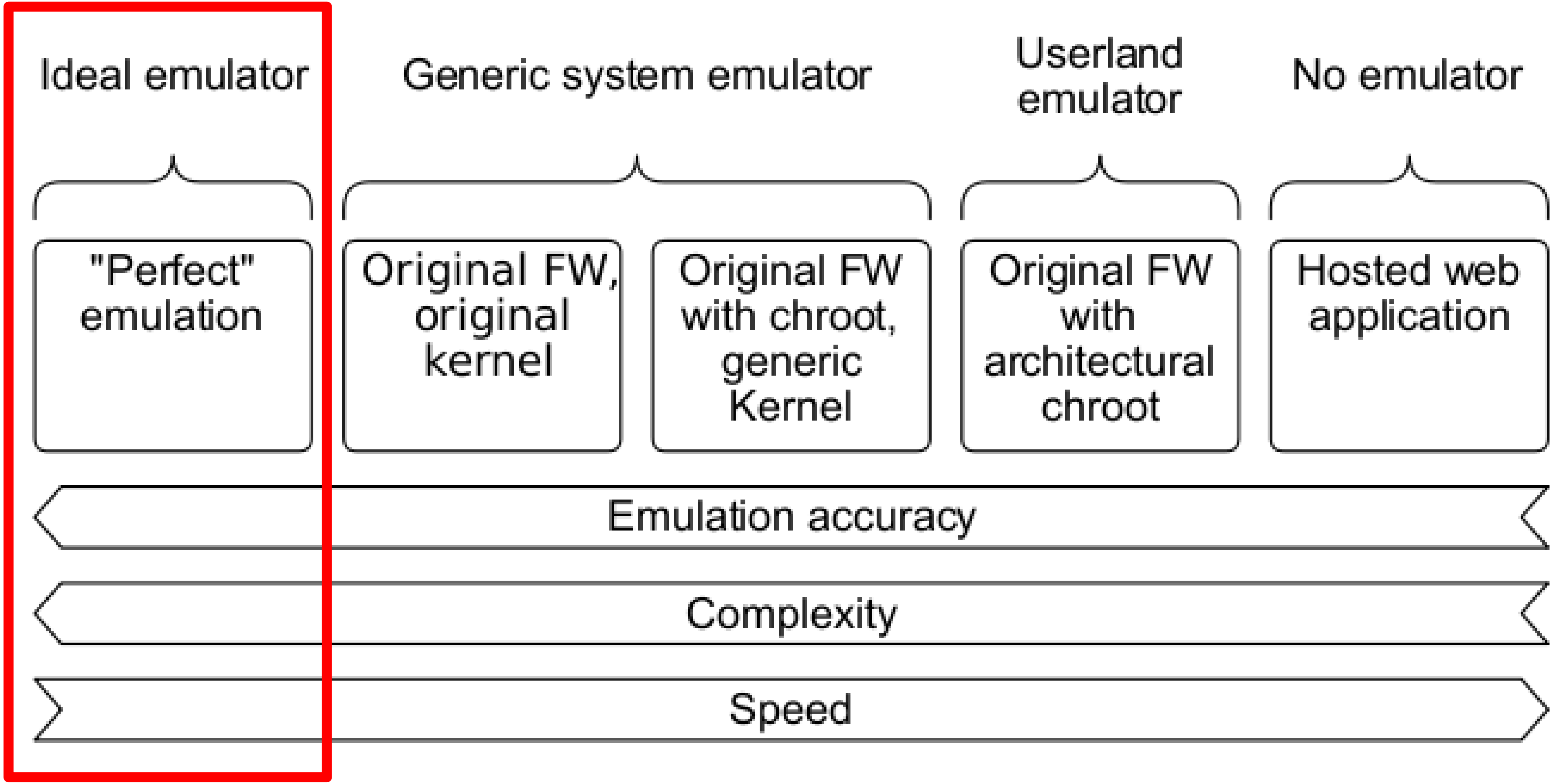
Scalable Framework: Dynamic Firmware Analysis



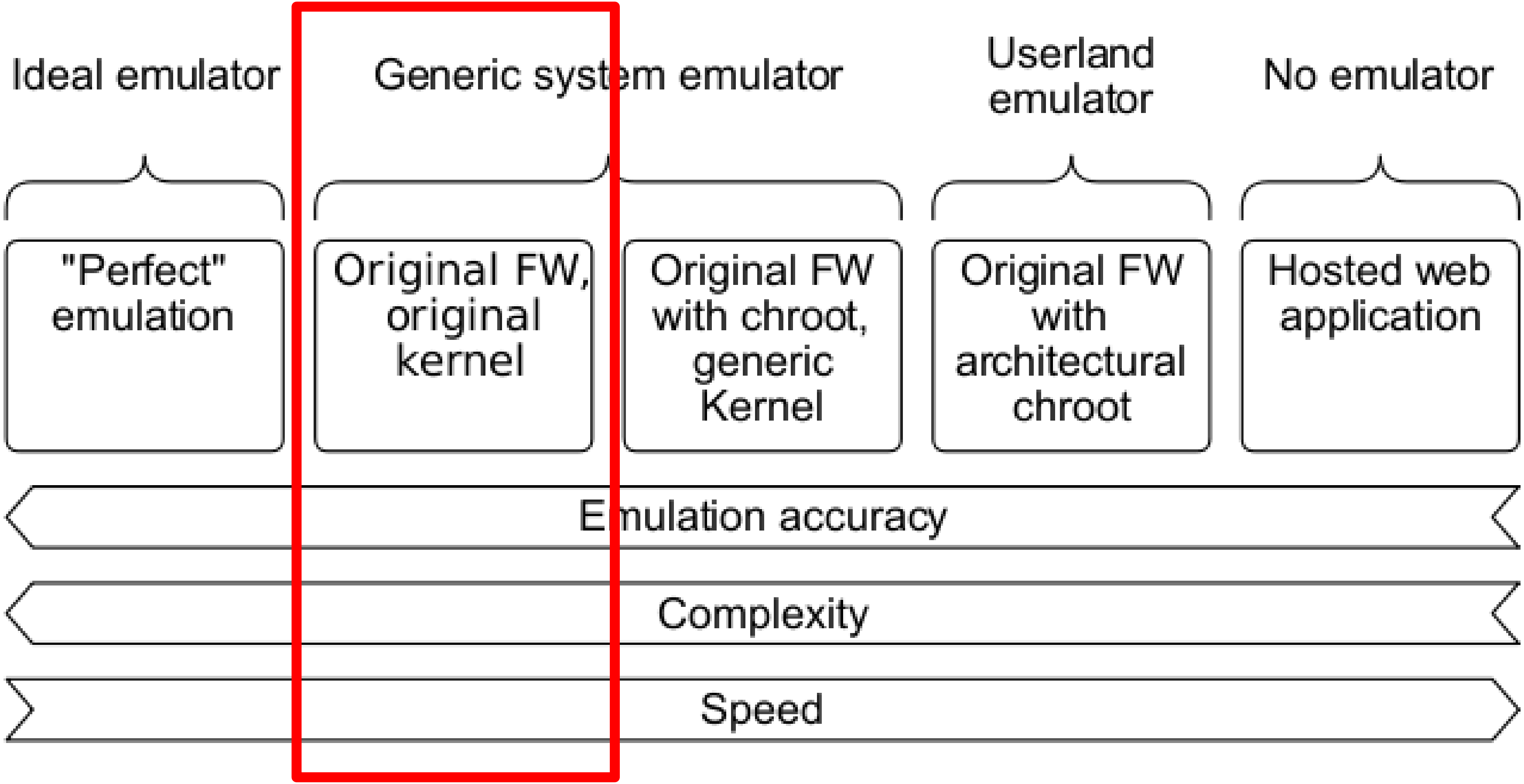
Scalable Framework: Dynamic Firmware Analysis



Embedded Devices Emulation: Mind the Scalability/Heterogeneity



Embedded Devices Emulation: Mind the Scalability/Heterogeneity



Ideal emulator

Generic system emulator

Userland emulator

No emulator

"Perfect" emulation

Original FW, original kernel

Original FW with chroot, generic Kernel

Original FW with architectural chroot

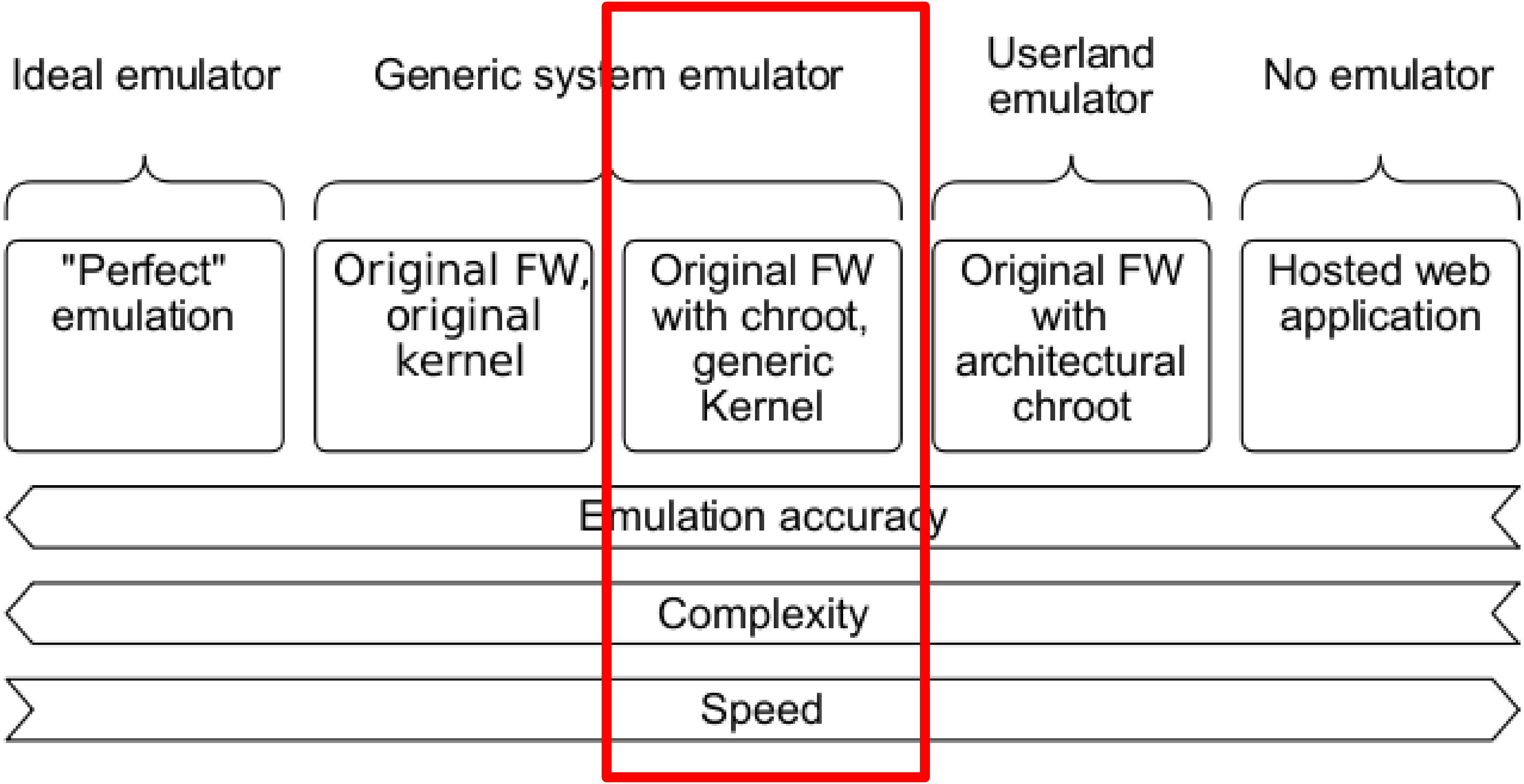
Hosted web application

Emulation accuracy

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

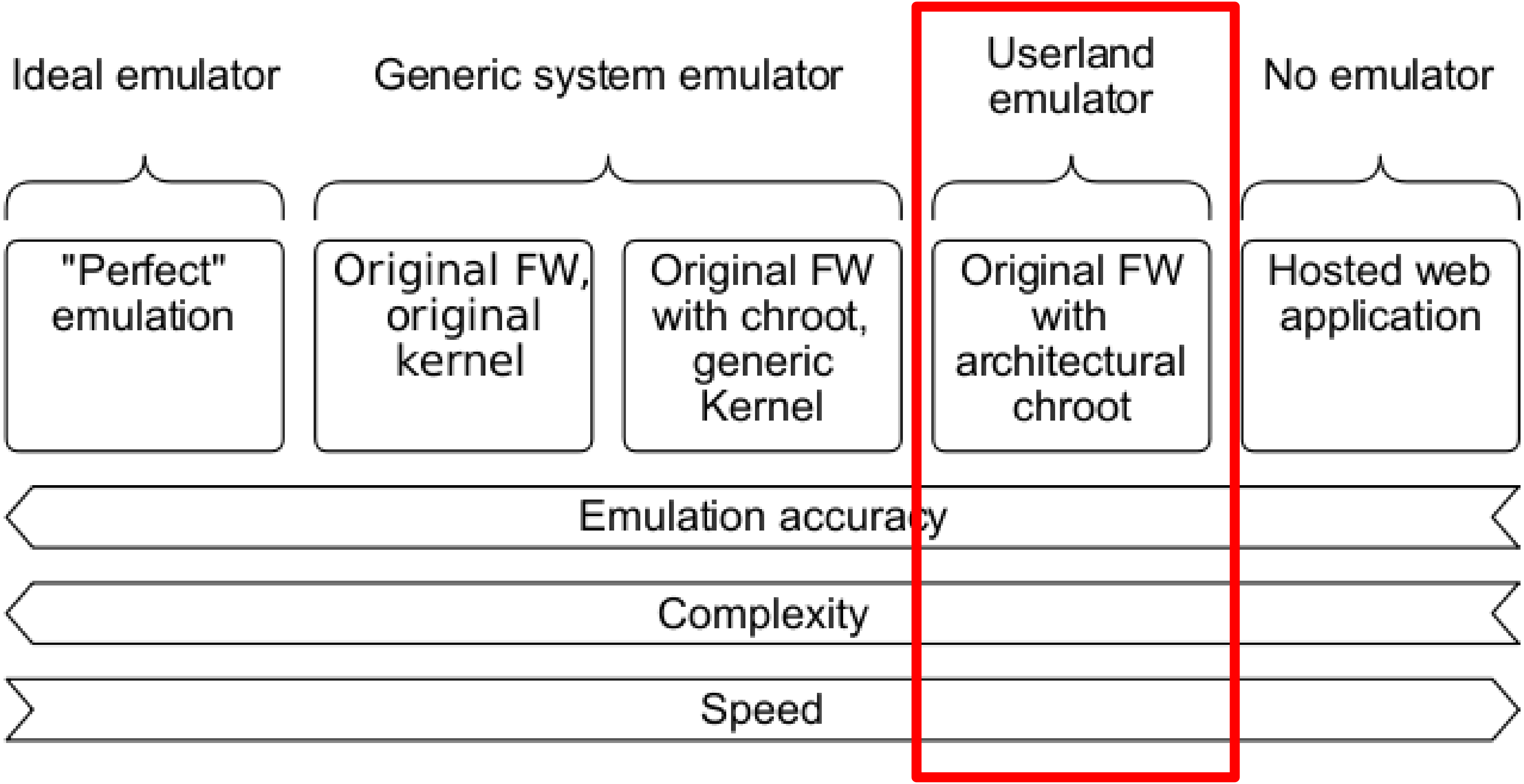
Hosted web application

Emulation accuracy

Complexity

Speed

Embedded Devices Emulation: Mind the Scalability/Heterogeneity



Ideal emulator

Generic system emulator

Userland emulator

No emulator

"Perfect" emulation

Original FW, original kernel

Original FW with chroot, generic Kernel

Original FW with architectural chroot

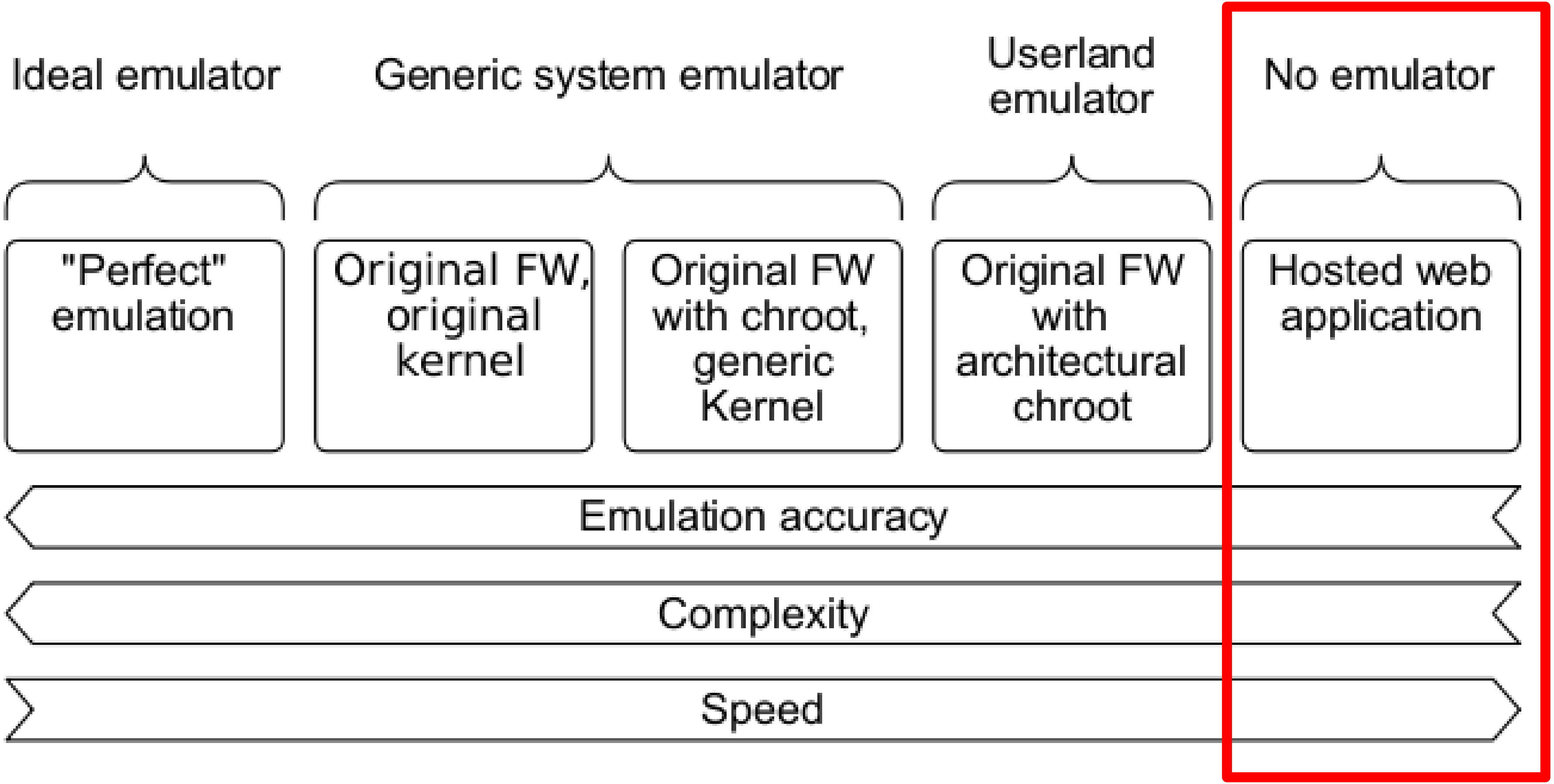
Hosted web application

Emulation accuracy

Complexity

Speed

Embedded Devices Emulation: Mind the Scalability/Heterogeneity



Ideal emulator

Generic system emulator

Userland emulator

No emulator

"Perfect" emulation

Original FW, original kernel

Original FW with chroot, generic Kernel

Original FW with architectural chroot

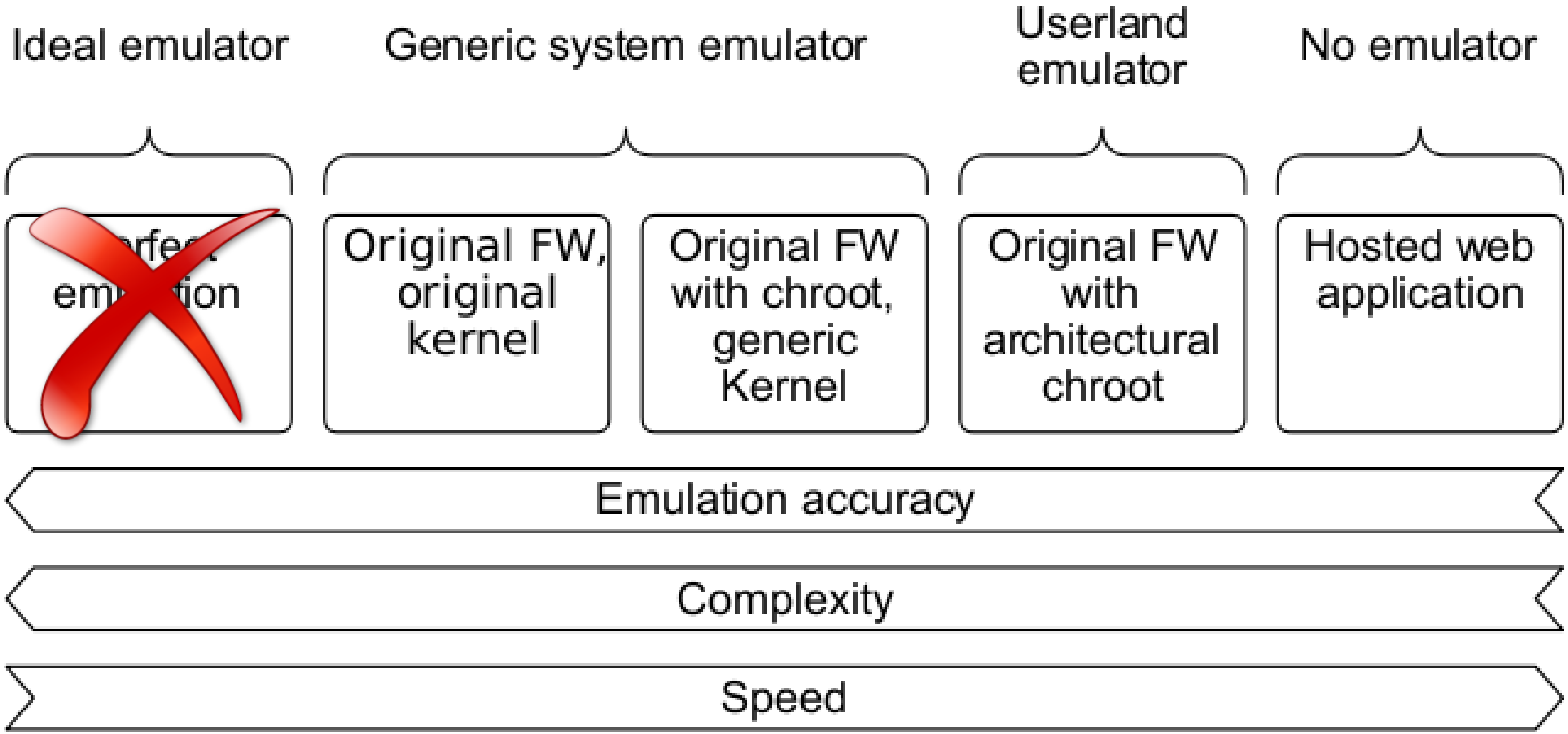
Hosted web application

Emulation accuracy

Complexity

Speed

Embedded Devices Emulation: Some modes are challenging



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

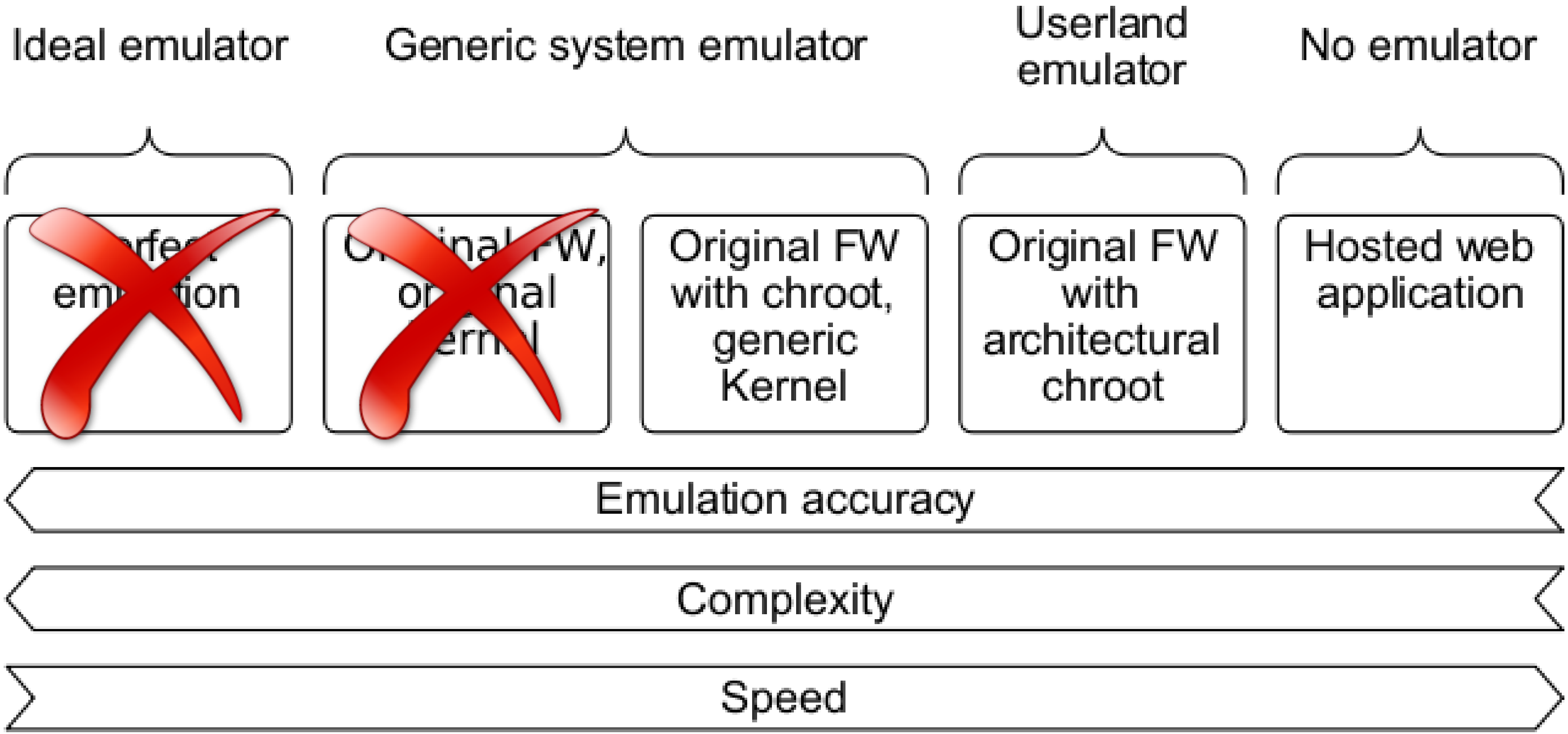
Hosted web application

Emulation accuracy

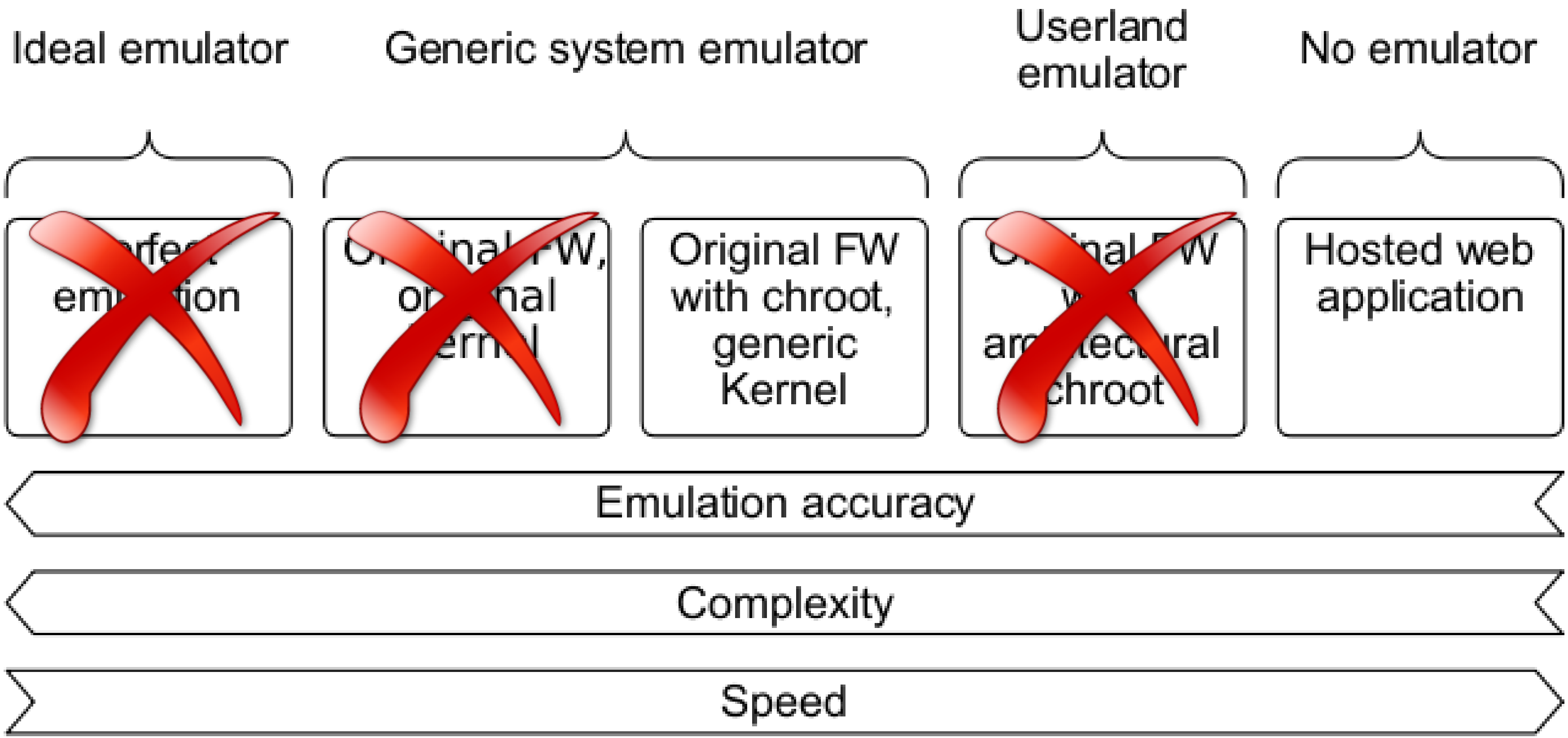
Complexity

Speed

Embedded Devices Emulation: Some modes are challenging



Embedded Devices Emulation: Some modes are challenging



Ideal emulator

Generic system emulator

Userland emulator

No emulator

Perfect emulation

Original FW, original Kernel

Original FW with chroot, generic Kernel

Original FW with architectural chroot

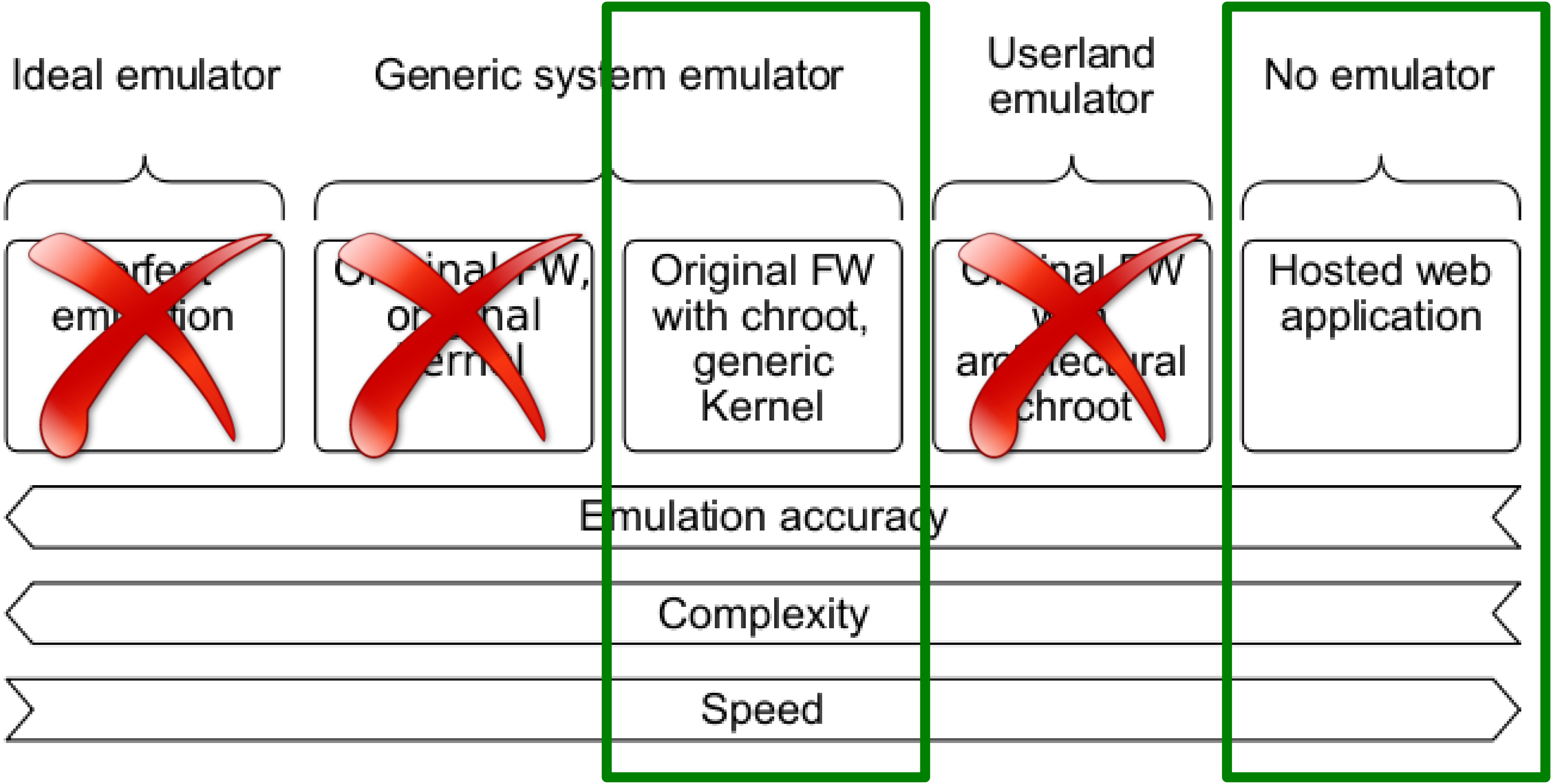
Hosted web application

Emulation accuracy

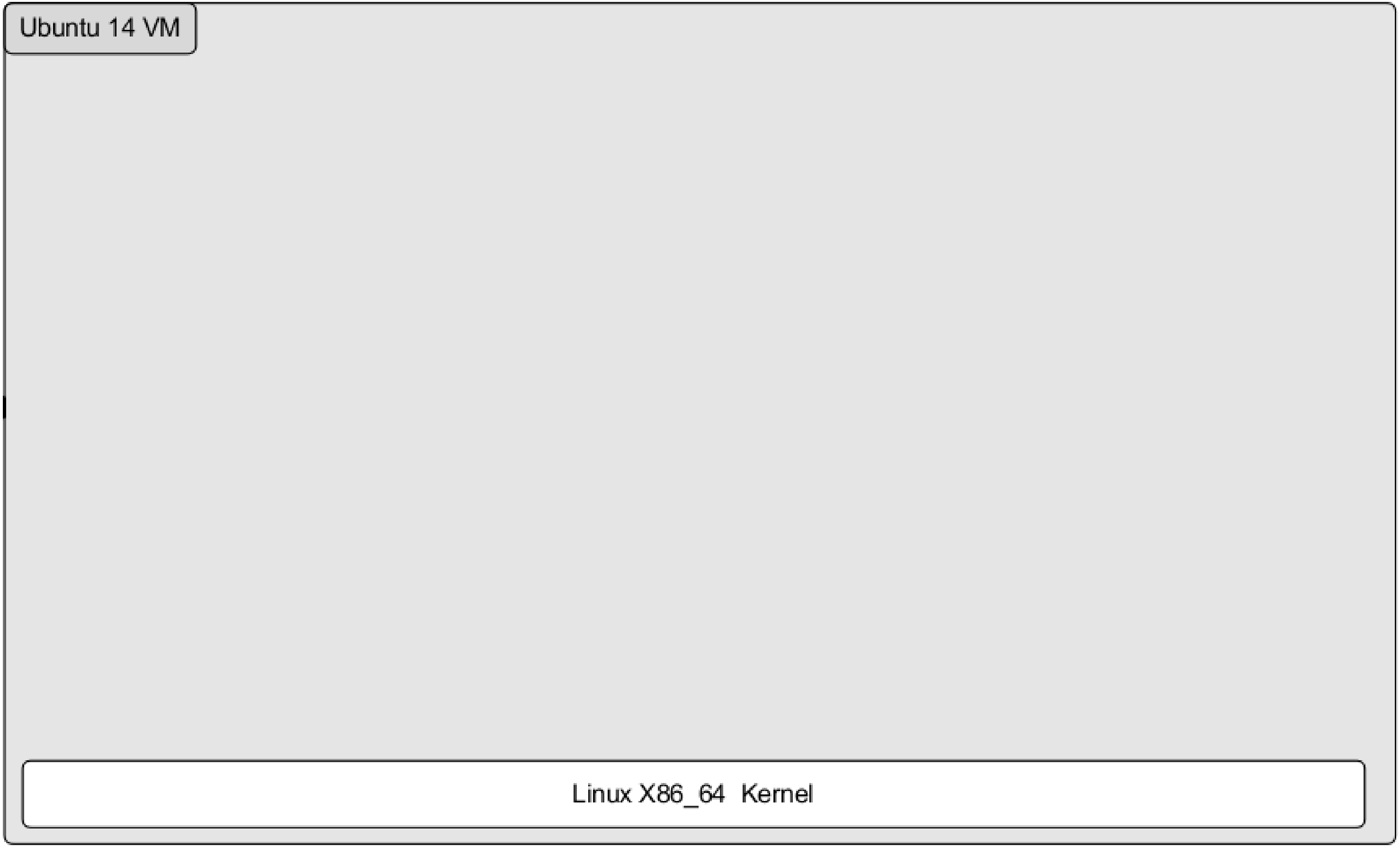
Complexity

Speed

Embedded Devices Emulation: Some modes are challenging



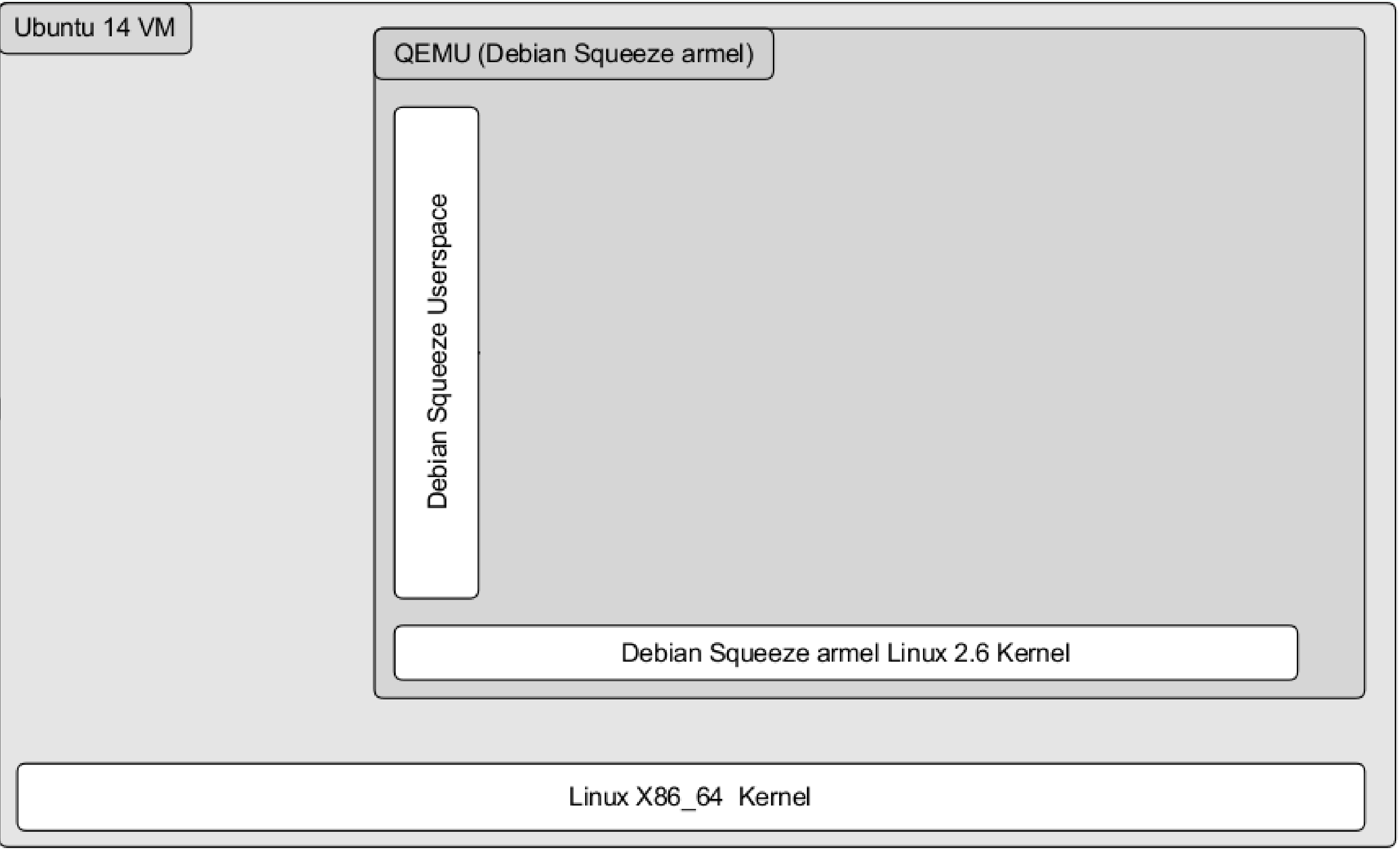
QEMU System Emulation: Original FW, Generic kernel, Chroot



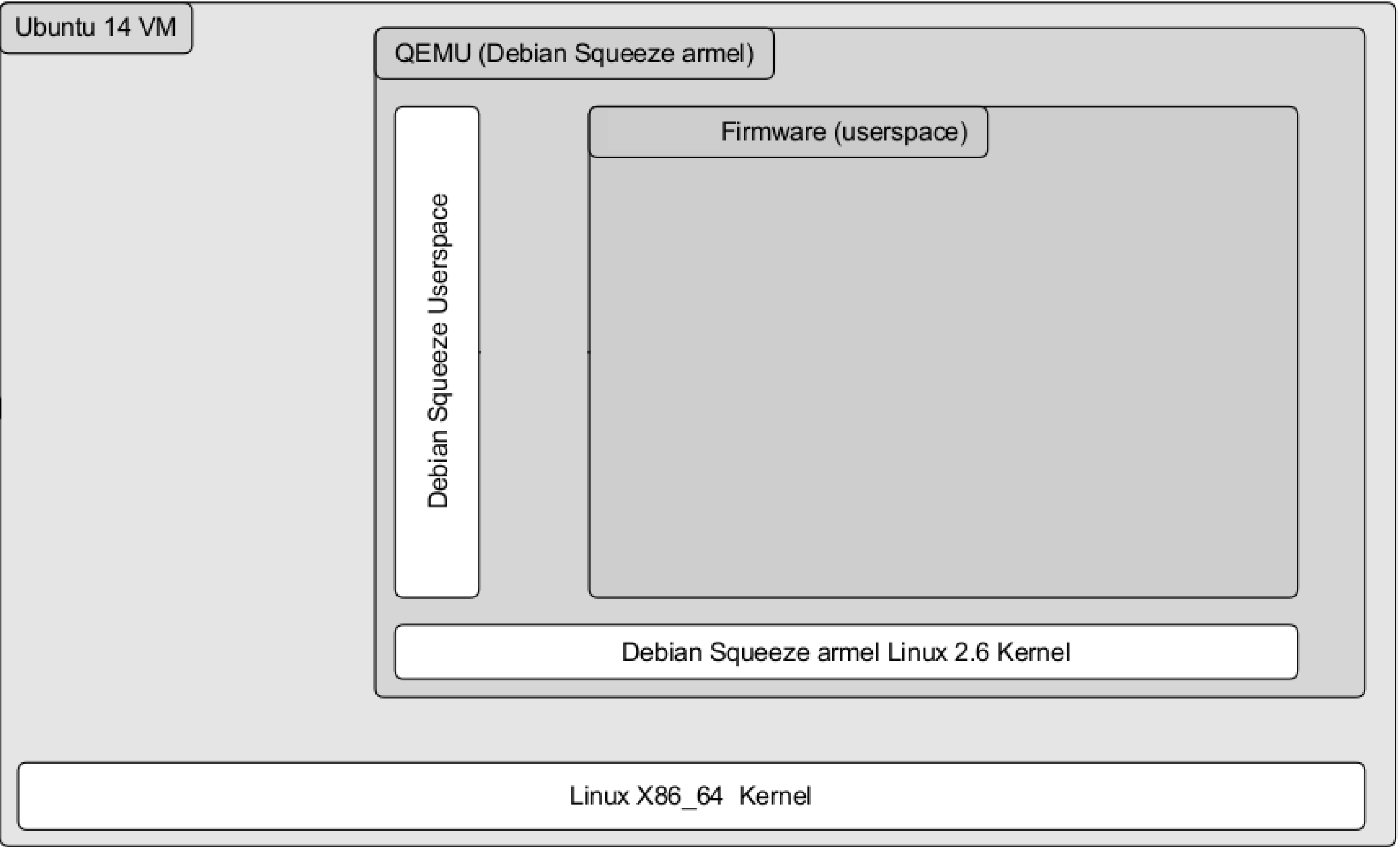
Ubuntu 14 VM

Linux X86_64 Kernel

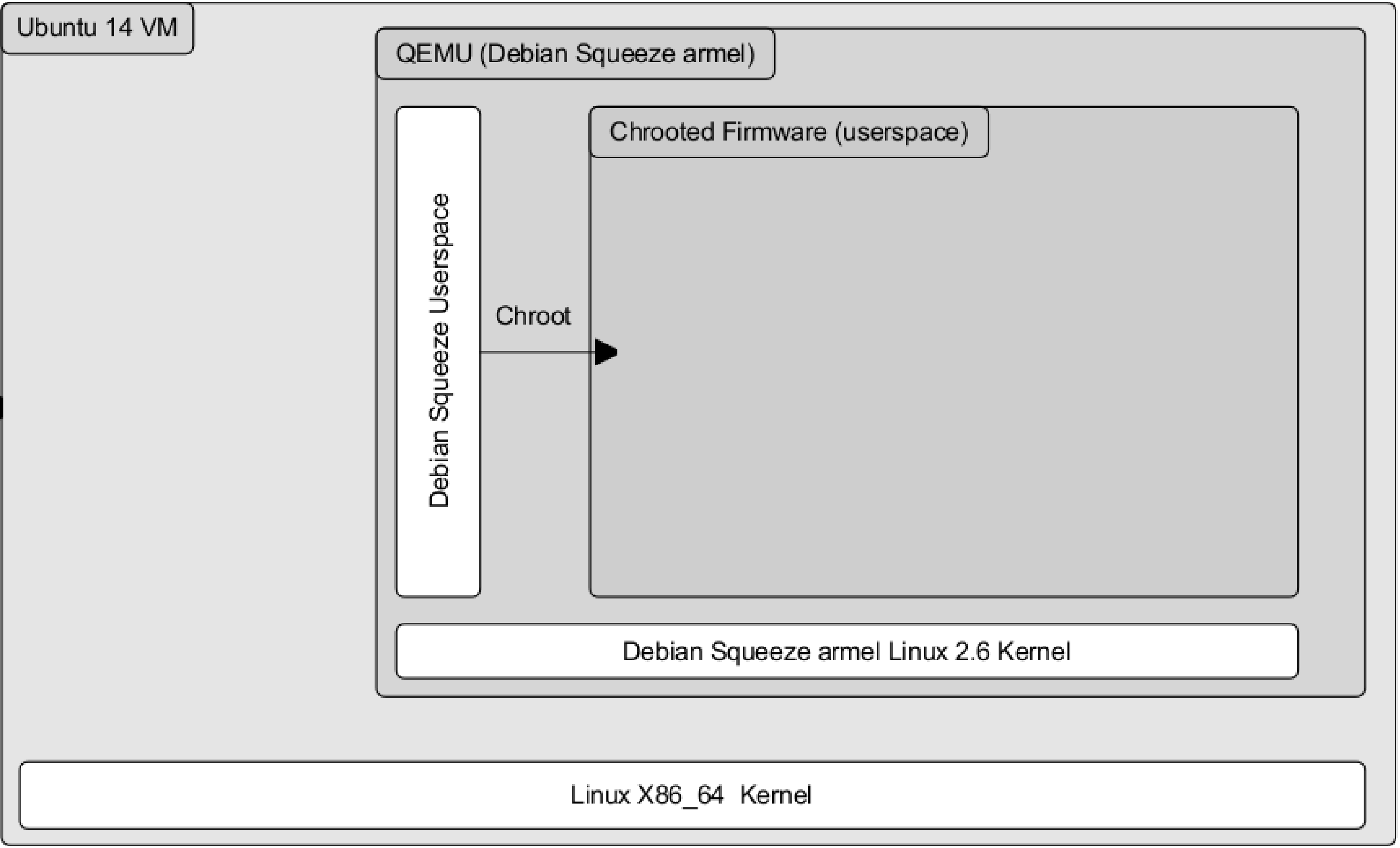
QEMU System Emulation: Original FW, Generic kernel, Chroot


Ubuntu 14 VM
QEMU (Debian Squeeze armel)
Debian Squeeze Userspace
Debian Squeeze armel Linux 2.6 Kernel
Linux X86_64 Kernel

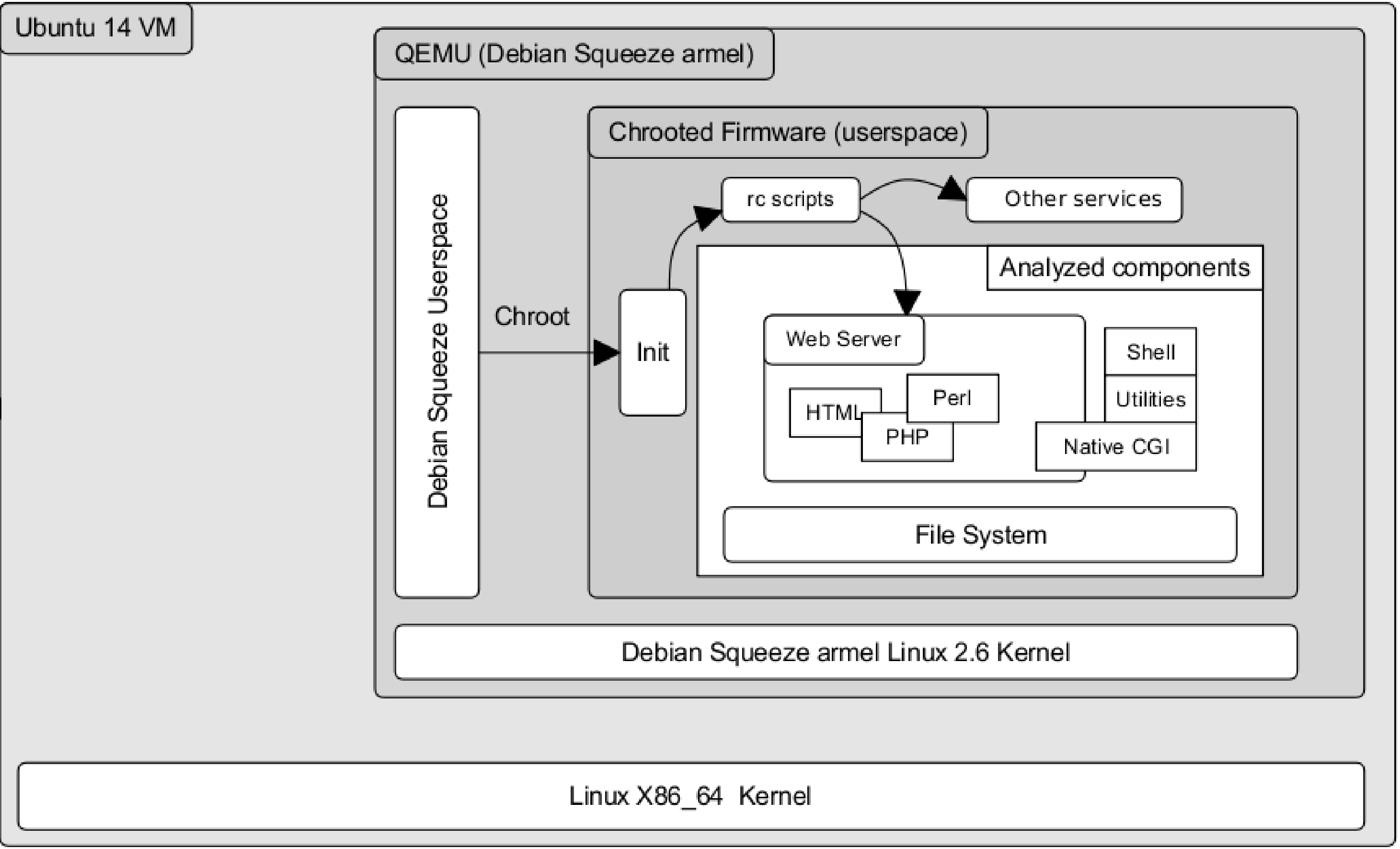
QEMU System Emulation: Original FW, Generic kernel, Chroot



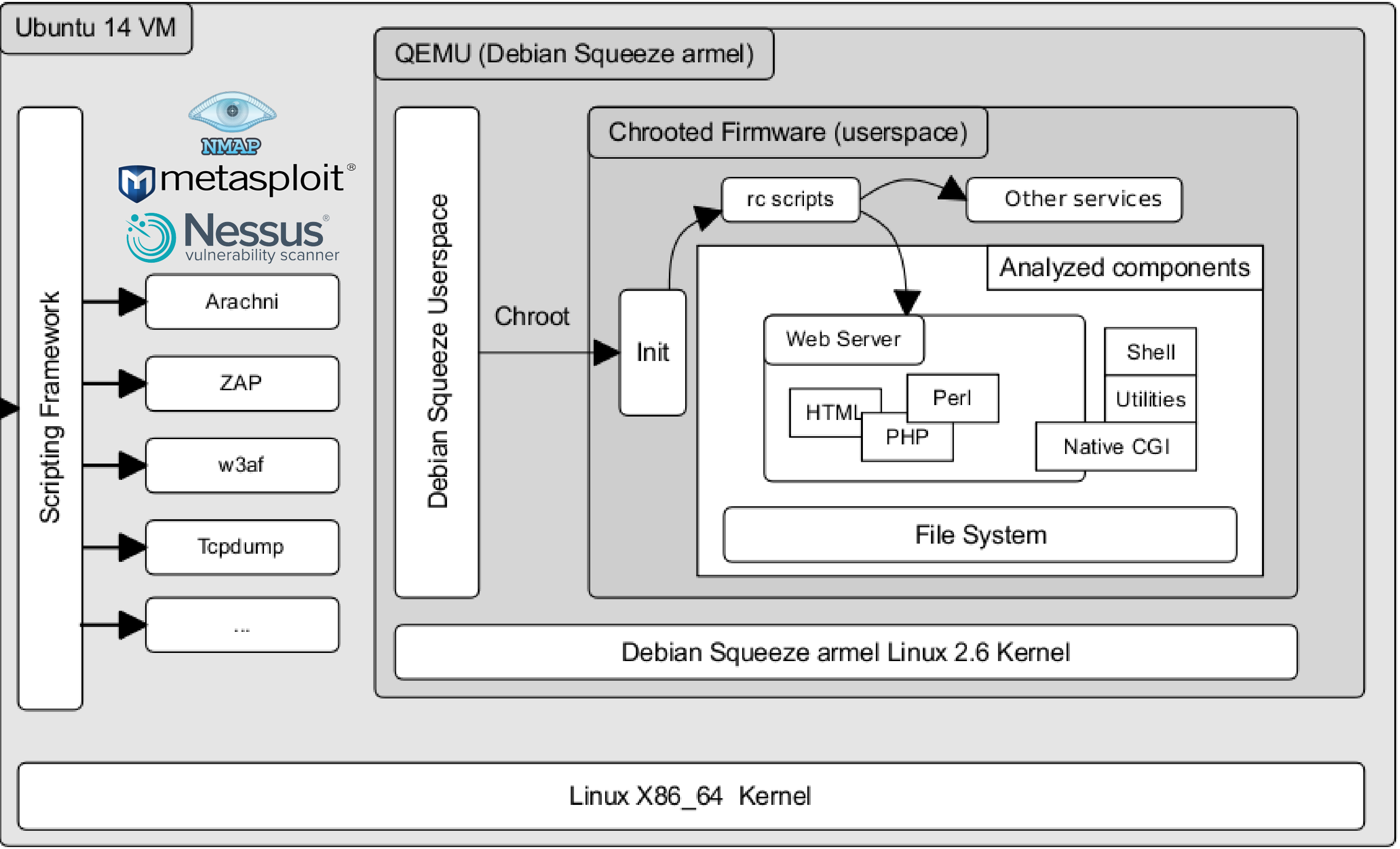
QEMU System Emulation: Original FW, Generic kernel, Chroot


Ubuntu 14 VM
QEMU (Debian Squeeze armel)
Debian Squeeze Userspace
Chroot
Chrooted Firmware (userspace)
Debian Squeeze armel Linux 2.6 Kernel
Linux X86_64 Kernel

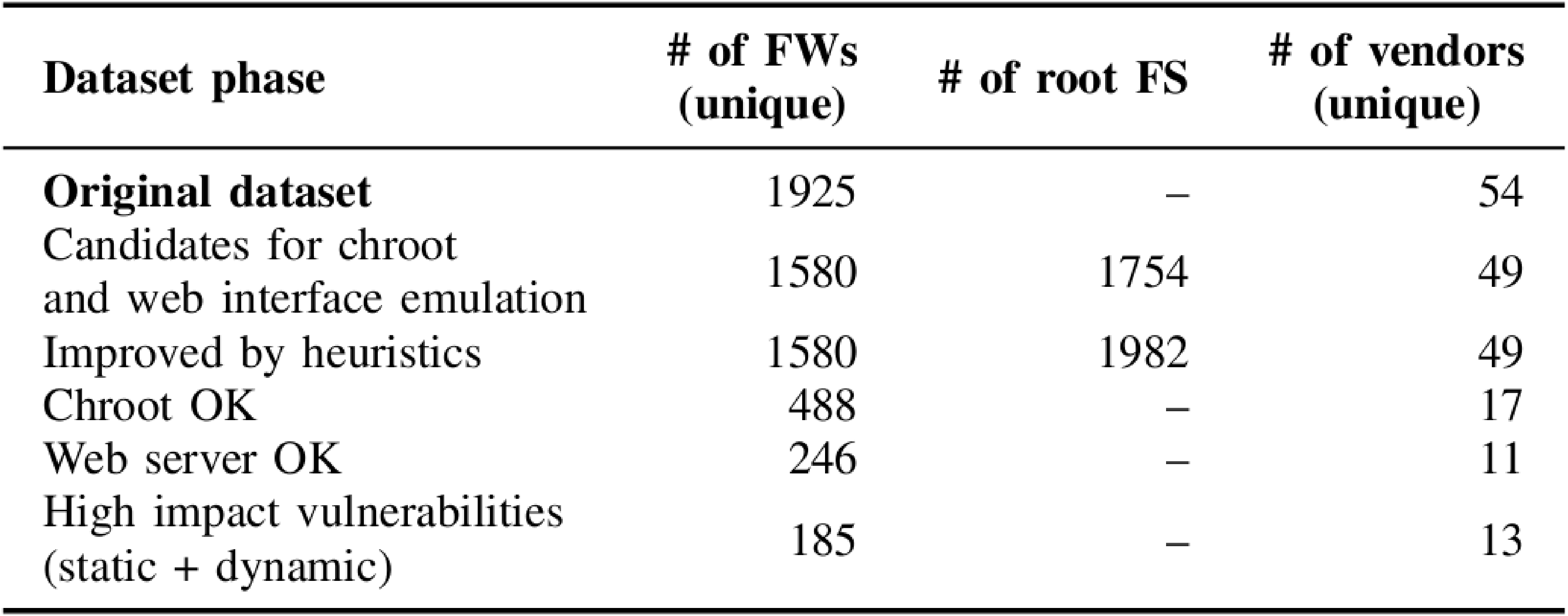
QEMU System Emulation: Original FW, Generic kernel, Chroot



QEMU System Emulation: Original FW, Generic kernel, Chroot



Datasets

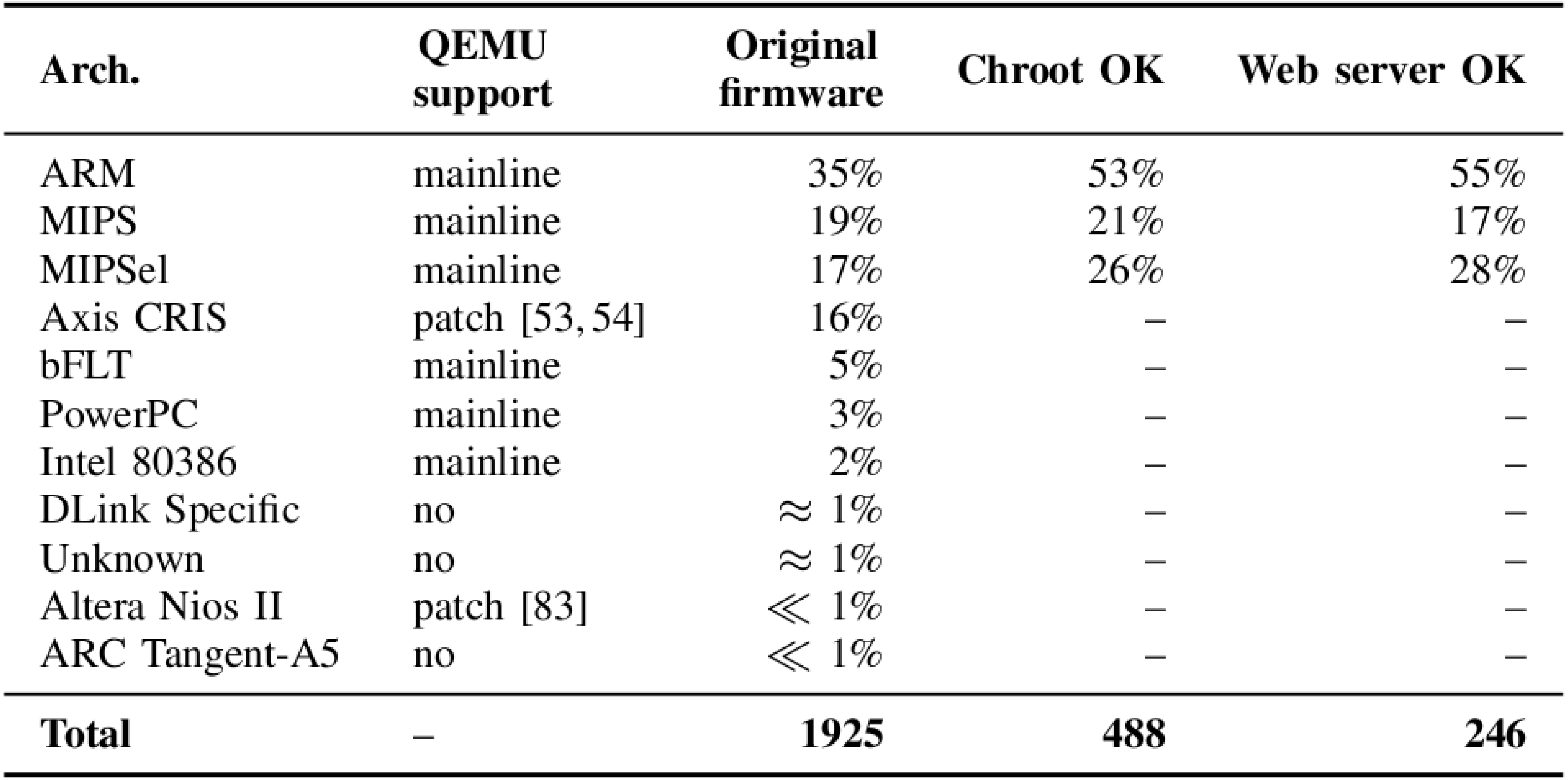


Dataset phase	# of FWs (unique)	# of root FS	# of vendors (unique)
Original dataset	1925	–	54
Candidates for chroot and web interface emulation	1580	1754	49
Improved by heuristics	1580	1982	49
Chroot OK	488	–	17
Web server OK	246	–	11
High impact vulnerabilities (static + dynamic)	185	–	13

Datasets: Failure Analysis

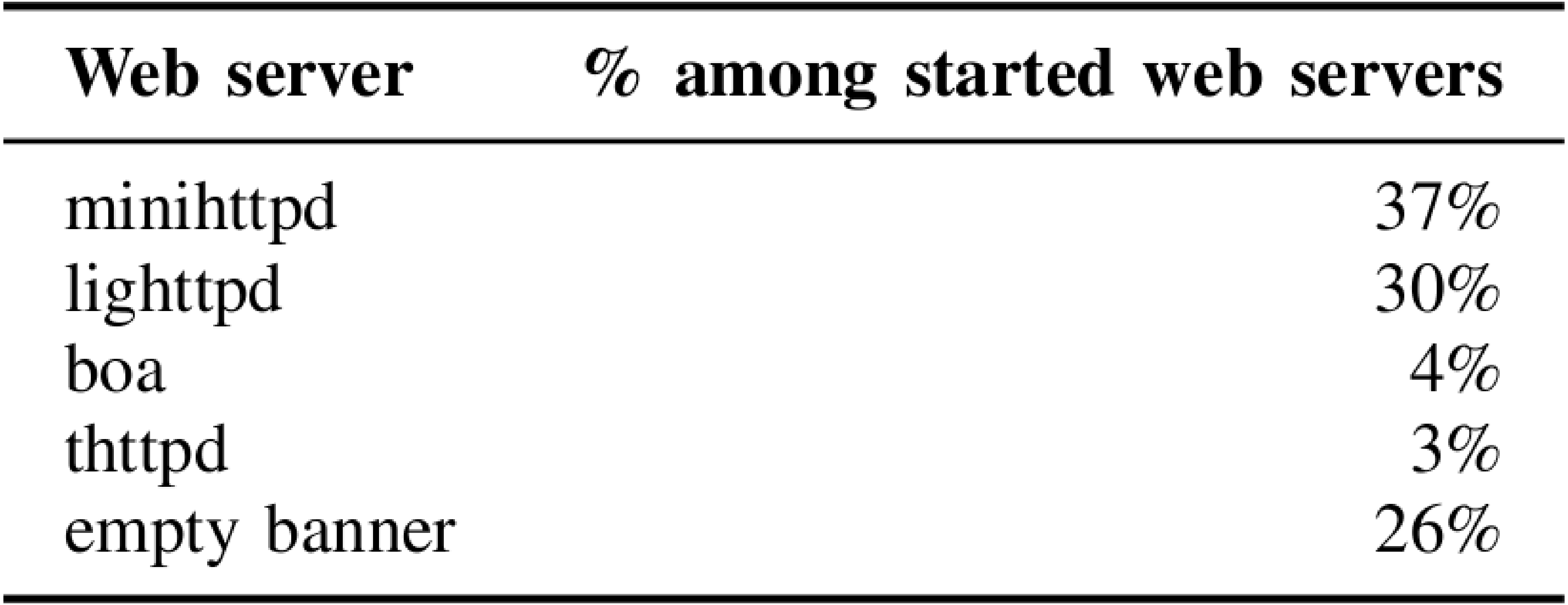
- Emulation failures limit the FW test coverage
  - “chroot failed” failures for 69% (or 1092) FWs
  - “webserver failed” failures for 50% (or 242) FWs
  - Failure analysis, random sampling
    - 95% confidence level and a ± 10% confidence interval for the accuracy of estimations
  - Fixing “chroot failed” should be relatively easy for 70.4% of the failures
  - Fixing “webserver failed” – should be relatively easy fir 34.8% of the failures

Datasets: Embedded CPU Architectures



Arch.	QEMU support	Original firmware	Chroot OK	Web server OK
ARM	mainline	35%	53%	55%
MIPS	mainline	19%	21%	17%
MIPSel	mainline	17%	26%	28%
Axis CRIS	patch [53, 54]	16%	–	–
bFLT	mainline	5%	–	–
PowerPC	mainline	3%	–	–
Intel 80386	mainline	2%	–	–
DLink Specific	no	$\approx$ 1%	–	–
Unknown	no	$\approx$ 1%	–	–
Altera Nios II	patch [83]	$\ll$ 1%	–	–
ARC Tangent-A5	no	$\ll$ 1%	–	–
Total	–	**1925**	**488**	**246**

Datasets: Embedded Web Servers



Web server	% among started web servers
minihttpd	37%
lighttpd	30%
boa	4%
thttpd	3%
empty banner	26%

Datasets: Embedded Network Services

- Network services – Fuzz 'em all!

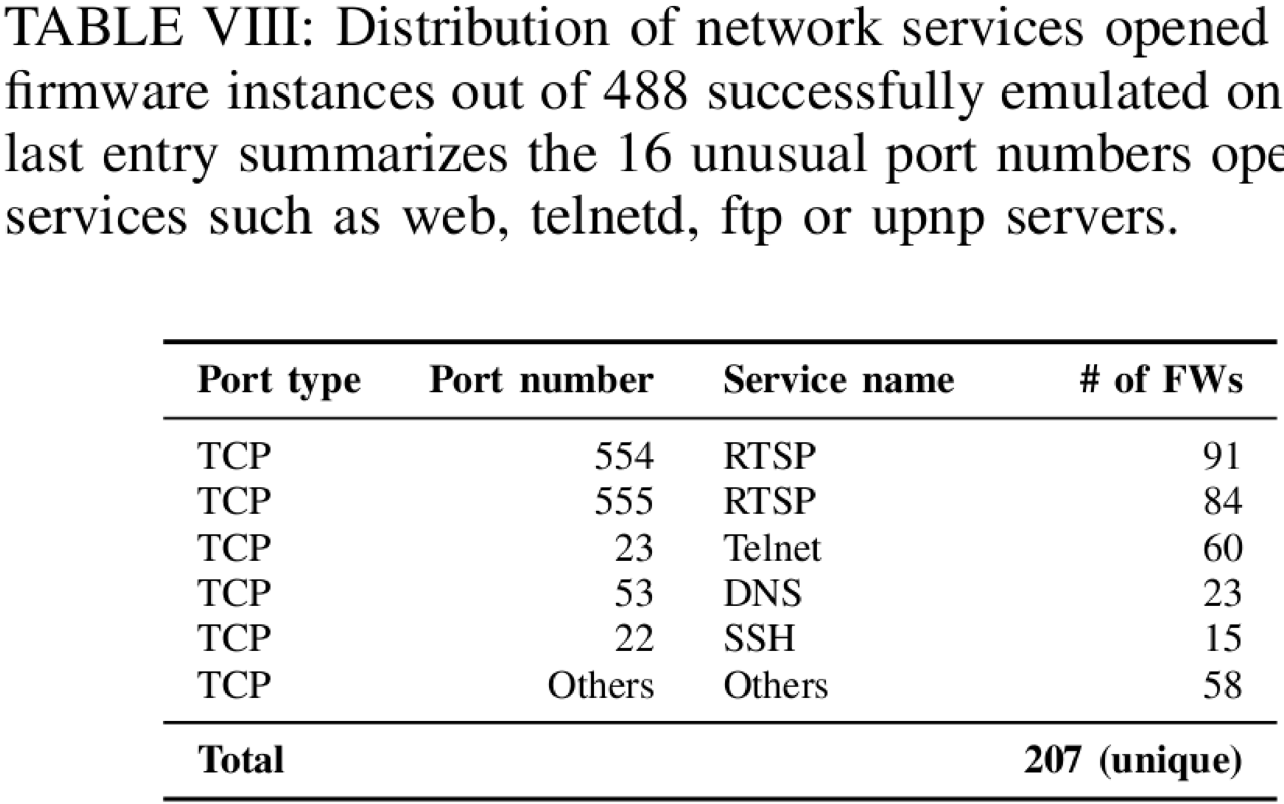
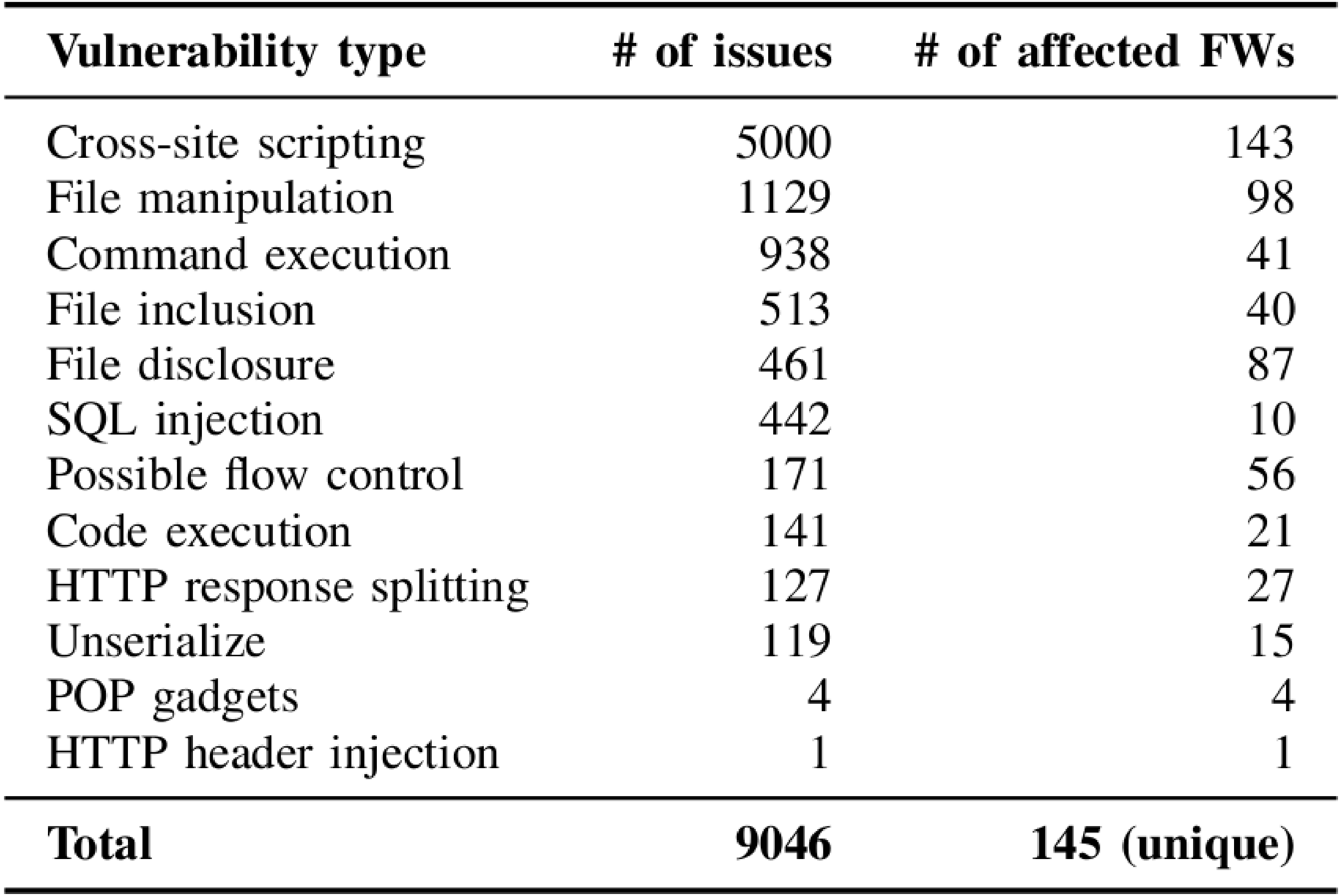


TABLE VIII: Distribution of network services opened by 207 firmware instances out of 488 successfully emulated ones. The last entry summarizes the 16 unusual port numbers opened by services such as web, telnetd, ftp or upnp servers.

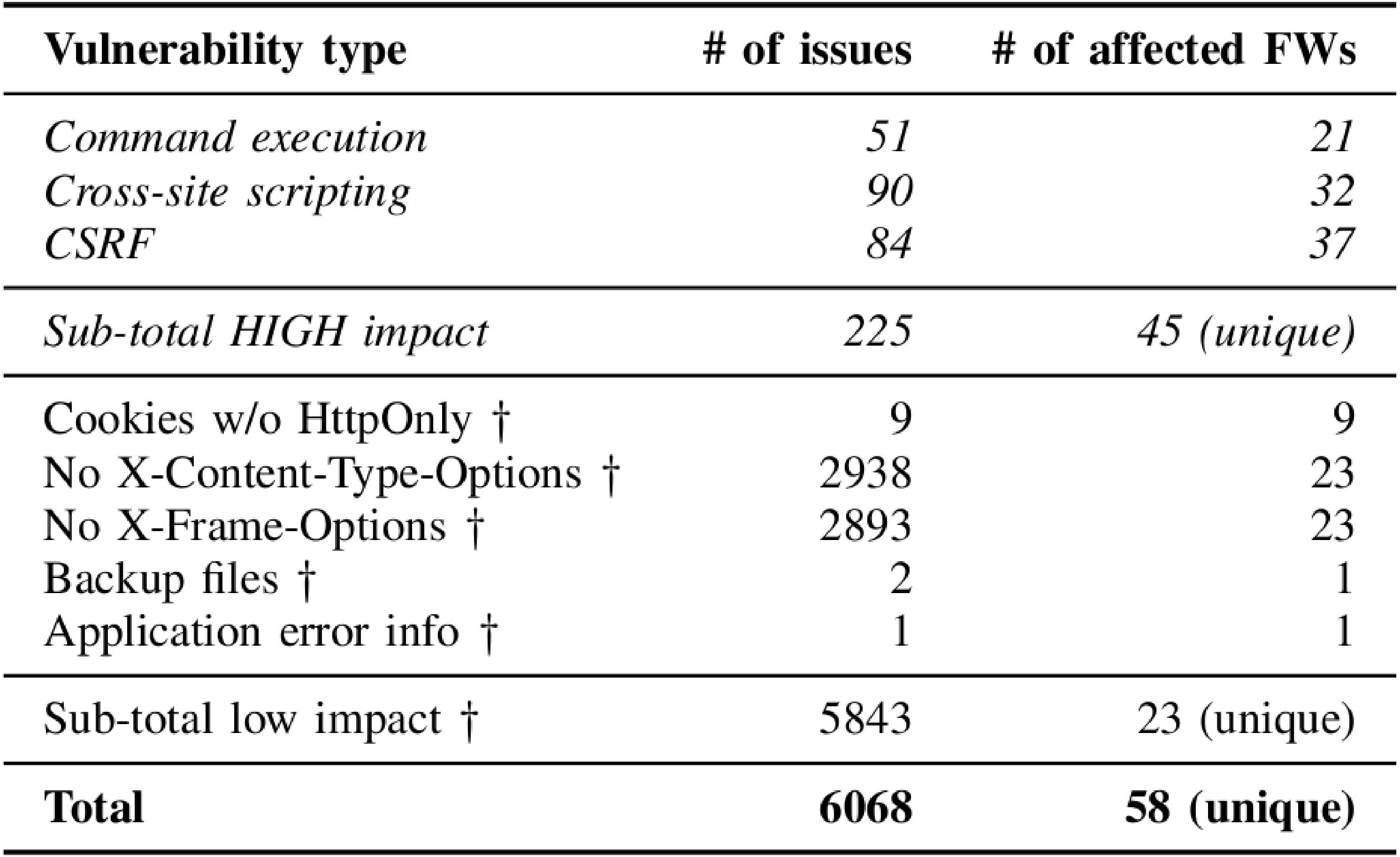
Port type	Port number	Service name	# of FWs
TCP	554	RTSP	91
TCP	555	RTSP	84
TCP	23	Telnet	60
TCP	53	DNS	23
TCP	22	SSH	15
TCP	Others	Others	58
Total			**207 (unique)**

Results: Static Analysis



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

Results: Dynamic Analysis



Vulnerability type	# of issues	# of affected FWs
Command execution	*51*	*21*
Cross-site scripting	*90*	*32*
CSRF	*84*	*37*
Sub-total HIGH impact	*225*	*45 (unique)*
Cookies w/o HttpOnly †	9	9
No X-Content-Type-Options †	2938	23
No X-Frame-Options †	2893	23
Backup files †	2	1
Application error info †	1	1
Sub-total low impact †	5843	23 (unique)
Total	**6068**	**58 (unique)**

Sample Vulnerabilities CVE-2011-1674 (existing)

- CVE-2011-1674
  - http://firmware.re/vulns/cve-2011-1674.php
- (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
    - http://WG103-DEVICE-IP/recreate.php?username=admin

Sample Vulnerabilities ACSA-2015-001 (0day)

- ACSA-2015-001
  - http://firmware.re/vulns/acsa-2015-001.php
  - 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

Sample Vulnerabilities ACSA-2015-001 (0day)

- 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

Sample Vulnerabilities ACSA-2015-001 (0day)

- Affected Modules (sha256)
  - 03bd170b6b284f43168dcf9de905ed33ae2edd721554cebec81894a8d5bcdea5
  - 2311b6a83298833d2cf6f6d02f38b04c8f562f3a1b5eb0092476efd025fd4004
  - 325c7fe9555a62c6ed49358c27881b1f32c26a93f8b9b91214e8d70d595d89bb
  - 33a29622653ef3abc1f178d3f3670f55151137941275f187a7c03ec2acdb5caa
  - 35c60f56ffc79f00bf1322830ecf65c9a8ca8e0f1d68692ee1b5b9df1bdef7c1
  - 40fbb495a60c5ae68d83d3ae69197ac03ac50a8201d2bccd23f296361b0040b9
  - 453658ac170bda80a6539dcb6d42451f30644c7b089308352a0b3422d21bdc01
  - 4679aca17917ab9b074d38217bb5302e33a725ad179f2e4aaf2e7233ec6bc842
  - 56714f750ddb8e2cf8c9c3a8f310ac226b5b0c6b2ab3f93175826a42ea0f4545
  - 70fe0274d6616126e758473b043da37c2635a871e295395e073fb782f955840e
  - 760bde74861b6e48dcbf3e5513aaa721583fbd2e69c93bccb246800e8b9bc1e6
  - 8bf836c5826a1017b339e23411162ef6f6acc34c3df02a8ee9e6df40abe681ff
  - 9f56e5656c137a5ce407eee25bf2405f56b56e69fa89c61cdfd65f07bc6600ef
  - a5ef01368da8588fc4bc72d3faaa20b21c43c0eaa6ef71866b7aa160e531a5b4
  - dcefcff36f2825333784c86212e0f1b73b25db9db78476d9c75035f51f135ef6

Sample Vulnerabilities ACSA-2015-002 (0day)

- ACSA-2015-002
  - 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
  - `curl --data 'reqMethod=json_cli_reqMethod" "json_cli_jsonData"; cat "/etc/passwd' http://NETGEAR-DEVICE-IP/login_handler.php`

Results: Summary

- 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)

Results: Summary

- 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

References

- ”Automated Dynamic Firmware Analysis at Scale: A Case Study on Embedded Web Interfaces” (ACM AsiaCCS 2016 to appear)
  - http://firmware.re/dynamicanalysis/
- ”A Large-Scale Analysis of the Security of Embedded Firmwares” (Usenix Security 2014)
  - http://firmware.re/usenixsec14/
- More: http://www.s3.eurecom.fr/~costin/

Tools

- 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-sc

Tools

- https://shodan.io
- https://zmap.io
- https://scans.io
- https://censys.io
- https://www.zoomeye.org/

Take Aways

- Large scale firmware analysis is absolutely necessary, especially with the IoT hype

Take Aways

- Large scale firmware analysis is absolutely necessary, especially with the IoT hype

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

Take Aways

- 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

Acknowledgements

- Dr. Jonas Zaddach
- Prof. Aurelien Francillon
- Prof. Davide Balzarotti
- Dr. Apostolis Zarras
- S3 SysSec research group

Feedback

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

@costinandrei