DEEPSEC

Large-scale Security Analysis of IoT Firmware

Daniel Nussko

Independent IT security researcher and penetration tester

Vienna, Austria

About me

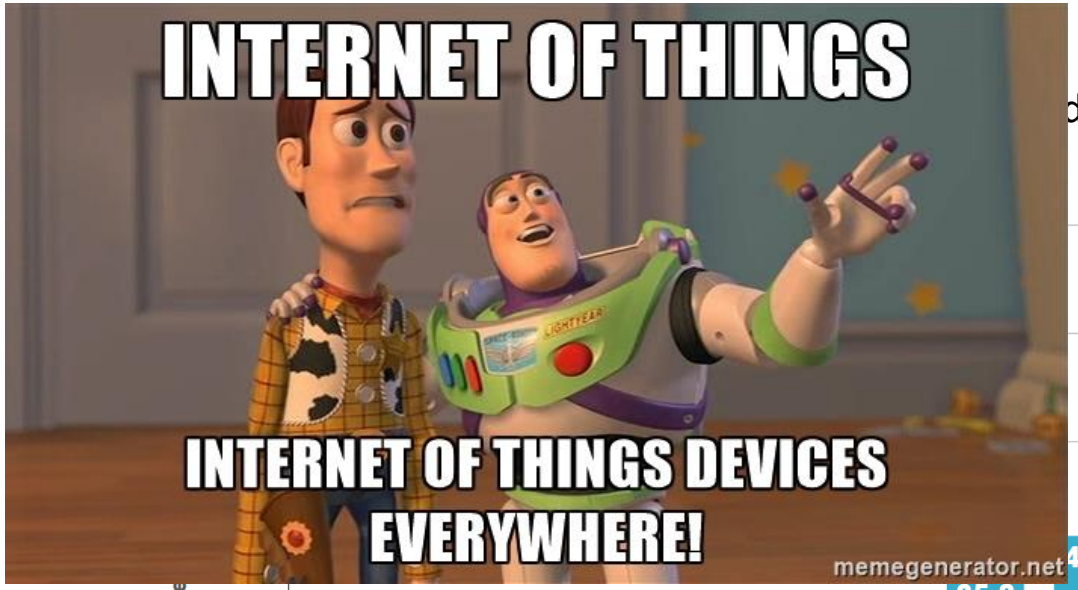
- Daniel Nussko
- Living in Heilbronn, Germany
- Independent IT security researcher and penetration tester
- Special interest in security of IoT devices

Landscape of IoT devices

Industrial IoT

Consumer & enterprise devices

Landscape of IoT devices



dwide from 2015 to 2025

Security problems of IoT devices

General reasons for bad security practices

- Hard competition, product security costs money
- IoT devices and their firmware are very diverse
- Short product lifecycle

Example: Penetration test of a security camera

- Buffer overflow in web interface
- Denial of Service via a crafted HTTP request
- Active debug interface

- Rebranded camera, vendor was not aware of that the original manufacturer
  - Implemented a debug interface
  - Released a new firmware version to fix vulnerabilities

Security analysis of IoT devices

- Penetration test
- Dynamic firmware analysis
- Static firmware analysis

Scope of the security analysis

Aim of this research

- Obtain a high-level overview of the security level of IoT firmware
- Analysis of a large number of firmware files
- Use an automated approach

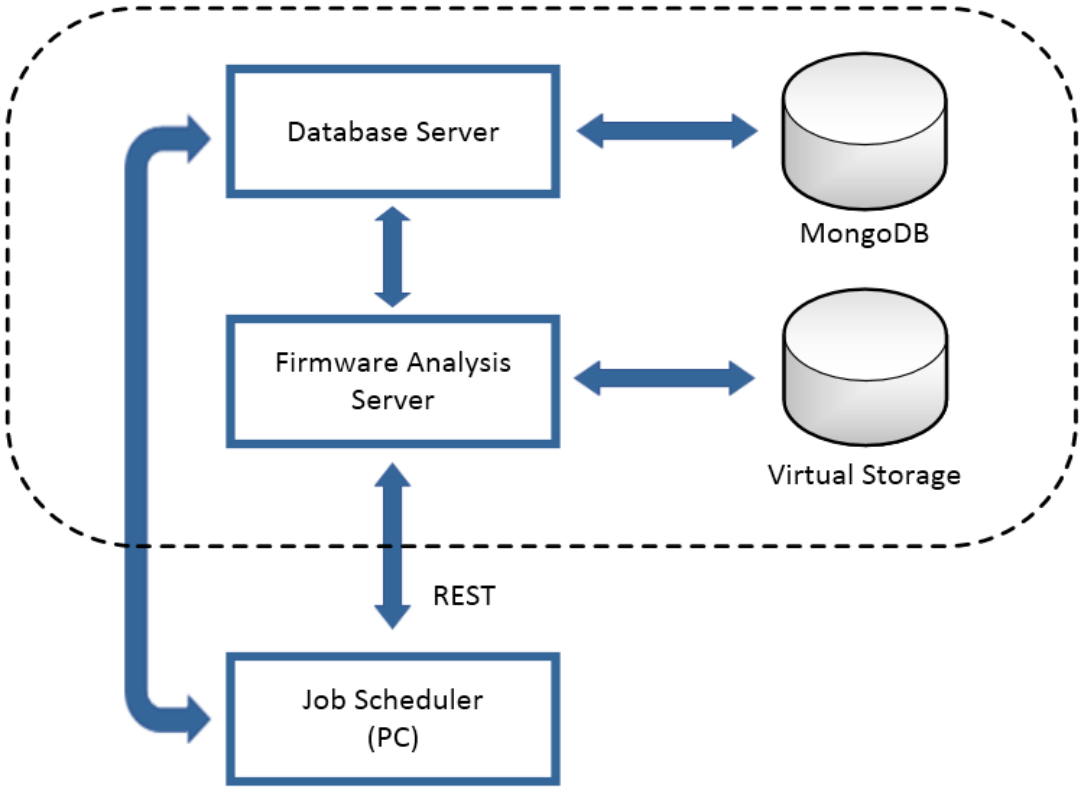
Analysis aspects

- Common binaries and libraries and their version
- Use of compiler-based exploit mitigation mechanisms in binaries and libraries
- Default user accounts and passwords
- Cryptographic material

Methodology for performing a large-scale analysis

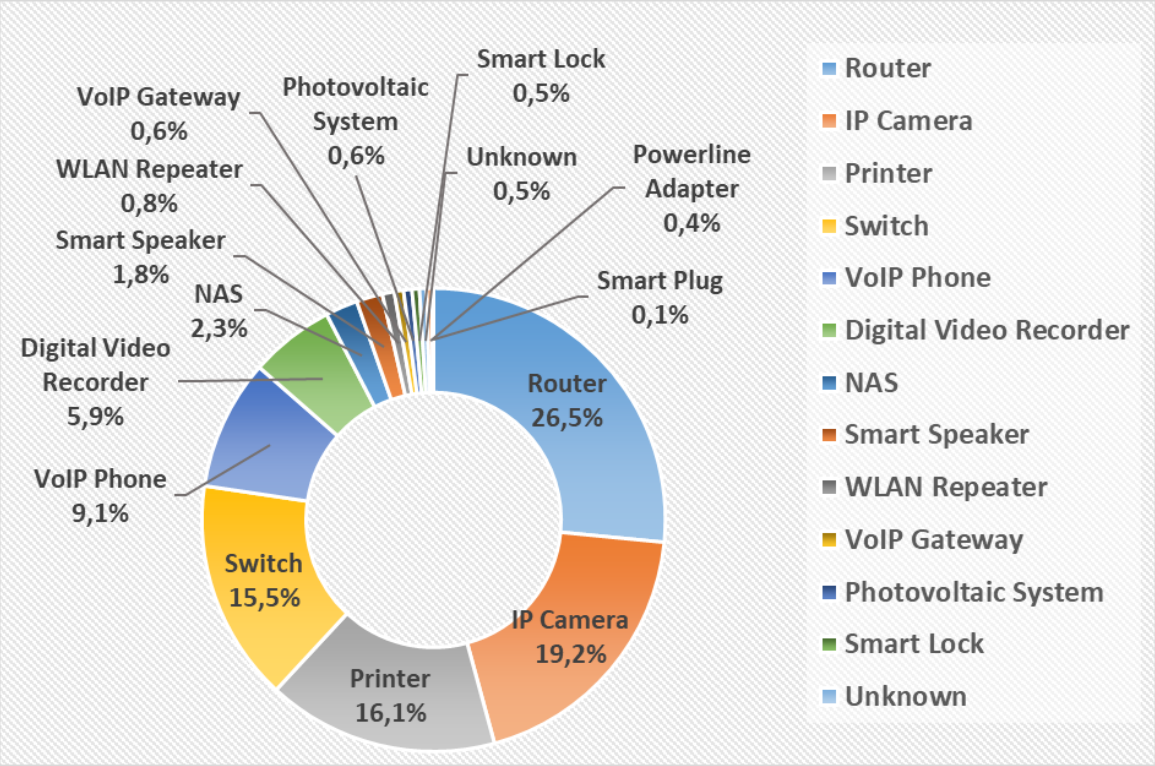
1. Collecting firmware images
2. Unpacking and data extraction
3. Security analysis
4. Storing results in database
5. Evaluation of results

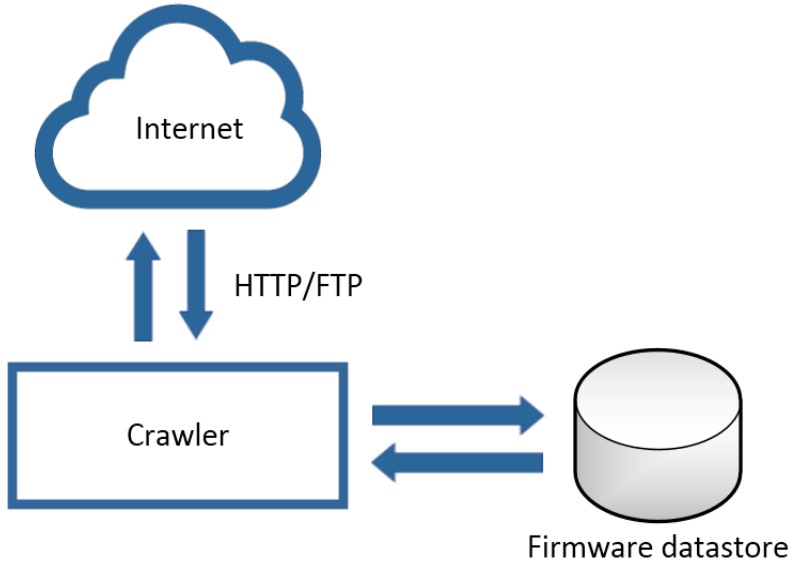
Architecture



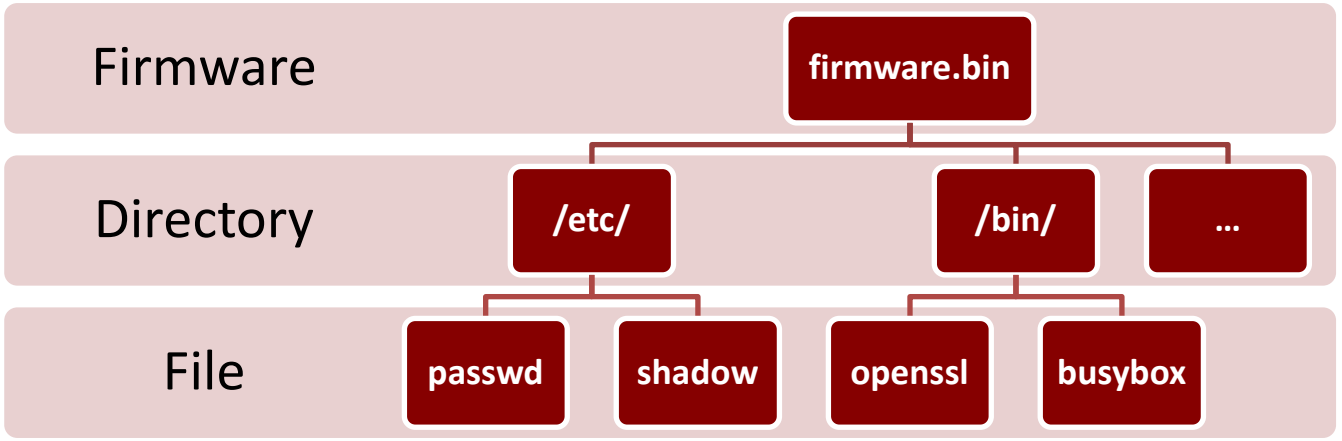
Firmware image collection

- Collection of 10K firmware images from 20 vendors
- Targeted crawling for firmware images on download pages of IoT vendors
- Scrapy Framework (Python)

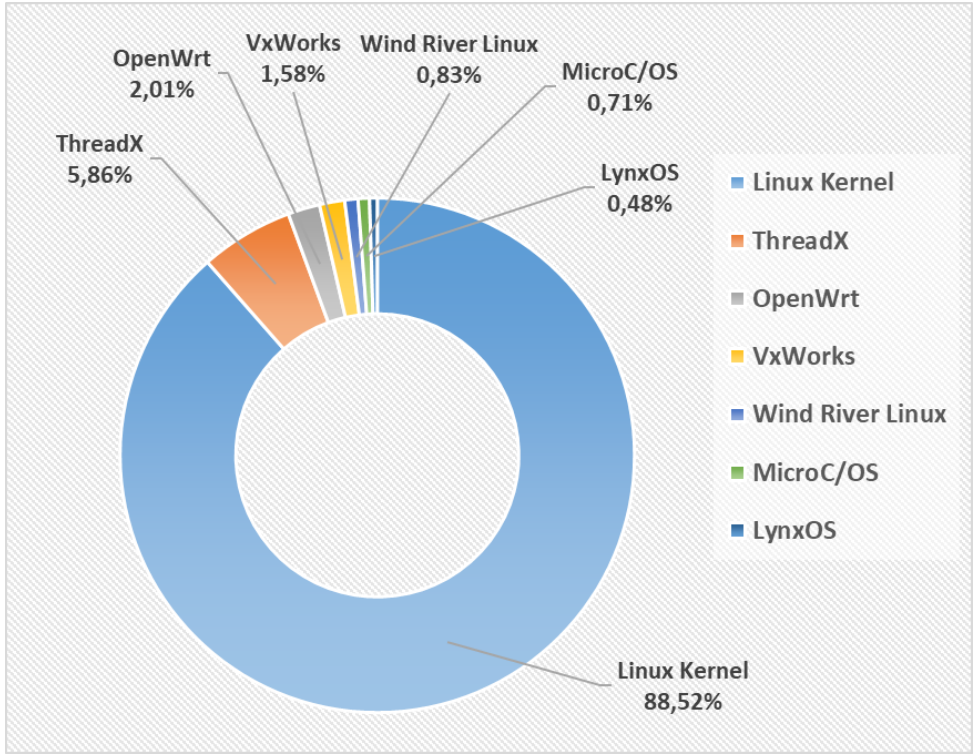


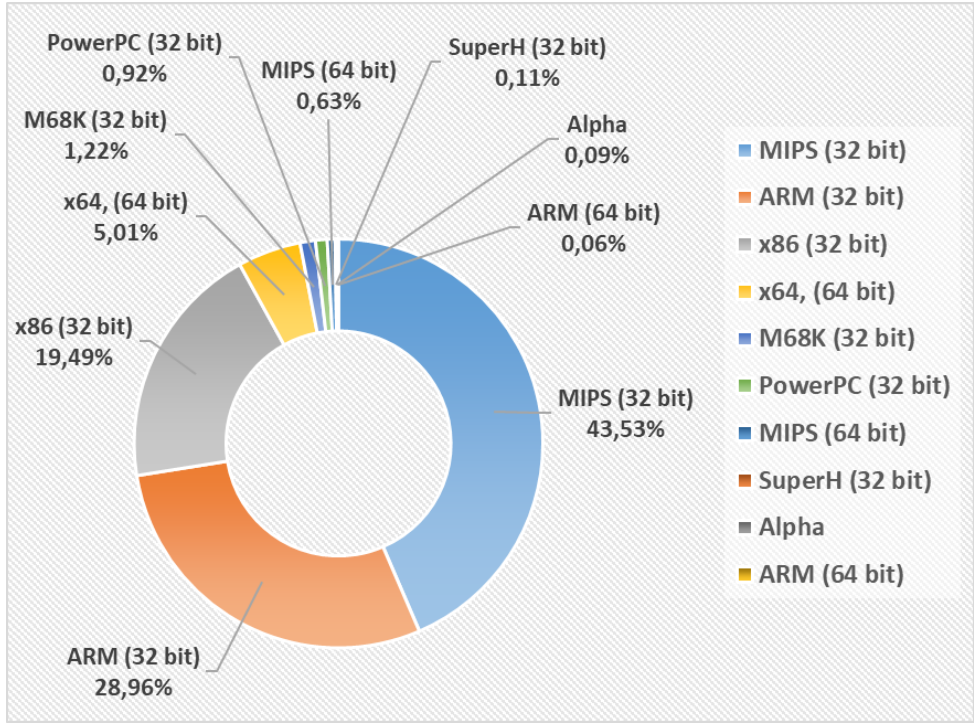


Unpacking the firmware

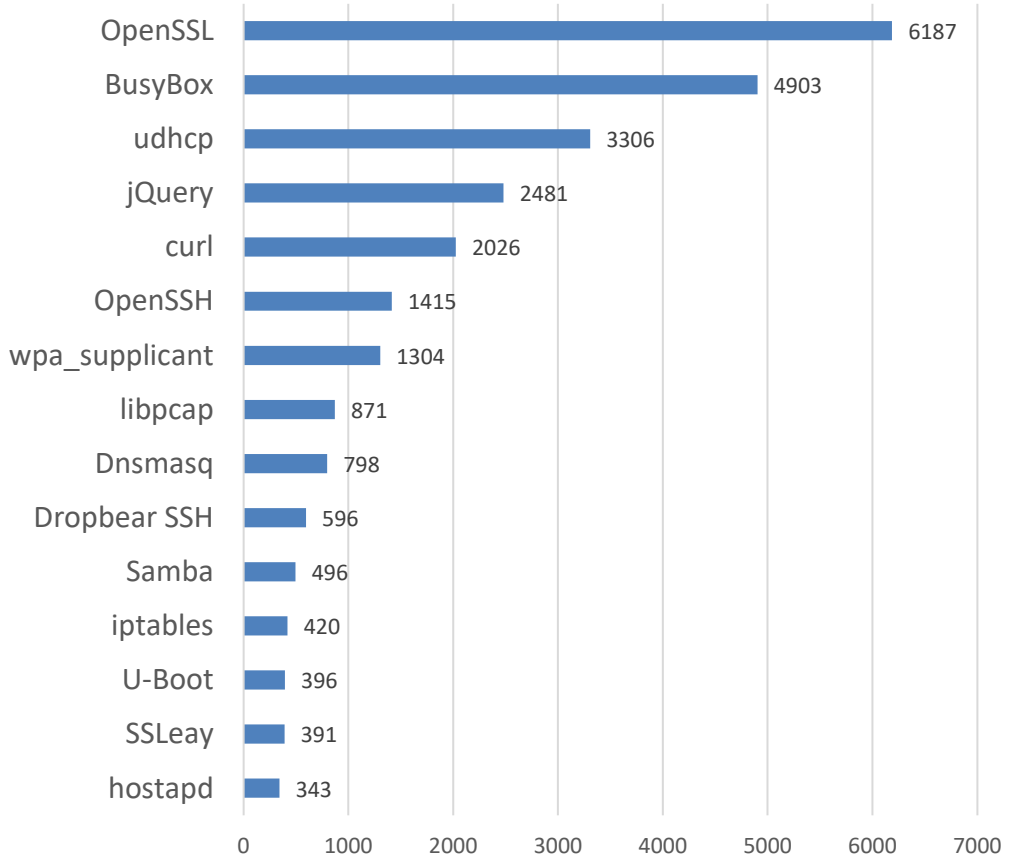


General observations





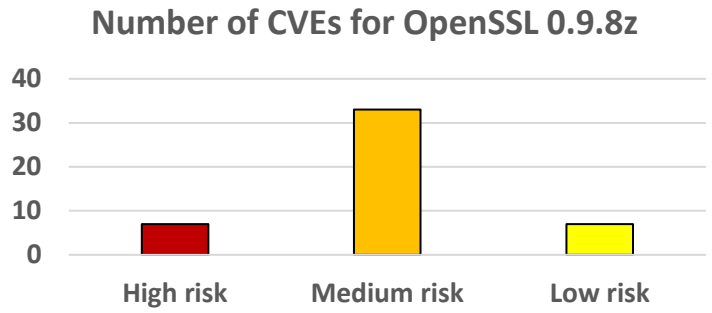
Identified software components



Outdated software components

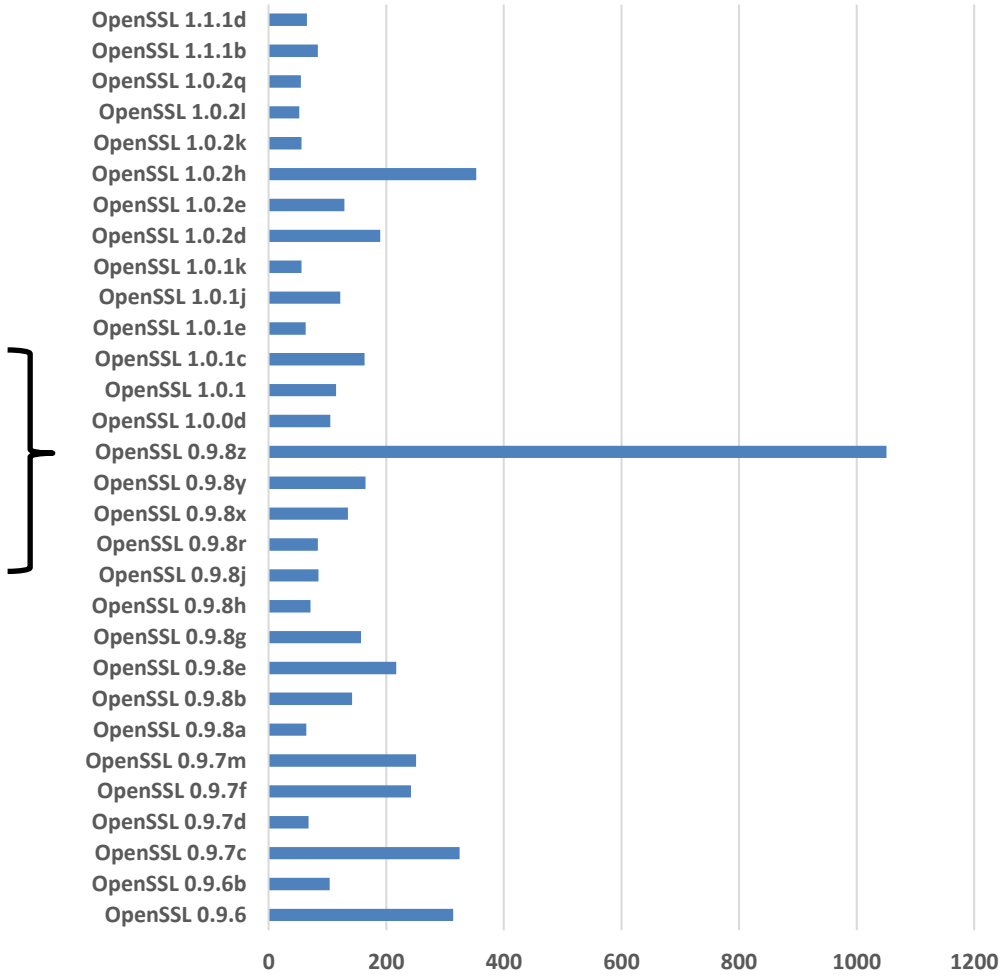
OpenSSL

- FREAK: 74.6% vulnerable
- POODLE: 49.3% vulnerable
- Heartbleed: 5.7% vulnerable

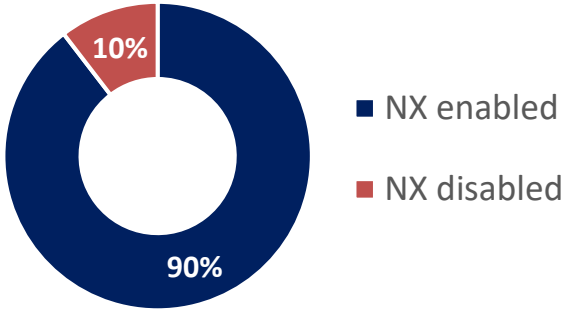


Linux kernel

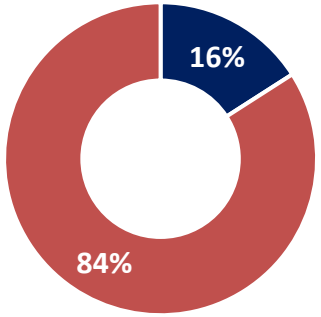
- 91.5% of kernel versions are EOL
- Bug fixes and security patches are no longer provided



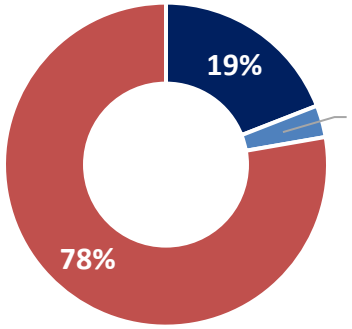
Exploit mitigation mechanisms



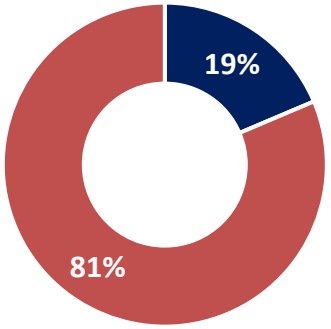
- NX enabled
- NX disabled



- Stack Smashing Protection enabled
- Stack Smashing Protection disabled

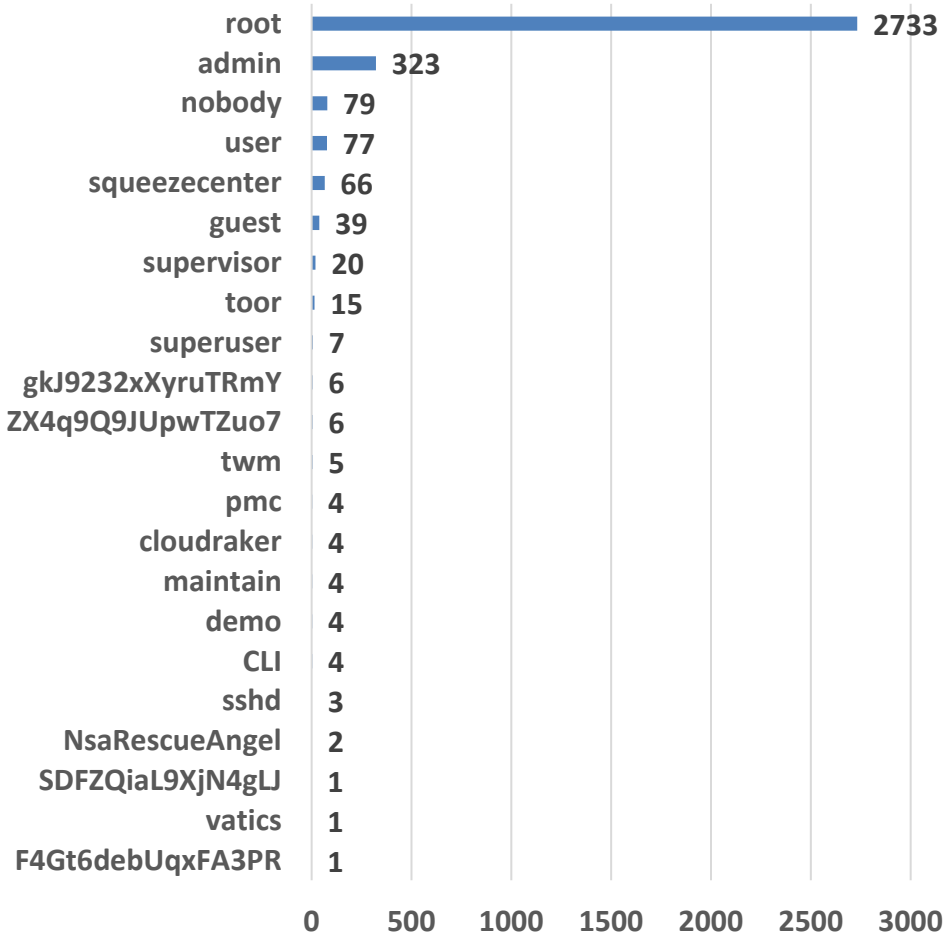


- RELRO fully enabled
- RELRO partially enabled
- RELRO disabled



- Fortify Source enabled
- Fortify Source disabled

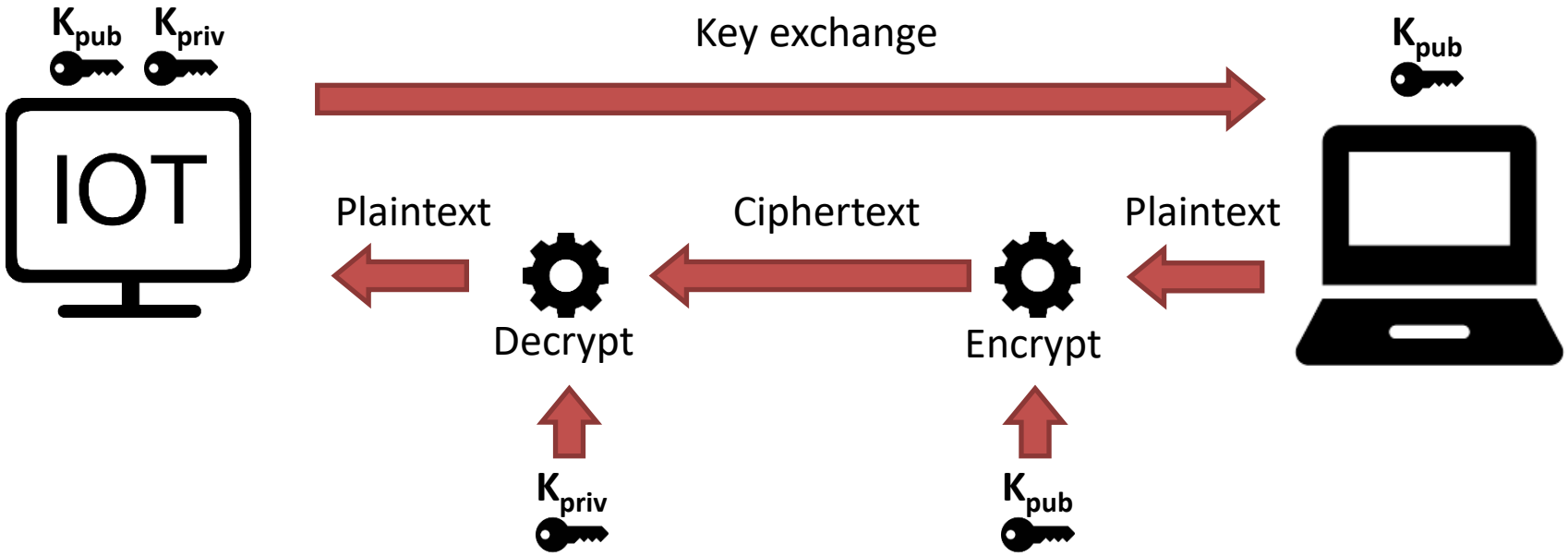
Factory default credentials



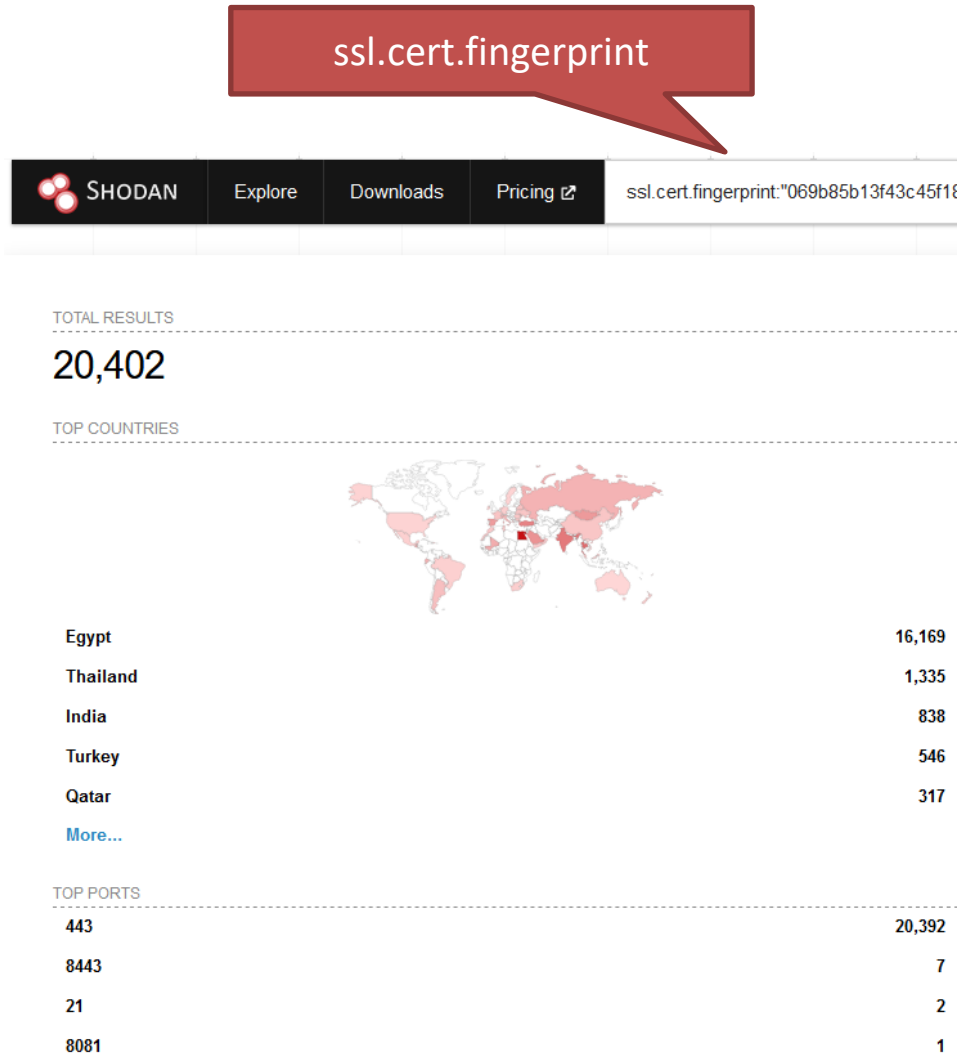
Identified cryptographic material

Type of cryptographic material	Number
TLS certificate	5171
PKCS8 private key	1484
RSA private key	843
Generic public key	158
RSA public key	94
Generic private key	91
PGP public key	4

Problem of hard-coded asymmetric keys



Problem of hard-coded asymmetric keys



Identification of backdoors

SSH public keys in „authorized_keys“ file

- Dropbear SSH server
- Allows key-based authentication

Backdoor in web interface

- Identified in firmware of 9 different products
- CGI script in web interface
- Enables SSH/Telnet access with user ‚NsaRescueAngel‘
- Vendor released a new firmware
- CVE numbers were assigned

Use of unusual software components

Tcpdump

- Used to capture network traffic
- Identified in 500 firmware images of 8 vendors
- Mainly identified in firmware of routers and VoIP phones

GDB

- Debugger, used for runtime analysis of executables
- Identified in 861 firmware images of 9 vendors
- Mainly identified in firmware of IP cameras, routers and switches

Concluding remarks

- 10K firmware images were analyzed
- Use of outdated software components and Linux kernels
- Use of hard-coded passwords and keys
- Identification of specific models by their TLS certificate
- Built-in backdoors are a real problem