Leaking Credentials

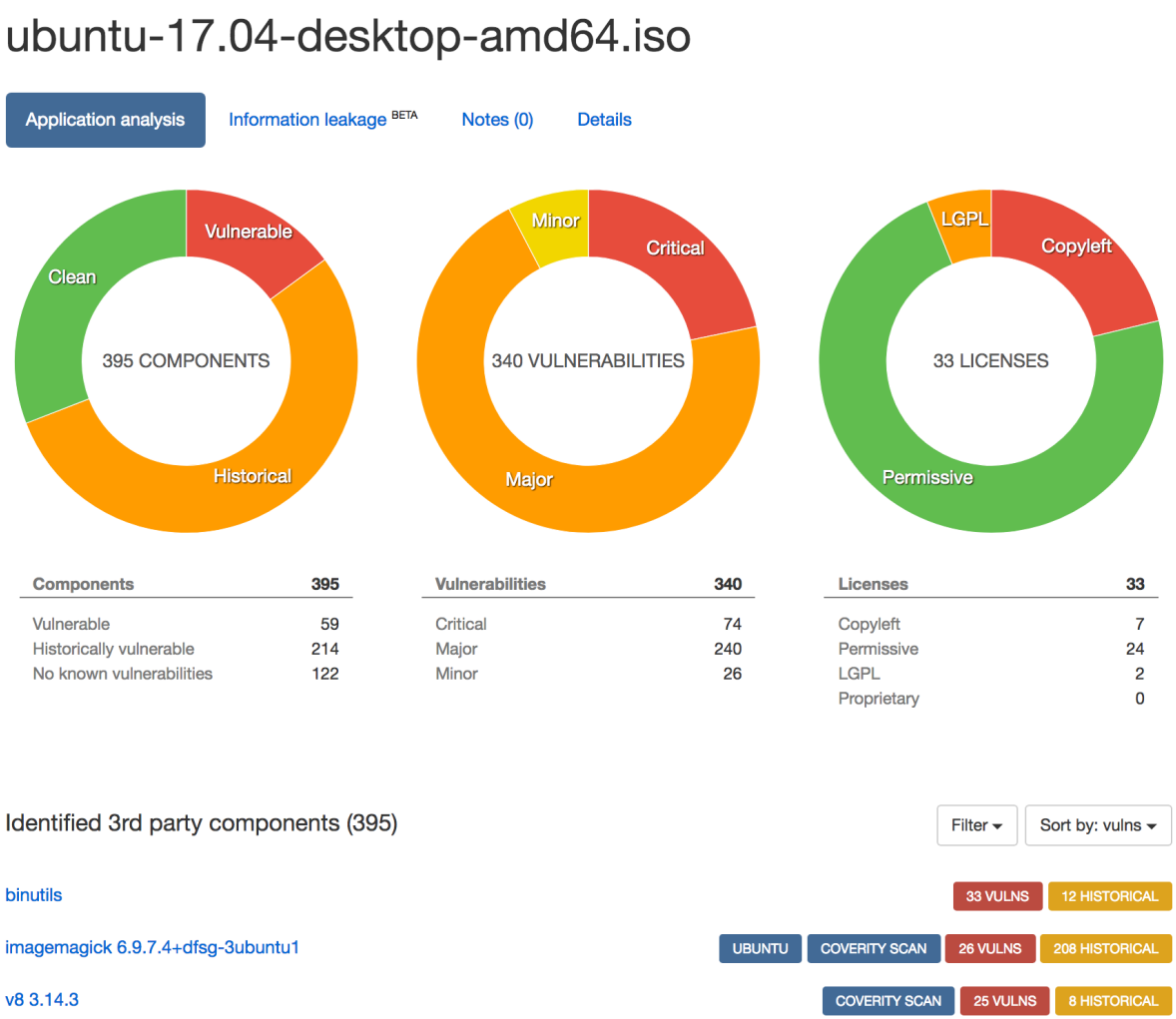
A security malpratice more common than expected

Bogdan Mihaila

SYNOPSYS®
Silicon to Software™

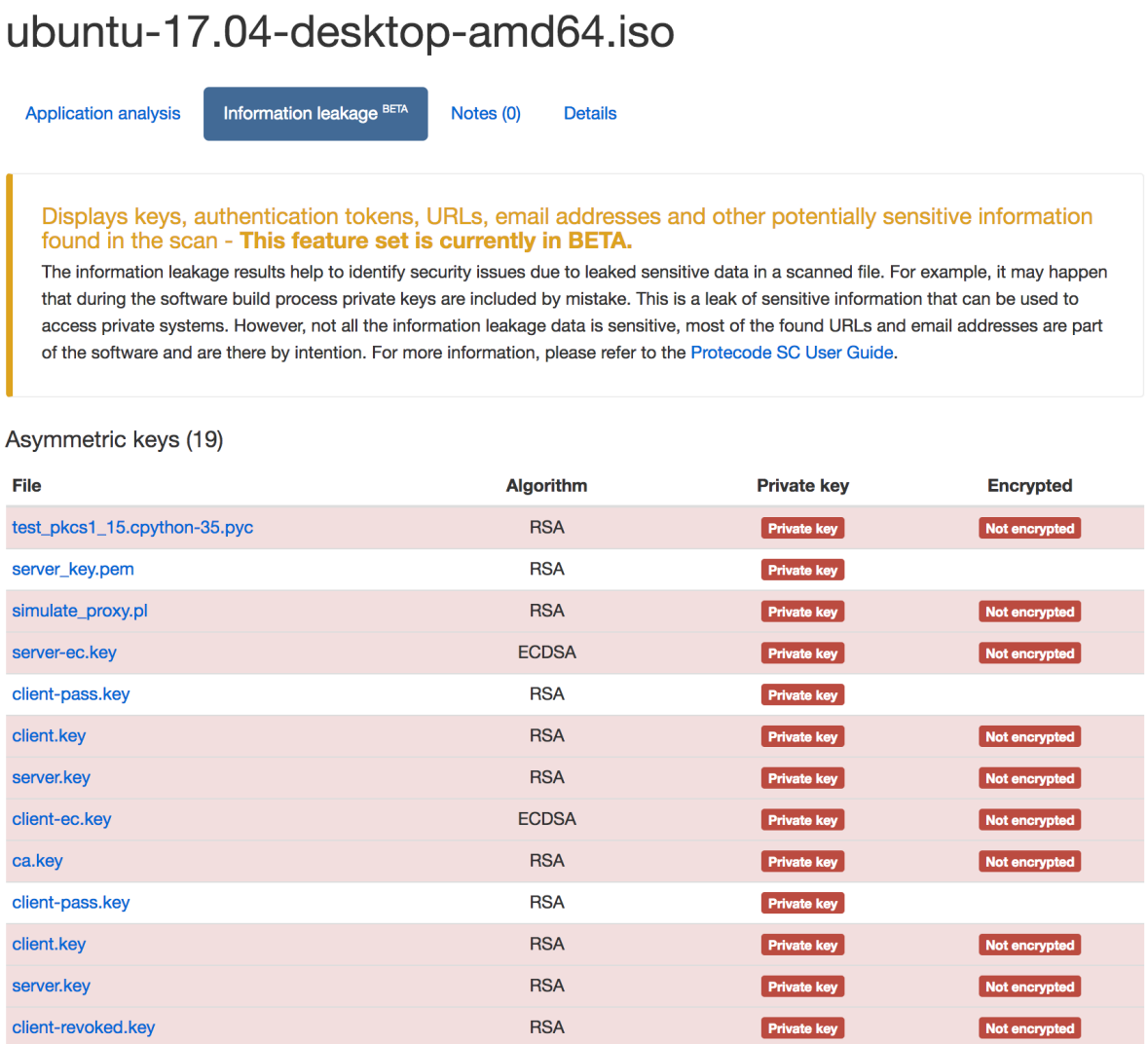
About the tool

We develop a tool “Protecode” that finds known Vulnerabilities (CVEs).
Think of it as a better and commercial version of “OWASP Dependency Check”



New research feature

We lately added the scraping of binaries for credentials, encryption keys and other potential leaks to the tool: the “Information leakage” feature



ubuntu-17.04-desktop-amd64.iso

Application analysis | Information leakage BETA | Notes (0) | Details

Displays keys, authentication tokens, URLs, email addresses and other potentially sensitive information found in the scan - **This feature set is currently in BETA.**

The information leakage results help to identify security issues due to leaked sensitive data in a scanned file. For example, it may happen that during the software build process private keys are included by mistake. This is a leak of sensitive information that can be used to access private systems. However, not all the information leakage data is sensitive, most of the found URLs and email addresses are part of the software and are there by intention. For more information, please refer to the Protecode SC User Guide.

Asymmetric keys (19)

File	Algorithm	Private key	Encrypted
test_pkcs1_15.cpython-35.pyc	RSA	Private key	Not encrypted
server_key.pem	RSA	Private key	
simulate_proxy.pl	RSA	Private key	Not encrypted
server-ec.key	ECDSA	Private key	Not encrypted
client-pass.key	RSA	Private key	
client.key	RSA	Private key	Not encrypted
server.key	RSA	Private key	Not encrypted
client-ec.key	ECDSA	Private key	Not encrypted
ca.key	RSA	Private key	Not encrypted
client-pass.key	RSA	Private key	
client.key	RSA	Private key	Not encrypted
server.key	RSA	Private key	Not encrypted
client-revoked.key	RSA	Private key	Not encrypted

WHY?

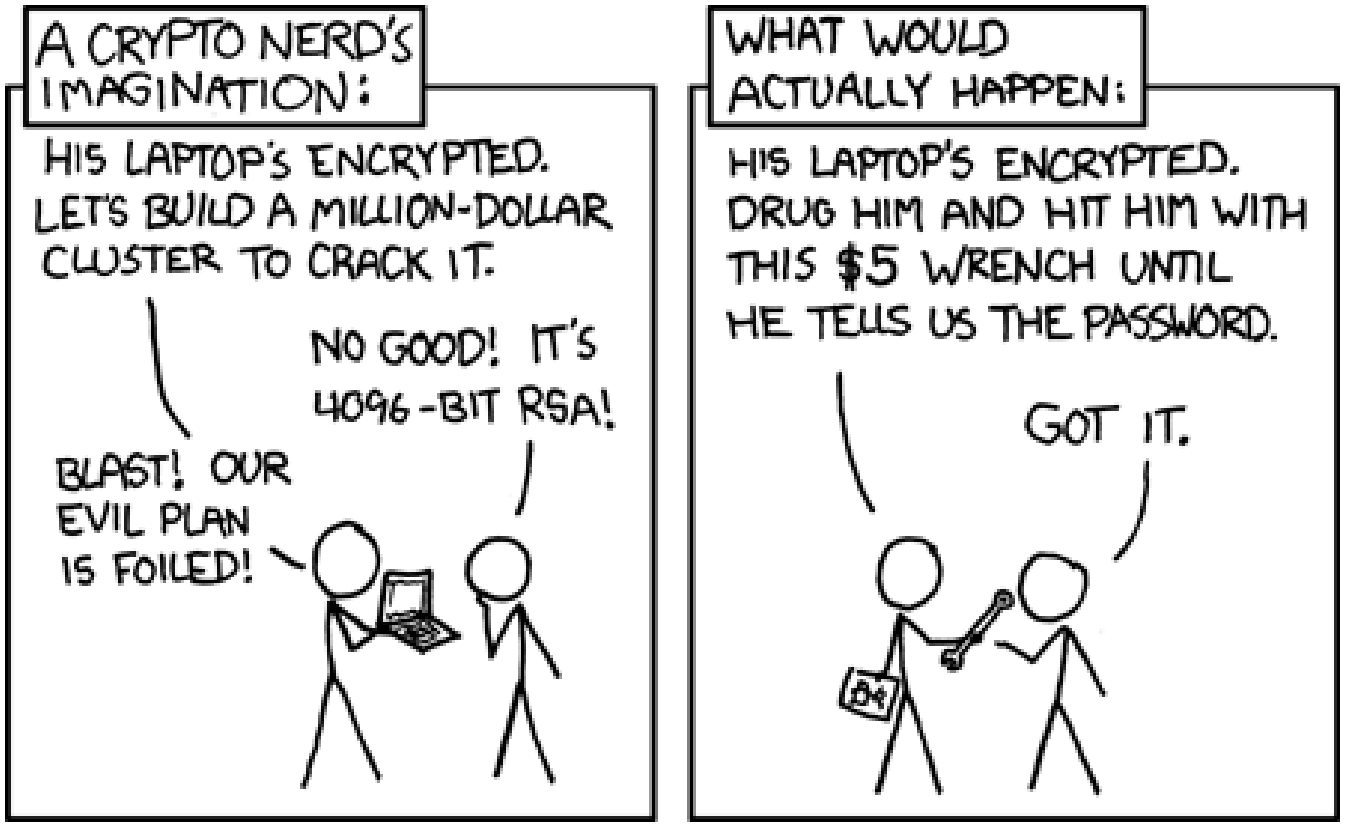
Because WHY NOT!

and well, we are security folks, soo . . .

and due to the XKCD security wisdom! (on next page)

XKCD security wisdom

From the department of good security analogies



https://xkcd.com/538/

How do you own the Internet?

Some examples:

- you are Tavis Ormandy and find vulnerabilities in all AV software
- you are Tavis Ormandy and find leaked credentials in webcaches
- you are the NSA and do this for a living
- you are the WannaCry Botnet and use known vulnerabilities
- you are the Mirai Botnet and use known credentials

Why is the “Botnet approach” becoming more common

- XKCD security wisdom
- proliferation of automated pentesting tools and script kiddies
- IoT - enough said!
- patching practice of legacy devices - see IoT above

Why are credentials left in deployed software

- different systems are used for devel, deploy, test, ...
  each requiring different credentials
- automation is used in continuous development (CI/CD)
  $\rightsquigarrow$ scripts need to access all those different systems
  so credentials are stored along
- it is hard to keep track of all the sensitive information for humans
- it is sometimes impossible to know where credentials land due to
  - logging
  - temp files
  - shell history
  - memory dumps
  - hidden files
  - filesystem artifacts (e.g. `rm` in a Dockerfile!)
  - version control systems (e.g. Github commits with "`removed` ...")
  - ...

What can you do with the leaked stuff?

> **Passwords**

- AWS secrects
- root passwords
- user passwords

⇒ **free lunch!**

What can you do with the leaked stuff?

> Private Keys

- used to access other remote systems? → see "**Passwords**"
- used to encrypt communication with current device, aka `host_keys`?
  → MITM (man in the middle)
  → scan the net to find devices using the key! moar MITM

⇒ **moar free lunch!**

What can you do with the leaked stuff?

> Public Keys

- used for trusted hosts, aka `authorized_keys`?
  → scan the net to find authorized devices!
  → see if corresponding private key can be found and MITM
  → modify in firmware and MITM updates etc.

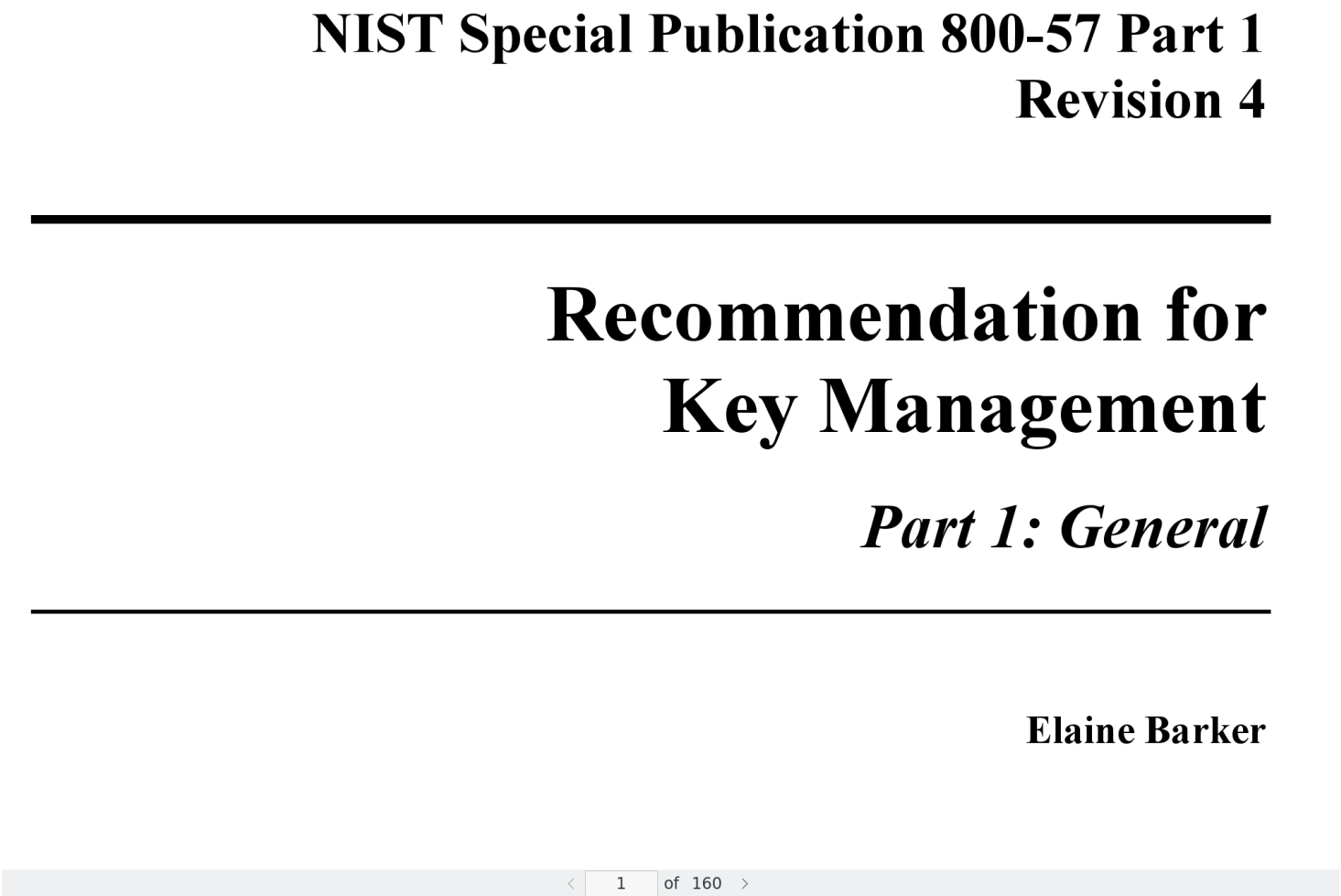
⇒ **work for your lunch!**

Why is it bad?

- passwords $\rightsquigarrow$ free lunch
- private keys $\rightsquigarrow$ free lunch
- public keys $\rightsquigarrow$ maybe some lunch someday
- . . . $\rightsquigarrow$ various shades of lunch
- emails, urls, . . . $\rightsquigarrow$ discloses new attack vectors

How to do it right? Guidelines

"Part 1" is already 160 pages long!



NIST Special Publication 800-57 Part 1
Revision 4

Recommendation for Key Management

Part 1: General

Elaine Barker

1 of 160

Key leaks that got public

Many public leaks over the last years!

- Github starts to alert on leaked ssh keys and auth. tokens (2016)
- Router firmwares ship with default ssh keys (2014)
- Google Play apps ship with remote auth. tokens (2013)
- (un)intentional backdoors in software! (over the years)
- . . . many more examples (next slides)

Key leaks that got public

itnews

GOVERNMENT IT | INFOSEC | FINANCE IT | TELCO | BENCHMARK AWARDS | Log In | Newsletter

AWS urges developers to scrub GitHub of secret keys

By Munir Kotadia
Mar 24 2014
10:18AM

SECURITY IS POWERED BY SC MAGAZINE

RELATED ARTICLES

Qld Health replaces firewall after WannaCry

Devs hit with unexpected bills after leaving secret keys exposed.

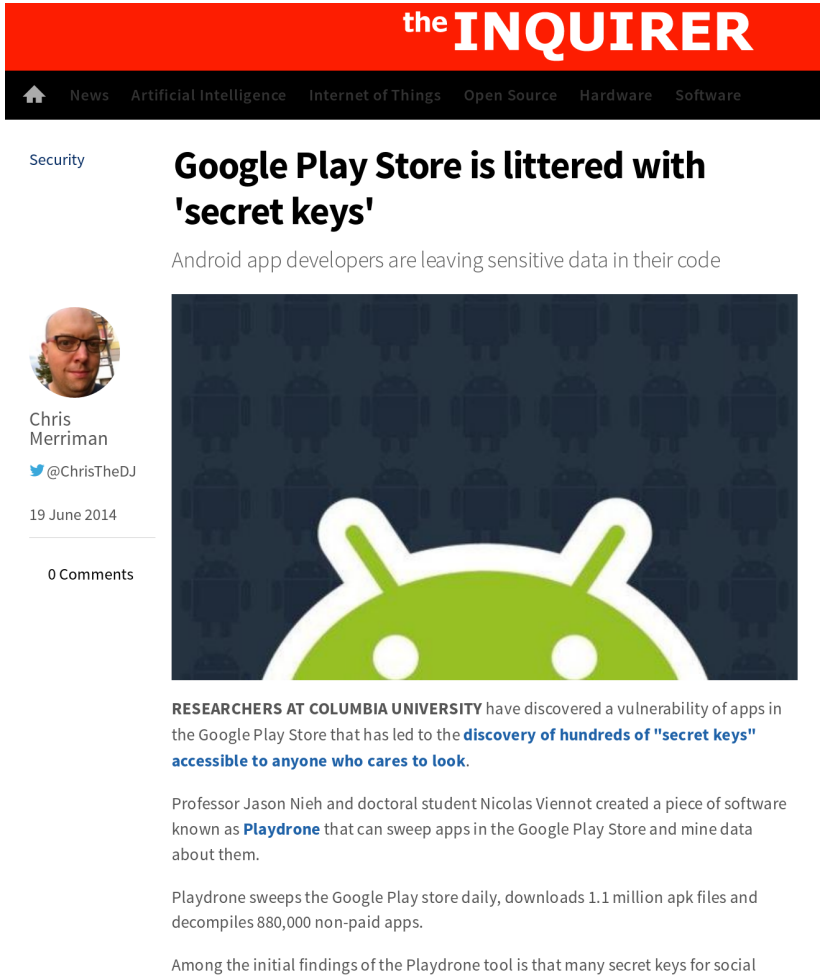
Amazon Web Services (AWS) is urging developers using the code sharing site GitHub to check their posts to ensure they haven't inadvertently exposed their log-in credentials.

Thousands of 'secret keys', which unlock access to private Amazon Web Services accounts are currently available unencrypted to members of the public with just two clicks of a mouse.

The secret keys are issued by Amazon Web Services when users open an account and provide applications access to AWS resources.

https://www.itnews.com.au/news/aws-urges-developers-to-scrub-github-of-secret-keys-375785

Key leaks that got public



the INQUIRER

News Artificial Intelligence Internet of Things Open Source Hardware Software

Security

Google Play Store is littered with 'secret keys'

Android app developers are leaving sensitive data in their code

Chris Merriman

@ChrisTheDJ

19 June 2014

0 Comments

RESEARCHERS AT COLUMBIA UNIVERSITY have discovered a vulnerability of apps in the Google Play Store that has led to the **discovery of hundreds of "secret keys" accessible to anyone who cares to look**.

Professor Jason Nieh and doctoral student Nicolas Viennot created a piece of software known as **Playdrone** that can sweep apps in the Google Play Store and mine data about them.

Playdrone sweeps the Google Play store daily, downloads 1.1 million apk files and decompiles 880,000 non-paid apps.

Among the initial findings of the Playdrone tool is that many secret keys for social

https://www.theinquirer.net/inquirer/news/2351080/google-play-store-is-littered-with-secret-keys

Key leaks that got public

NETWORKWORLD FROM IDG

Home > Security

Millions of embedded devices use the same hard-coded SSH and TLS private keys

The keys were hard-coded by manufacturers and can be used by attackers to launch man-in-the-middle attacks

By Lucian Constantin
Romania Correspondent, IDG News Service | NOV 26, 2015 6:56 AM PT

Credit: Michael Homnick / PC World

Thousands of routers, modems, IP cameras, VoIP phones and other embedded devices share the same hard-coded SSH (Secure Shell) host keys or HTTPS (HTTP

RELATED


Is BYOK the key to secure cloud computing?

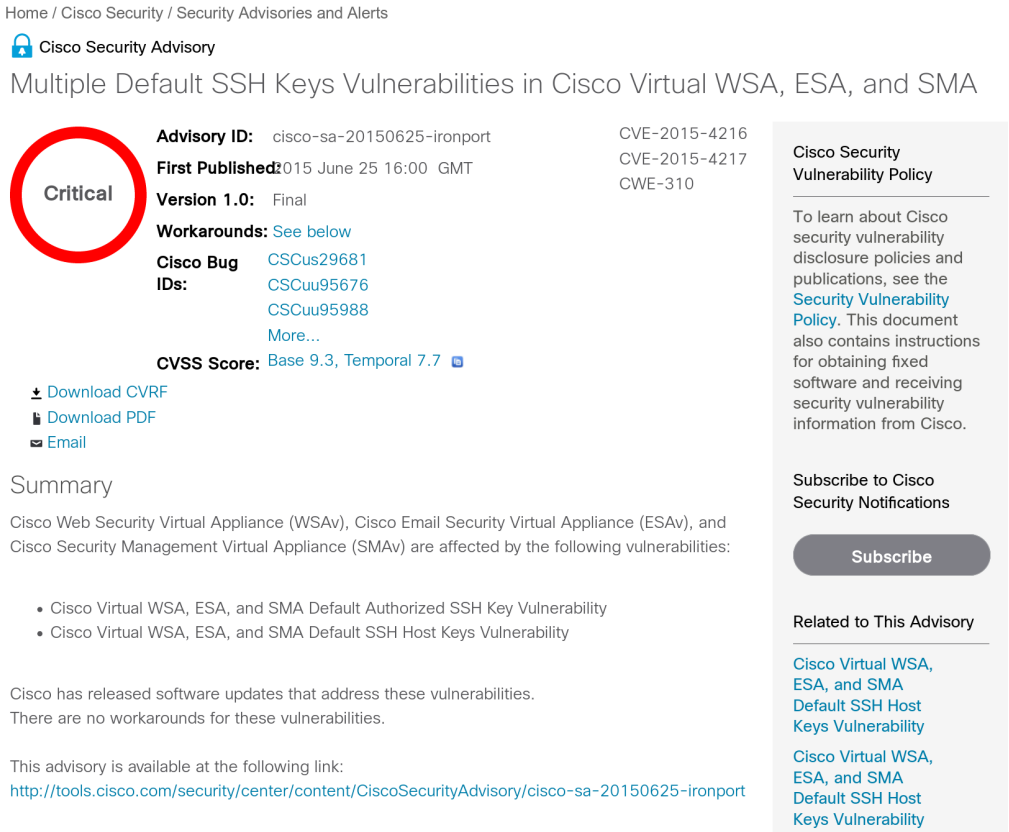

Review: Password managers help keep hackers at bay


Essential tools for everyday encryption


VIDEO
Setting up DLP features for email security.

http://www.networkworld.com/article/3009139/millions-of-embedded-devices-use-the-same-hard-coded-ssh-and-tls-private-keys.html

Key leaks that got public



Home / Cisco Security / Security Advisories and Alerts

Cisco Security Advisory

Multiple Default SSH Keys Vulnerabilities in Cisco Virtual WSA, ESA, and SMA

Critical

Advisory ID: cisco-sa-20150625-ironport
First Published: 2015 June 25 16:00 GMT
Version 1.0: Final
Workarounds: See below
Cisco Bug IDs: CSCus29681 CSCuu95676 CSCuu95988 More...
CVSS Score: Base 9.3, Temporal 7.7

CVE-2015-4216
CVE-2015-4217
CWE-310

Download CVRF
Download PDF
Email

Summary

Cisco Web Security Virtual Appliance (WSAv), Cisco Email Security Virtual Appliance (ESAv), and Cisco Security Management Virtual Appliance (SMAv) are affected by the following vulnerabilities:

- Cisco Virtual WSA, ESA, and SMA Default Authorized SSH Key Vulnerability
- Cisco Virtual WSA, ESA, and SMA Default SSH Host Keys Vulnerability

Cisco has released software updates that address these vulnerabilities.
There are no workarounds for these vulnerabilities.

This advisory is available at the following link:
http://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20150625-ironport

Cisco Security Vulnerability Policy

To learn about Cisco security vulnerability disclosure policies and publications, see the Security Vulnerability Policy. This document also contains instructions for obtaining fixed software and receiving security vulnerability information from Cisco.

Subscribe to Cisco Security Notifications

Subscribe

Related to This Advisory

Cisco Virtual WSA, ESA, and SMA Default SSH Host Keys Vulnerability

Cisco Virtual WSA, ESA, and SMA Default SSH Host Keys Vulnerability

https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20150625-ironport

Key leaks that got public

News from SEC Consult's experts and 0day research lab.

Wednesday, November 25, 2015

House of Keys: Industry-Wide HTTPS Certificate and SSH Key Reuse Endangers Millions of Devices Worldwide

© seanlockephotography | Fotolia

In the course of an internal research project we have analyzed the firmware images of more than 4000 embedded devices of over 70 vendors. The devices we have looked at include Internet gateways, routers, modems, IP cameras, VoIP phones, etc. We have specifically analyzed cryptographic keys (public keys, private keys, certificates) in firmware images. The most common use of these static keys is:

- SSH Host keys (keys required for operating a SSH server)
- X.509 Certificates used for HTTPS (default server certificate for web based management)

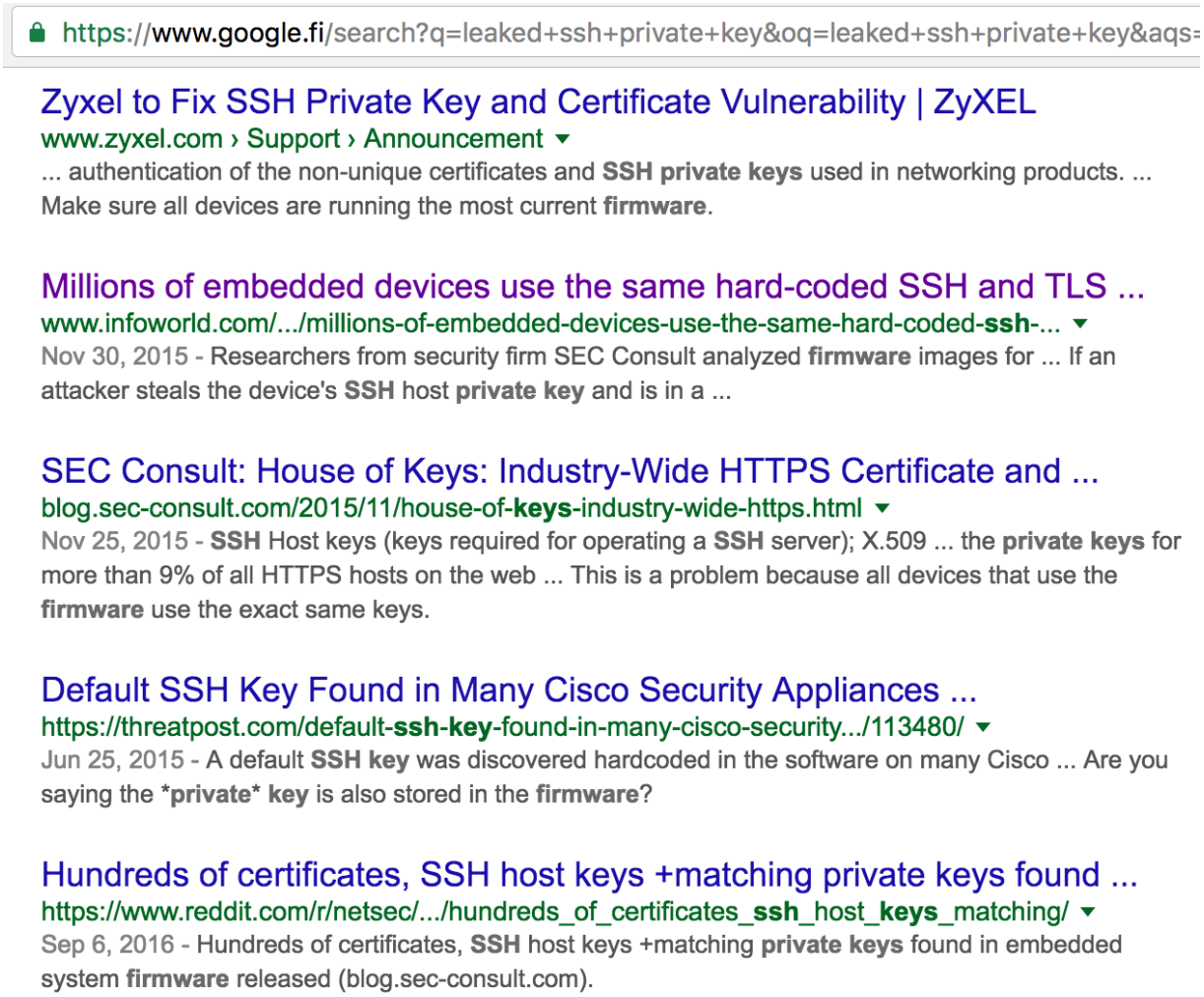
In total we have found more than **580 unique private** keys distributed over all the analysed devices. Correlation via the modulus allows us to find matching certificates.

We have correlated our data with data from Internet-wide scans (Scans.io and Censys.io) and found that our data set (580 unique keys) contains:

http://blog.sec-consult.com/2015/11/house-of-keys-industry-wide-https.html

Key leaks that got public continued

Search for “leaked ssh private key”



Key leaks that got public continued

Search for “leaked ssh private key”

Device Vulnerabilities Fixed: Garrettcom Magnum Series – Network ...
https://blog.qualys.com/.../06/.../device-vulnerabilities-fixed-garrettcom-magnum-seri...
Jun 16, 2015 - The **firmware** also contained hardcoded RSA **private keys** and certificate ... The key are meant to provide **ssh** access to the device without any ...

GitHub - devttys0/littleblackbox: Database of private SSL/SSH keys for ...
https://github.com/devttys0/littleblackbox
littleblackbox - Database of **private** SSL/**SSH keys** for embedded devices. ... Search the database for a given hardware/**firmware** version: $ littleblackbox ...

Embedded Devices Share, Reuse Private SSH Keys, HTTPs Certificates
www.securitynewspaper.com/.../embedded-devices-share-reuse-**private-ssh-keys**-https...
Dec 1, 2015 - Embedded Devices Share, Reuse **Private SSH Keys**, HTTPs Certificates. ... **SSH keys** and X.509 certificates in the **firmware** of more than 4,000 ...

Ruggedcom ROS Hard-Coded RSA SSL Private Key (Update A) | ICS ...
https://ics-cert.us-cert.gov/advisories/ICSA-12-354-01A
Apr 29, 2013 - Ruggedcom ROS Hard-Coded RSA SSL **Private Key** (Update A) ... 3.11 and prior; ROX I OS **firmware** used by RX1000 and RX1100 series products. ... A procedure on how to generate a new **SSH** key can be obtained from ...

D-Link Addresses Non-Unique Certificates and SSH Private Key ...
www.dlink.com/fi/.../industry-wide-issue_devices-using-**ssh**-non-unique-certificates

Key leaks that got public continued

Shodan research on common key fingerprints

Duplicate SSH Keys Everywhere

17 FEBRUARY 2015 on Facets, research, SSH

Back in December when I revamped the SSH banner and started collecting the fingerprint I noticed an odd behavior. It turns out that a few SSH keys are used a lot more than once. For example, the following SSH fingerprint can be found on more than 250,000 devices!

```
dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0
```

And there are many more fingerprints that are also duplicated, which you can check out yourself using the following Python code:

https://blog.shodan.io/duplicate-ssh-keys-everywhere/

Key leaks that got public continued

Difficulty of exploitation is “low skill”

AFFECTED PRODUCTS

Advantech reports that the vulnerability affects the following products:

- EKI-136* product line prior to firmware version 1.27,
- EKI-132* product line prior to firmware version 1.98, and
- EKI-122*-BE product line prior to firmware version 1.65.

IMPACT

An attacker who exploits this vulnerability may be able to intercept communications to and from this device.

Impact to individual organizations depends on many factors that are unique to each organization. NCCIC/ICS-CERT recommends that organizations evaluate the impact of this vulnerability based on their operational environment, architecture, and product implementation.

BACKGROUND

Advantech is based in Taiwan and has distribution offices in 21 countries worldwide.

The EKI-1200 series Modbus gateways are bidirectional gateways for integrating Modbus/RTU and Modbus/ASCII serial devices to TCP/IP networked-based devices. These products are deployed in industrial automation globally.

VULNERABILITY CHARACTERIZATION

VULNERABILITY OVERVIEW

HARD-CODED CREDENTIALS[a]

The firmware contains hard-coded SSH keys that cannot be changed by the user.

CVE-2015-6476[b] has been assigned to this vulnerability. A CVSS v3 base score of 6.5 has been assigned; the CVSS vector string is (CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N).[c]

VULNERABILITY DETAILS

EXPLOITABILITY

This vulnerability could be exploited remotely.

EXISTENCE OF EXPLOIT

No known public exploits specifically target this vulnerability.

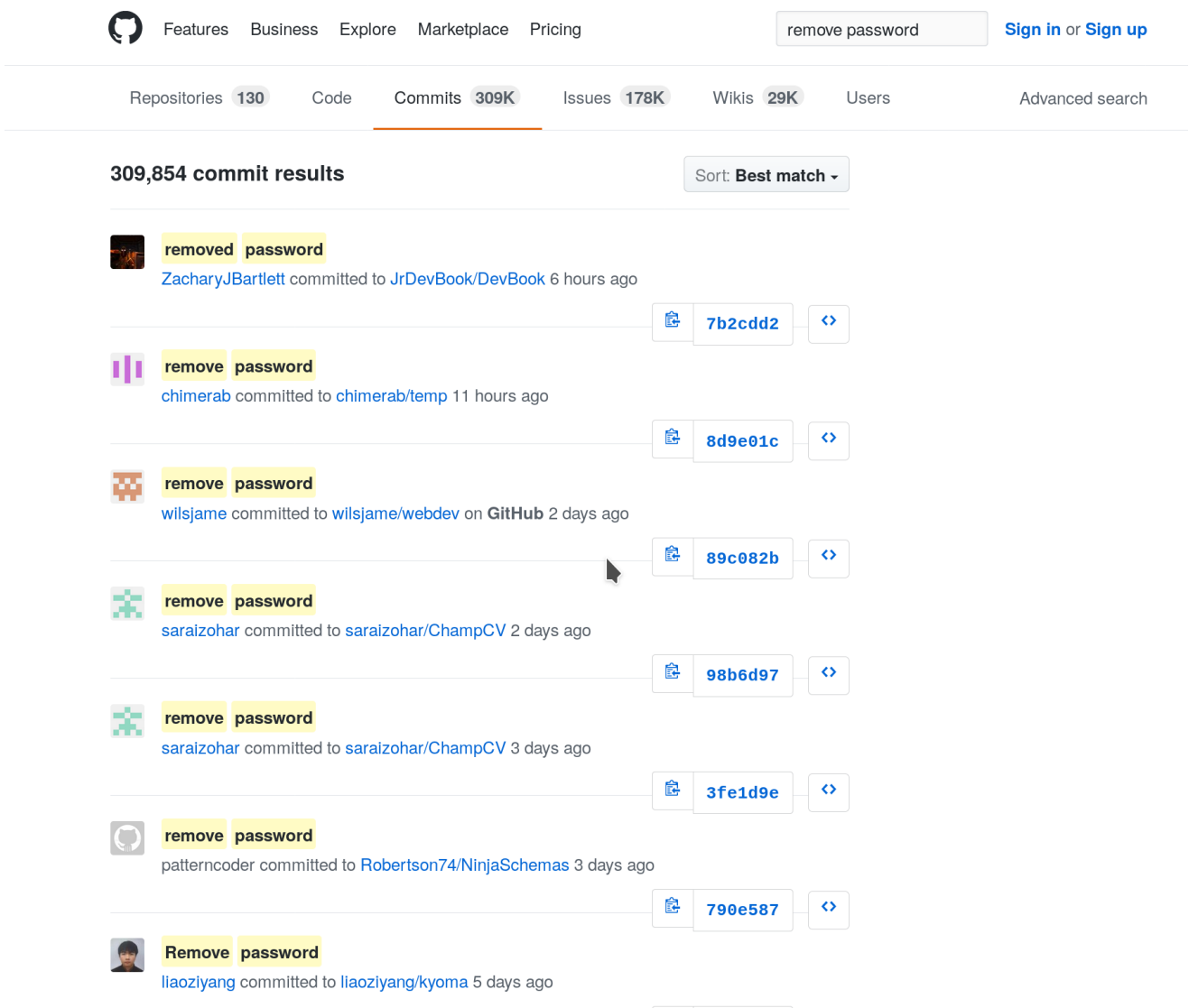
DIFFICULTY

An attacker with a low skill would be able to exploit this vulnerability.

https://ics-cert.us-cert.gov/advisories/ICSA-15-309-01

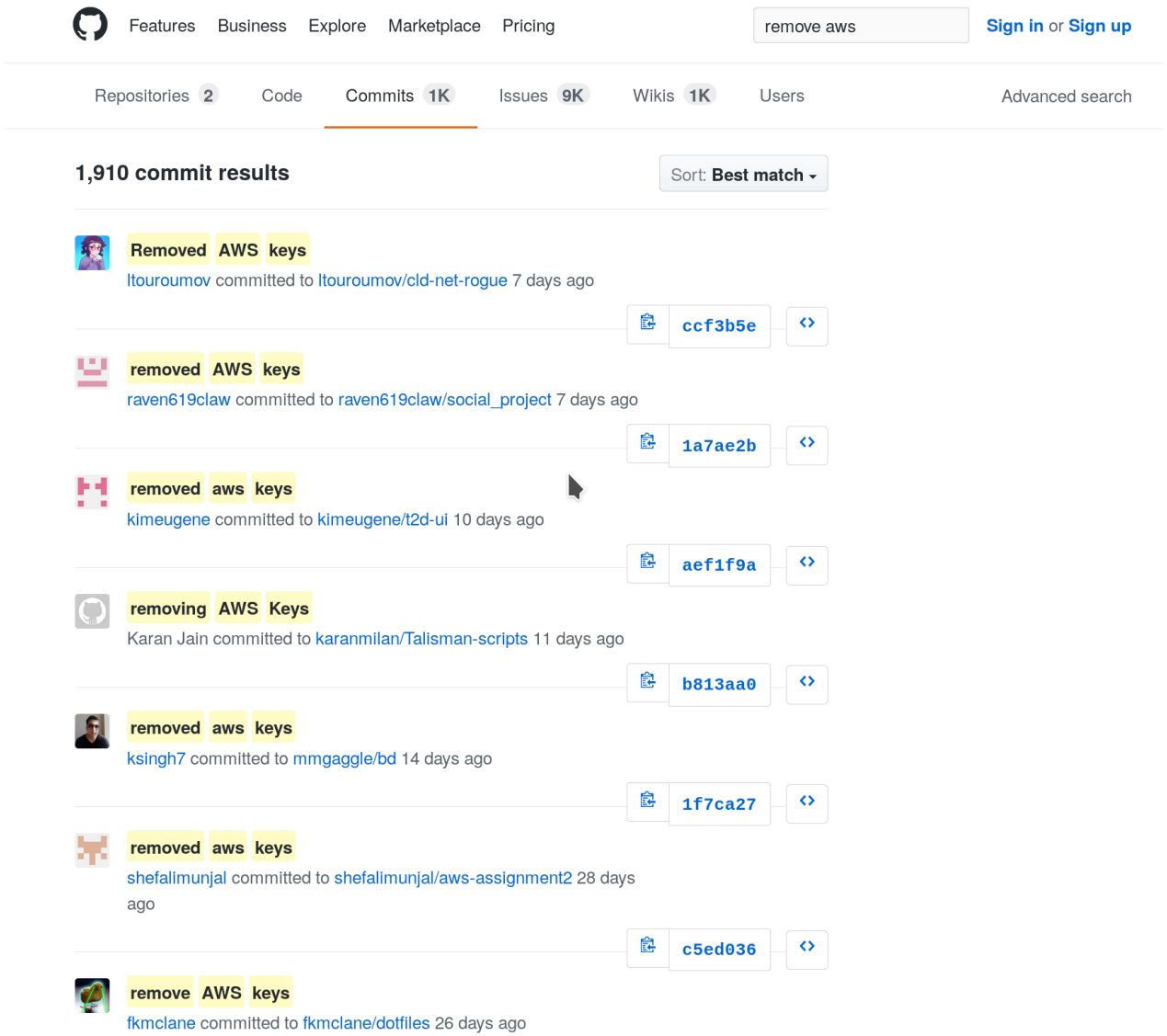
Key leaks that got public continued

Search Github for commit messages



Key leaks that got public continued

Search Github for commit messages



Other people looking for leaks already!

Collecting private keys

littleblackbox

Database of private SSL/SSH keys for embedded devices

House of Keys

Certificates including the matching private key and various private keys found in the firmware of embedded systems. Most certificates are used as HTTPS server certificates. Some private keys are used as SSH host keys. Other uses of these cryptographic keys were not analyzed yet. The names of products that contained the certificates/keys are included as well.

Further information:

- http://blog.sec-consult.com/2015/11/house-of-keys-industry-wide-https.html
- http://blog.sec-consult.com/2016/09/house-of-keys-9-months-later-40-worse.html

SSH Bad Keys

This is a collection of static SSH keys (host and authentication) that have made their way into software and hardware products. This was inspired by the Little Black Box project, but focused primarily on SSH (as opposed to TLS) keys.

Academia did study this, too

A. Costin et al. - A Large-Scale Analysis of the Security of Embedded Firmwares

D. Chen et al. - Towards Fully Automated Dynamic Analysis for Embedded Firmware

Another effective technique for large-scale measurement of embedded device security is network scanning, which avoids direct analysis of firmware images. Using tools such as Nmap, Cui and Stolfo [10] identified approximately 540,000 publicly-accessible embedded devices with *default* access credentials. Over the course of a 4-month longitudinal study, they discovered that less than 3% of access credentials were changed, which suggests that user awareness is lacking. Likewise, using the

Certificates and Private RSA Keys Statistics Many vendors include self-signed certificates inside their firmware images [43, 42]. Due to bad practices in both *release management* and *software design*, some vendors also include the private keys (e.g., PEM, GPG), as confirmed by recent advisories [49, 51].

We developed two simple plugins for our system which collect SSL certificates and private keys. These plugins also collect their fingerprints and check for empty or trivial passphrases. So far, we have been able to extract 109 private RSA keys from 428 firmware images and 56 self-signed SSL certificates out of 344 firmware images. In total, we obtained 41 self-signed SSL certificates together with their corresponding private RSA keys. By looking up those certificates in the public ZMap datasets [36], we were able to automatically locate about 35,000 active online devices.

VI. Related Work

With the increasing prevalence of embedded devices, sever related works have performed large-scale analyses of firmwa images, using a variety of analysis techniques. For exa ple, Heffner[9] performed large-scale extraction of embedde firmware images to gather a database of over 2,000 hardcode SSL private keys. Likewise, Rapid7[10] used a similar analys for hardcoded SSH private keys, albeit on a smaller scale.

Using static analysis, Costin et al. [8] recently analyzed a dataset of approximately 32,000 firmware images. They discovered a total of 38 previously-unknown vulnerabilities including hard-coded back-doors, embedded private key-pairs and XSS vulnerabilities, all of which were obtained "without performing sophisticated static analysis".

5.1 Backdoors in Plain Sight

Many backdoors in embedded systems have been reported recently, ranging from very simple cases [44] to others that were more difficult to discover [50, 64]. In one famous case [44], the backdoor was found to be activated by the string "`xmlset_roodkcableoj28840ybtide`" (i.e., `edit by 04882 joel backdoor` in reverse). This fully functional backdoor was affecting three vendors. Interestingly enough, this backdoor may have been detected earlier by a simple keyword matching on the

What to look for

Currently implemented

- Email addresses
- IP addresses
- URLs
- SSH keys
- AWS secrets

Planned for next versions

- passwords anywhere
- EXIF data, Word documents
- deleted files
- GPG keys, certificates
- DB dumps, keyrings
- JWT, oAuth, . . . tokens
- dotfiles, shell history
- logfiles, XML
- config files (e.g. webservers)
- credit card numbers?
- URL analysis
- known backdoors

What to look for

Currently implemented

- Email addresses
- IP addresses
- URLs
- SSH keys
- AWS secrets

Open for ideas

- ...
- ...
- ...
- ...

Planned for next versions

- passwords anywhere
- EXIF data, Word documents
- deleted files
- GPG keys, certificates
- DB dumps, keyrings
- JWT, oAuth, ... tokens
- dotfiles, shell history
- logfiles, XML
- config files (e.g. webservers)
- credit card numbers?
- URL analysis
- known backdoors

Mass-scan study

Made a wide scan of stuff from the internet:

- 1k Github repositories
- 1.3k Docker containers
- 44k Firmware images (routers, modems, . . .)
- 100k various files (customer uploaded scans)
- some Android apps
- some Linux based appliance images

. . . and more to come soon!

Mass-scan study results

Statistics of what was found so far:

- 647k files with SSH private keys = 4k unique keys (~200 = 5% found on the net!)
- 57k files with AWS secrets = 3k unique secrets (~300 = 10% valid!)
- 1920k unique URLs
- 86k unique IP addresses
- 104k unique Email addresses

Mass scan cherry picks!

The results contain of course a lot of non-usable stuff like test-keys, but looking at the paths . . . some examples

Private keys

- /root/.ssh/id_rsa
- . . . paypal/keys/privatekey.pem
- . . . keys/private/corporate.pem
- . . . bank_integration/private.pem
- . . . keys/production.pem

Mass scan cherry picks!

The results contain of course a lot of non-usable stuff like test-keys, but looking at the paths . . . some examples

AWS Secrects

- . . . container/configs/aws_prod.yml
- . . . config/app/production.json
- . . . config/aws-pre-production.json
- . . . config/secrets/staging.yml
- . . . usr/app/Makefile
- . . . app/config/secrets.yml
- golang-src-1.4.2-9.el7.noarch.rpm ?!

Mitigation

- no silver bullet due to “all those places”
- improve development practice and awareness
- separation of systems, keys (in case of compromise)
- use tools for avoiding leaks (credentials vaults/password safes)
- use tools for finding leaks (glorified `grep`!)

Conclusion

- there is a lot of low hanging fruit for attackers
- problem awareness is still low but growing
- issue will likely become worse due to modern DevOps

Demonstration time

Demonstrate some of the findings!