Do security certification schemes improve security?

Analyzing artifacts of Common Criteria & FIPS 140 certifications

EurOpen 2022

Adam Janovsky | `adamjanovsky@mail.muni.cz`

MUNI
FI

Analyzing artifacts of Common Criteria & FIPS 140 certifications

Do security certification schemes improve security?

EurOpen 2022

Adam Janovsky | `adamjanovsky@mail.muni.cz`

About me

- PhD candidate @ FI MUNI, Brno
- Centre for Research on Cryptography and Security
- Part-time researcher at Invasys, Brno

About me

- PhD candidate @ FI MUNI, Brno
- Centre for Research on Cryptography and Security
- Part-time researcher at Invasys, Brno
- Mining data about cryptography usage from large datasets
  - 150+ million of RSA keys
  - 600+ thousand of Android APKs
  - Now 30+ thousand pdf documents related to certifications

About me

- PhD candidate @ FI MUNI, Brno
- Centre for Research on Cryptography and Security
- Part-time researcher at Invasys, Brno
- Mining data about cryptography usage from large datasets
  - 150+ million of RSA keys
  - 600+ thousand of Android APKs
  - Now 30+ thousand pdf documents related to certifications

📤 Slideshow available from: ajanovsky.cz/europen.pdf

Joint work 👏

Petr Svenda, Jan Jancar, Jiri Michalik, Stanislav Bobon, Adam Janovsky, Vashek Matyas, and others…

Work in progress 🔧

2017 at CRoCS

- Return of the Coppersmith Attack: Practical Factorization of Widely Used RSA Moduli
- CVE-2017-15361

2017 at CRoCS

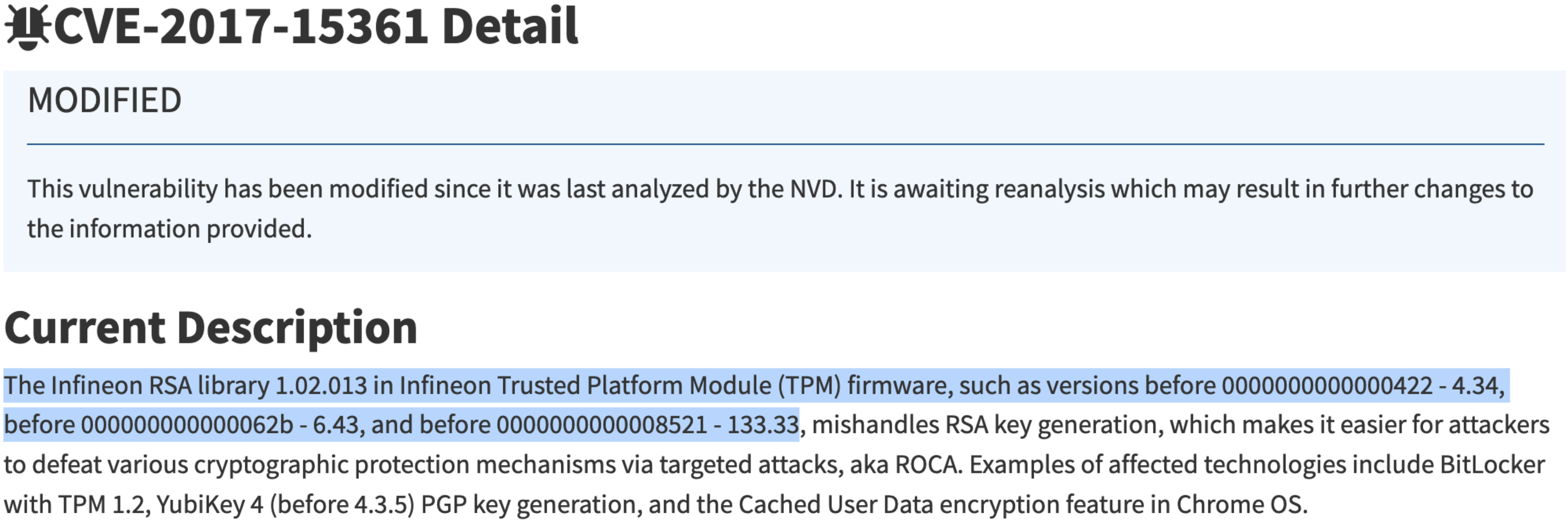
- Return of the Coppersmith Attack: Practical Factorization of Widely Used RSA Moduli
- CVE-2017-15361

2017 at CRoCS

- Return of the Coppersmith Attack: Practical Factorization of Widely Used RSA Moduli
- CVE-2017-15361

Which certified devices are affected?

Finding affected certified devices…



CVE-2017-15361 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Current Description

The Infineon RSA library 1.02.013 in Infineon Trusted Platform Module (TPM) firmware, such as versions before 0000000000000422 - 4.34, before 000000000000062b - 6.43, and before 0000000000008521 - 133.33, mishandles RSA key generation, which makes it easier for attackers to defeat various cryptographic protection mechanisms via targeted attacks, aka ROCA. Examples of affected technologies include BitLocker with TPM 1.2, YubiKey 4 (before 4.3.5) PGP key generation, and the Cached User Data encryption feature in Chrome OS.

Excerpt from RoCA NVD record.

Finding affected certified devices...

Finding affected certified devices…

- The affected "thing" is *The Infineon RSA library 1.02.013 in Infineon Trusted Platform Module (TPM) firmware*

Finding affected certified devices...

- The affected "thing" is *The Infineon RSA library 1.02.013 in Infineon Trusted Platform Module (TPM) firmware*
- Browsing through all Common Criteria certificates you notice `BSI-DSZ-CC-0907-2013`

```
Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010
or v1.02.013, EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 libraries and with
specific IC-dedicated software.
```

Finding affected certified devices...

- The affected "thing" is *The Infineon RSA library 1.02.013 in Infineon Trusted Platform Module (TPM) firmware*
- Browsing through all Common Criteria certificates you notice `` `BSI-DSZ-CC-0907-2013` ``

```
Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010
or v1.02.013, EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 libraries and with
specific IC-dedicated software.
```

What if this product is used in other certificated products?

So you browse all CC certificates again

So you browse all CC certificates again

- And you find a fairly similar certificate `` `BSI-DSZ-CC-0926-2014` ``

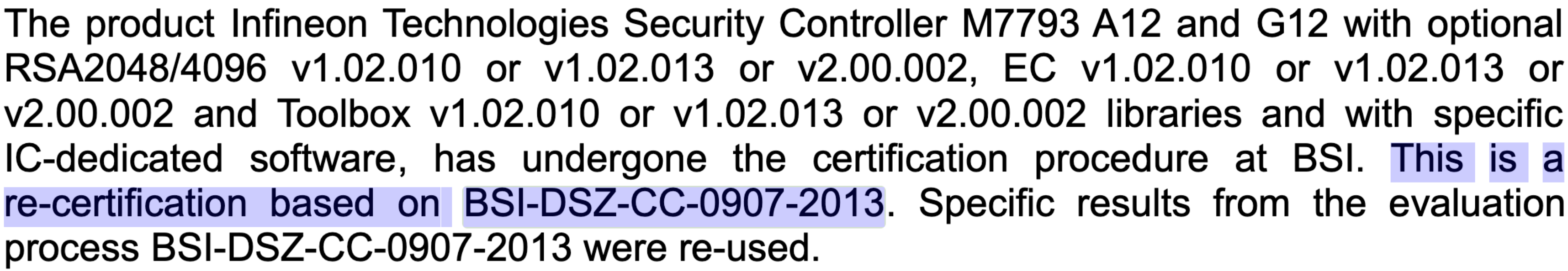
```
Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010 or v1.02.013
or v2.00.002, EC v1.02.010 or v1.02.013 or v2.00.002 and Toolbox v1.02.010 or v1.02.013 or v2.00.002 libraries
and with specific IC-dedicated software
```

So you browse all CC certificates again

- And you find a fairly similar certificate `` `BSI-DSZ-CC-0926-2014` ``

```
Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010 or v1.02.013
or v2.00.002, EC v1.02.010 or v1.02.013 or v2.00.002 and Toolbox v1.02.010 or v1.02.013 or v2.00.002 libraries
and with specific IC-dedicated software
```

- You open its certification report and find



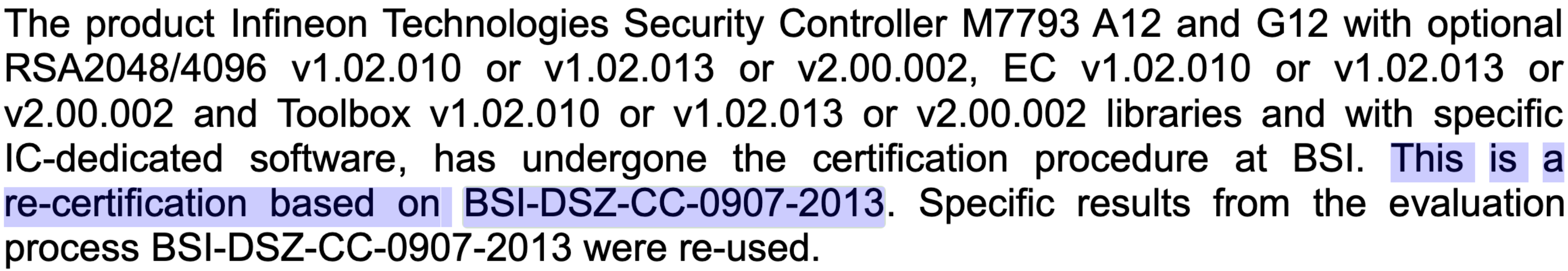
The product Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010 or v1.02.013 or v2.00.002, EC v1.02.010 or v1.02.013 or v2.00.002 and Toolbox v1.02.010 or v1.02.013 or v2.00.002 libraries and with specific IC-dedicated software, has undergone the certification procedure at BSI. This is a re-certification based on BSI-DSZ-CC-0907-2013. Specific results from the evaluation process BSI-DSZ-CC-0907-2013 were re-used.

So you browse all CC certificates again

- And you find a fairly similar certificate `` `BSI-DSZ-CC-0926-2014` ``

```
Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010 or v1.02.013
or v2.00.002, EC v1.02.010 or v1.02.013 or v2.00.002 and Toolbox v1.02.010 or v1.02.013 or v2.00.002 libraries
and with specific IC-dedicated software
```

- You open its certification report and find



The product Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010 or v1.02.013 or v2.00.002, EC v1.02.010 or v1.02.013 or v2.00.002 and Toolbox v1.02.010 or v1.02.013 or v2.00.002 libraries and with specific IC-dedicated software, has undergone the certification procedure at BSI. This is a re-certification based on BSI-DSZ-CC-0907-2013. Specific results from the evaluation process BSI-DSZ-CC-0907-2013 were re-used.

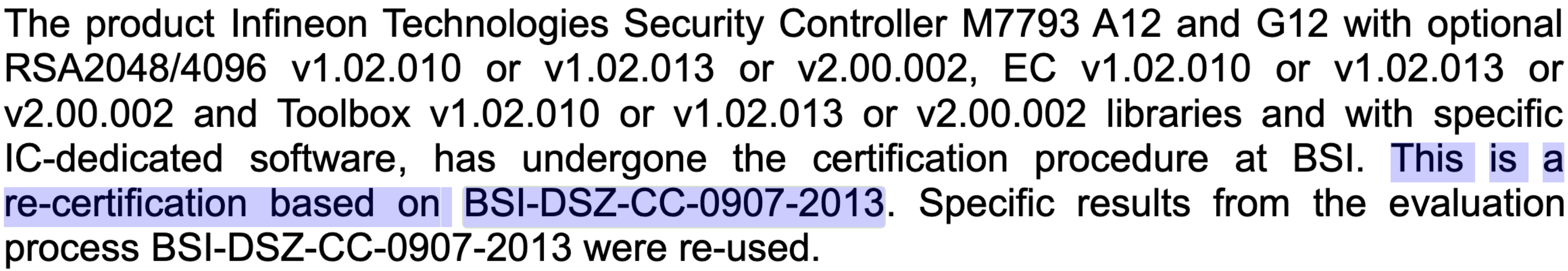
But what if this product is used in some certified product?

So you browse all CC certificates again

- And you find a fairly similar certificate `BSI-DSZ-CC-0926-2014`

```
Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010 or v1.02.013
or v2.00.002, EC v1.02.010 or v1.02.013 or v2.00.002 and Toolbox v1.02.010 or v1.02.013 or v2.00.002 libraries
and with specific IC-dedicated software
```

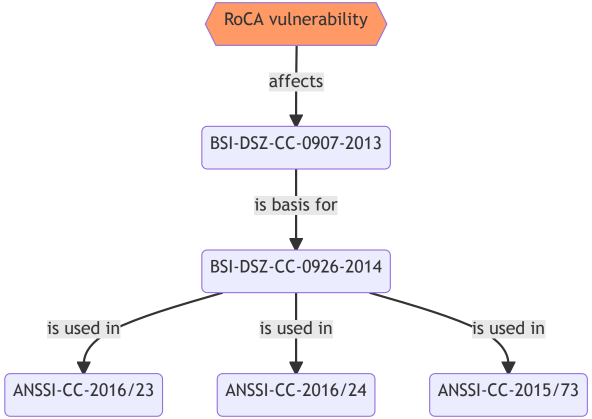
- You open its certification report and find



The product Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010 or v1.02.013 or v2.00.002, EC v1.02.010 or v1.02.013 or v2.00.002 and Toolbox v1.02.010 or v1.02.013 or v2.00.002 libraries and with specific IC-dedicated software, has undergone the certification procedure at BSI. This is a re-certification based on BSI-DSZ-CC-0907-2013. Specific results from the evaluation process BSI-DSZ-CC-0907-2013 were re-used.

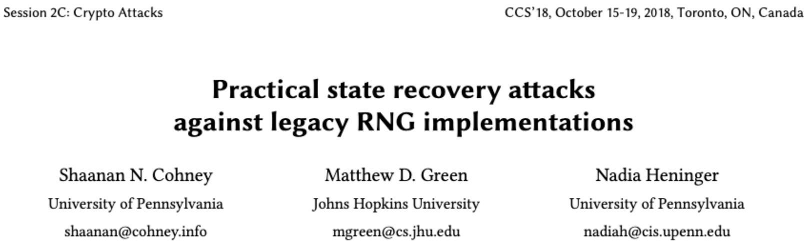
But what if this product is used in some certified product?

- Spoiler, it is: `ANSSI-CC-2016/23`, `ANSSI-CC-2016/24`, `ANSSI-CC-2015/73`


RoCA vulnerability
affects
BSI-DSZ-CC-0907-2013
is basis for
BSI-DSZ-CC-0926-2014
is used in
is used in
is used in
ANSSI-CC-2016/23
ANSSI-CC-2016/24
ANSSI-CC-2015/73

This pattern repeats

- State of `ANSI X9.17/X9.31` PRNG can be recovered if its internal key is not secret.



Session 2C: Crypto Attacks

CCS'18, October 15-19, 2018, Toronto, ON, Canada

Practical state recovery attacks against legacy RNG implementations

Shaanan N. Cohney
University of Pennsylvania
shaanan@cohney.info

Matthew D. Green
Johns Hopkins University
mgreen@cs.jhu.edu

Nadia Heninger
University of Pennsylvania
nadiah@cis.upenn.edu

This pattern repeats

- State of `ANSI X9.17/X9.31` PRNG can be recovered if its internal key is not secret.
- But FIPS 140-2 algorithm doesn't require that

Session 2C: Crypto Attacks CCS'18, October 15-19, 2018, Toronto, ON, Canada

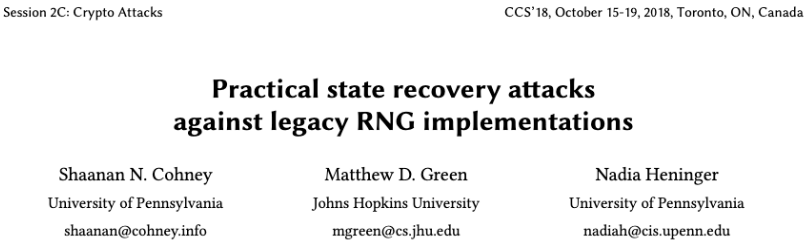
Practical state recovery attacks against legacy RNG implementations

Shaanan N. Cohney
University of Pennsylvania
shaanan@cohney.info

Matthew D. Green
Johns Hopkins University
mgreen@cs.jhu.edu

Nadia Heninger
University of Pennsylvania
nadiah@cis.upenn.edu

This pattern repeats



Session 2C: Crypto Attacks

CCS'18, October 15-19, 2018, Toronto, ON, Canada

Practical state recovery attacks against legacy RNG implementations

Shaanan N. Cohney
University of Pennsylvania
shaanan@cohney.info

Matthew D. Green
Johns Hopkins University
mgreen@cs.jhu.edu

Nadia Heninger
University of Pennsylvania
nadiah@cis.upenn.edu

- State of `ANSI X9.17/X9.31` PRNG can be recovered if its internal key is not secret.
- But FIPS 140-2 algorithm doesn't require that

So, is there a (certified) product with the insecure configuration?

This pattern repeats

Session 2C: Crypto Attacks CCS'18, October 15-19, 2018, Toronto, ON, Canada

Practical state recovery attacks against legacy RNG implementations

Shaanan N. Cohney
University of Pennsylvania
shaanan@cohney.info

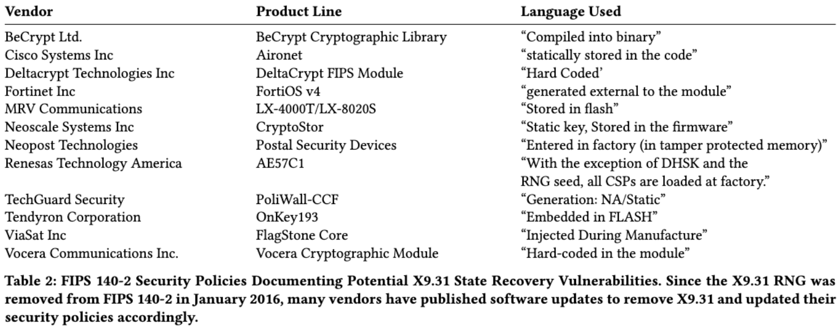
Matthew D. Green
Johns Hopkins University
mgreen@cs.jhu.edu

Nadia Heninger
University of Pennsylvania
nadiah@cis.upenn.edu

- State of `` `ANSI X9.17/X9.31` `` PRNG can be recovered if its internal key is not secret.
- But FIPS 140-2 algorithm doesn't require that

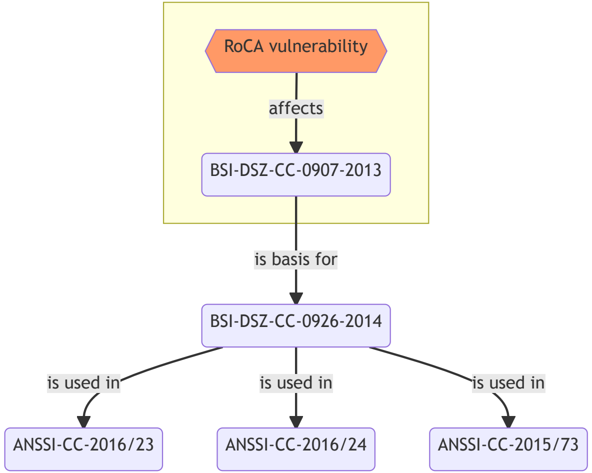
So, is there a (certified) product with the insecure configuration?

- After reviewing 1411 certificates, the authors conclude that

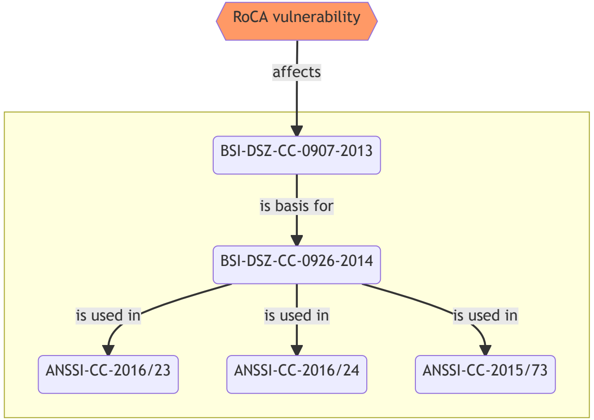


Vendor	Product Line	Language Used
BeCrypt Ltd.	BeCrypt Cryptographic Library	"Compiled into binary"
Cisco Systems Inc	Aironet	"statically stored in the code"
Deltacrypt Technologies Inc	DeltaCrypt FIPS Module	"Hard Coded'
Fortinet Inc	FortiOS v4	"generated external to the module"
MRV Communications	LX-4000T/LX-8020S	"Stored in flash"
Neoscale Systems Inc	CryptoStor	"Static key, Stored in the firmware"
Neopost Technologies	Postal Security Devices	"Entered in factory (in tamper protected memory)"
Renesas Technology America	AE57C1	"With the exception of DHSK and the RNG seed, all CSPs are loaded at factory."
TechGuard Security	PoliWall-CCF	"Generation: NA/Static"
Tendyron Corporation	OnKey193	"Embedded in FLASH"
ViaSat Inc	FlagStone Core	"Injected During Manufacture"
Vocera Communications Inc.	Vocera Cryptographic Module	"Hard-coded in the module"

Table 2: FIPS 140-2 Security Policies Documenting Potential X9.31 State Recovery Vulnerabilities. Since the X9.31 RNG was removed from FIPS 140-2 in January 2016, many vendors have published software updates to remove X9.31 and updated their security policies accordingly.



RoCA vulnerability
affects
BSI-DSZ-CC-0907-2013
is basis for
BSI-DSZ-CC-0926-2014
is used in
is used in
is used in
ANSSI-CC-2016/23
ANSSI-CC-2016/24
ANSSI-CC-2015/73


RoCA vulnerability
affects
BSI-DSZ-CC-0907-2013
is basis for
BSI-DSZ-CC-0926-2014
is used in
is used in
is used in
ANSSI-CC-2016/23
ANSSI-CC-2016/24
ANSSI-CC-2015/73

Common Criteria 101

Common Criteria 101

1. Product vendor makes some claims about *Target of evaluation* (product, or even parts of it)
    - Security functional components - security functions provided by the product
    - Security assurance components - measures taken to protect the product

Common Criteria 101

1. Product vendor makes some claims about *Target of evaluation* (product, or even parts of it)
    - Security functional components - security functions provided by the product
    - Security assurance components - measures taken to protect the product
2. *Security target* constructed from scratch or from *protection profile*

Common Criteria 101

1. Product vendor makes some claims about *Target of evaluation* (product, or even parts of it)
   - Security functional components - security functions provided by the product
   - Security assurance components - measures taken to protect the product
2. *Security target* constructed from scratch or from *protection profile*
3. Evaluation assurance level (EAL) specifies what and to what extent must be validated

Common Criteria 101

1. Product vendor makes some claims about *Target of evaluation* (product, or even parts of it)
   - Security functional components - security functions provided by the product
   - Security assurance components - measures taken to protect the product
2. *Security target* constructed from scratch or from *protection profile*
3. Evaluation assurance level (EAL) specifies what and to what extent must be validated
4. Independent laboratory validates the claims
   - expensive, time-demanding

Common Criteria 101

1. Product vendor makes some claims about *Target of evaluation* (product, or even parts of it)
   - Security functional components - security functions provided by the product
   - Security assurance components - measures taken to protect the product
2. *Security target* constructed from scratch or from *protection profile*
3. Evaluation assurance level (EAL) specifies what and to what extent must be validated
4. Independent laboratory validates the claims
   - expensive, time-demanding
5. 📄 You receive the certification and sell your stuff to governments 💰💰💰

Common Criteria 101

1. Product vendor makes some claims about *Target of evaluation* (product, or even parts of it)
   - Security functional components - security functions provided by the product
   - Security assurance components - measures taken to protect the product
2. *Security target* constructed from scratch or from *protection profile*
3. Evaluation assurance level (EAL) specifies what and to what extent must be validated
4. Independent laboratory validates the claims
   - expensive, time-demanding
5. 📄 You receive the certification and sell your stuff to governments 💰💰💰

A certificate is valid for ~3-6 years and can be partially updated with maintenance updates.

What are the problems?

What are the problems?

- Certification artifacts are written by people for people.

What are the problems?

- Certification artifacts are written by people for people.
- The artifacts are pdf files; with typos and not always in English.

What are the problems?

- Certification artifacts are written by people for people.
- The artifacts are pdf files; with typos and not always in English.
- There's no unique naming (or IDs) of the certified devices.
    - Difficult linking to vulnerability databases.
    - Difficult to build reference graphs.

What are the problems?

- Certification artifacts are written by people for people.
- The artifacts are pdf files; with typos and not always in English.
- There's no unique naming (or IDs) of the certified devices.
  - Difficult linking to vulnerability databases.
  - Difficult to build reference graphs.

They are not meant to be processed automatically.

Can we do better? 🤷‍♂️

Our approach

Our approach

- Don’t try to change CC or FIPS 140 schemes.

Our approach

- Don’t try to change CC or FIPS 140 schemes.
- Build a tool capable of robust processing of certification artifacts.

Our approach

- Don’t try to change CC or FIPS 140 schemes.
- Build a tool capable of robust processing of certification artifacts.
- Automate all that we can.

Our approach

- Don’t try to change CC or FIPS 140 schemes.
- Build a tool capable of robust processing of certification artifacts.
- Automate all that we can.
- Build a frontend for it.

Our approach

- Don’t try to change CC or FIPS 140 schemes.
- Build a tool capable of robust processing of certification artifacts.
- Automate all that we can.
- Build a frontend for it.
- Analyze interesting trends and write a paper about those.

CC certification artifacts

- The ECDH (ECC Diffie-Hellman) key exchange algorithm can be used to establish cryptographic keys. It can be also used as secure point multiplication.
- Provide secure point addition for Elliptic Curves over GF(p).

The TOE supports various key sizes for ECC over GF(p) up to a limit of 576 bits for signature generation, key pair generation and key exchange. For signature verification the TOE supports key sizes up to a limit of 576 bits . To fend off attackers with high attack potential an adequate key length must be used (references can be found in national and international documents and standards).

SHA

- The SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512 algorithms can be used for different purposes such as computing hash values in the course of digital signature creation or key derivation.

To fend off attackers with high attack potential an adequate security level must be used (references can be found in national and international documents and standards). In particular this means that SHA-1 shall not be used.

Resistance of cryptographic algorithms against attacks

The cryptographic algorithms are resistant against attacks as described in JIL, Attack Methods for Smartcards and Similar Devices [32], which include Side Channel Attacks, Perturbation attacks, Differential Fault Analysis (DFA) and timing attacks, except for SHA, which is only resistant against Side Channel Attacks and timing attacks.

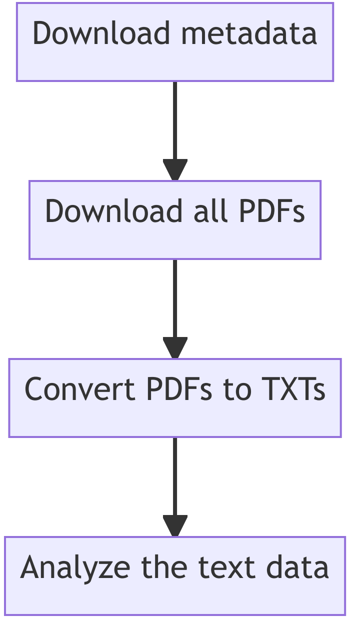
More details about conditions and restrictions for resistance against attacks are given in the user documentation of the Crypto Library [11][12].

Random number generation

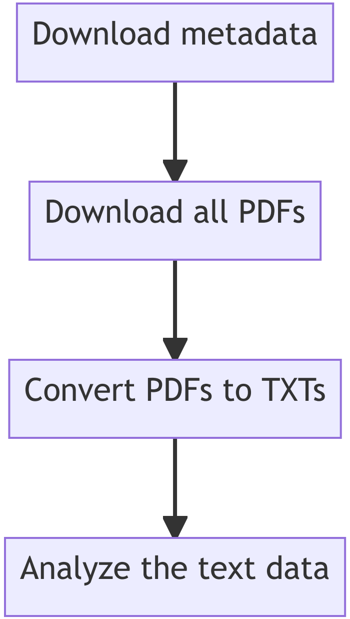
- The TOE provides access to random numbers generated by a software (pseudo) random number generator and functions to perform a test of the hardware (true) random number generator at initialisation.

Processing pipeline

Processing pipeline



Processing pipeline



Download metadata

Download all PDFs

Convert PDFs to TXTs

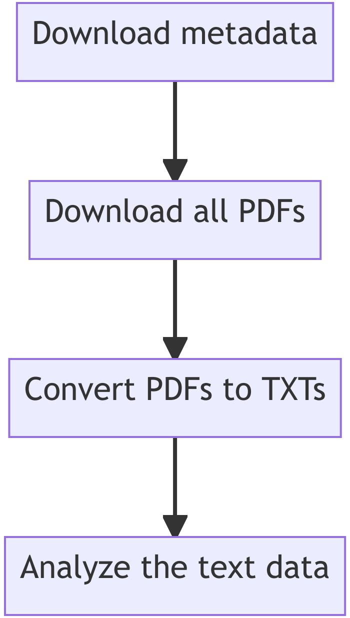
Analyze the text data

Run the pipeline

```
from sec_certs.dataset import CCDataset

dset = CCDataset()
dset.get_certs_from_web()
dset.process_protection_profiles()
dset.download_all_pdfs()
dset.convert_all_pdfs()
dset.analyze_certificates()
```

Processing pipeline



Run the pipeline

```
from sec_certs.dataset import CCDataset

dset = CCDataset()
dset.get_certs_from_web()
dset.process_protection_profiles()
dset.download_all_pdfs()
dset.convert_all_pdfs()
dset.analyze_certificates()
```

View the results

```
print(dset.to_pandas().head(n=3))
```

	cert_id	name	status	category	manufacturer	scheme	security_level
dgst							
8298c7814b3b2860	KECS-CR-22-11	KSignAccess V4.1	active	Access Control Devices and Systems	KSign Co., LTD.	KR	{}
9a1c767d358eee50	DXC-EFC-T092-ETR 1.0	VeroGuard HSM Digital ID for Open Networks v1.0	active	Access Control Devices and Systems	VeroGuard Systems Pty Ltd	AU	{ALC_FLR.1, EAL2+}
979e00ac7d3e229c	KECS-CR-21-63	SafeIdentity v5.1	active	Access Control Devices and Systems	Hancom With Inc.	KR	{}

Data extraction

Data extraction

1. Read all text files and match regular expressions

```
`"AES[-]*(?:128|192|256|)"`
```

Data extraction

1. Read all text files and match regular expressions

```
`"AES[-]*(?:128|192|256|)"`
```

2. Clean the matched expressions

Data extraction

1. Read all text files and match regular expressions

```
`"AES[-]*(?:128|192|256|)"`
```

2. Clean the matched expressions
3. Correlate with external database

- National Vulnerability Database
- 🤝 Plug in your data source here

Data extraction

1. Read all text files and match regular expressions

```
`"AES[-]*(?:128|192|256|)"`
```

2. Clean the matched expressions
3. Correlate with external database

- National Vulnerability Database
- 🤝 Plug in your data source here

Extracted data

- Certification IDs
- Security assurance requirements
- Security functional components
- References to other standards (FIPS, ISO/IEC, …)
- Security levels
- Javacard platform, API constants
- Cryptographic algorithms
- Utilized elliptic curves
- Cryptographic libraries
- Defenses
- Vulnerabilities

Data serialization

Data serialization

JSON or Pandas

```
from sec_certs.sample import CommonCriteriaCert
certificate = dset["5efe98a1ba4df4d7"]
certificate.to_json("./certificate.json")
other = CommonCriteriaCert.from_json("./certificate.json")
```

Data serialization

JSON or Pandas

```
from sec_certs.sample import CommonCriteriaCert
certificate = dset["5efe98a1ba4df4d7"]
certificate.to_json("./certificate.json")
other = CommonCriteriaCert.from_json("./certificate.json")
```

```
"report_references": {
  "directly_referenced_by": ["BSI-DSZ-CC-0926-2014"],
  "directly_referencing": ["BSI-DSZ-CC-0757-2011"],
  "indirectly_referenced_by": [
      "BSI-DSZ-CC-0926-2014",
      "BSI-DSZ-CC-0926-V2-2017"
  ],
  "indirectly_referencing": ["BSI-DSZ-CC-0757-2011"]
}
```

Data serialization

JSON or Pandas

```
from sec_certs.sample import CommonCriteriaCert
certificate = dset["5efe98a1ba4df4d7"]
certificate.to_json("./certificate.json")
other = CommonCriteriaCert.from_json("./certificate.json")
```

```
"report_references": {
  "directly_referenced_by": ["BSI-DSZ-CC-0926-2014"],
  "directly_referencing": ["BSI-DSZ-CC-0757-2011"],
  "indirectly_referenced_by": [
      "BSI-DSZ-CC-0926-2014",
      "BSI-DSZ-CC-0926-V2-2017"
  ],
  "indirectly_referencing": ["BSI-DSZ-CC-0757-2011"]
}
```

```
  "heuristics": {
    "cert_id": "BSI-DSZ-CC-0907-2013",
    "cert_lab": ["BSI"],
    "cpe_matches": ["cpe:2.3:a:infineon:rsa_library:1.02.01
    "related_cves": ["CVE-2017-15361"],
    …
```

Data serialization

JSON or Pandas

```
from sec_certs.sample import CommonCriteriaCert
certificate = dset["5efe98a1ba4df4d7"]
certificate.to_json("./certificate.json")
other = CommonCriteriaCert.from_json("./certificate.json")
```

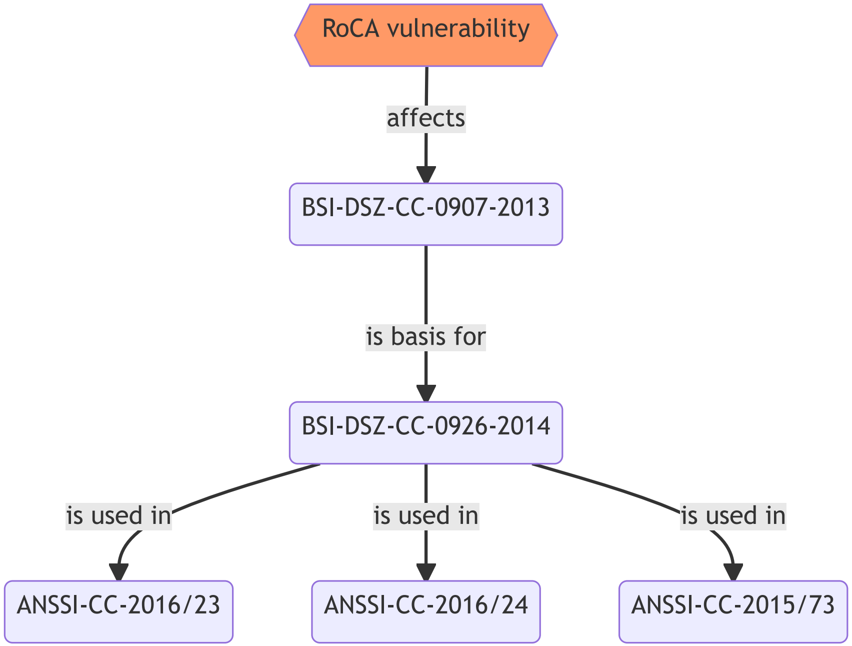
```
"report_references": {
  "directly_referenced_by": ["BSI-DSZ-CC-0926-2014"],
  "directly_referencing": ["BSI-DSZ-CC-0757-2011"],
  "indirectly_referenced_by": [
      "BSI-DSZ-CC-0926-2014",
      "BSI-DSZ-CC-0926-V2-2017"
  ],
  "indirectly_referencing": ["BSI-DSZ-CC-0757-2011"]
}
```

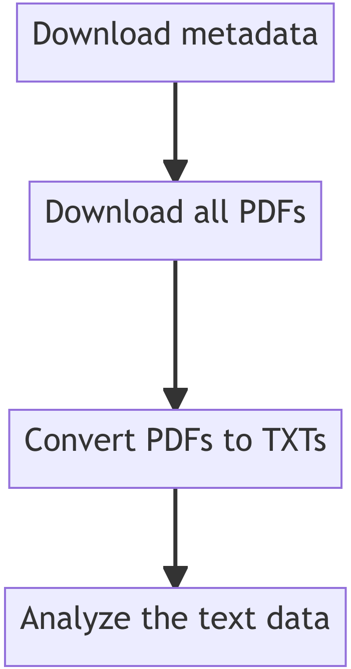
```
  "heuristics": {
    "cert_id": "BSI-DSZ-CC-0907-2013",
    "cert_lab": ["BSI"],
    "cpe_matches": ["cpe:2.3:a:infineon:rsa_library:1.02.01
    "related_cves": ["CVE-2017-15361"],
    ...
```

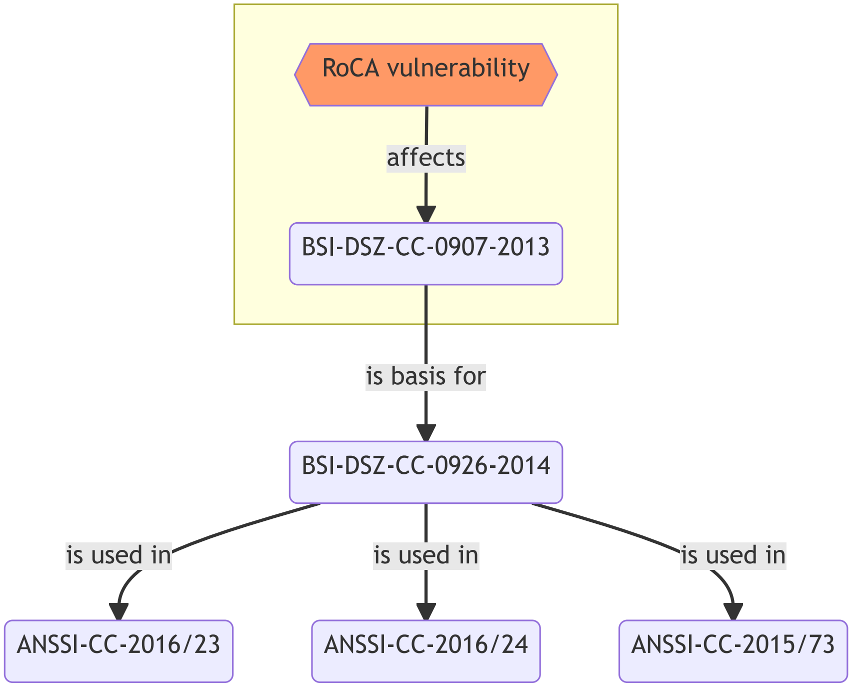
PDF data

```
"rules_crypto_algs": {
  "ECDSA": 7,
  "RNG": 10,
  "RSA-2048": 1,
  "RSA2048": 2,
  "RSA4096": 2,
  "TDES": 2,
  "TRNG": 4},
"rules_crypto_libs": {"v1.02.013": 28},
"rules_defenses": {
  "DFA": 5,
  "DPA": 6,
  "SPA": 5,
  "physical probing": 1,
  "physical tampering": 1},
"rules_ecc_curves": {"P-192": 2},
"rules_standard_id": {
  "AIS31": 3,
  "FIPS PUB 197": 1,
  "RFC 5639": 1,
  "RFC5639": 1},
"rules_vendor": {"Infineon": 40}
```

Linking certified products to vulnerabilities


RoCA vulnerability
affects
BSI-DSZ-CC-0907-2013
is basis for
BSI-DSZ-CC-0926-2014
is used in
is used in
is used in
ANSSI-CC-2016/23
ANSSI-CC-2016/24
ANSSI-CC-2015/73


Download metadata
Download all PDFs
Convert PDFs to TXTs
Analyze the text data



RoCA vulnerability

affects

BSI-DSZ-CC-0907-2013

is basis for

BSI-DSZ-CC-0926-2014

is used in

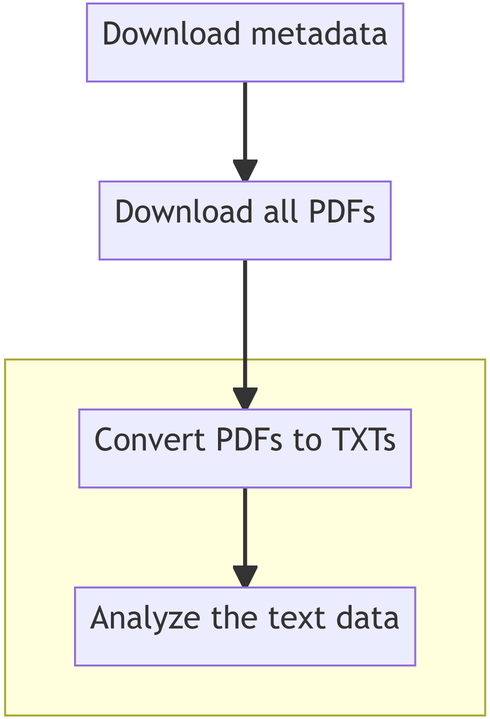
ANSSI-CC-2016/23

is used in

ANSSI-CC-2016/24

is used in

ANSSI-CC-2015/73



Download metadata

Download all PDFs

Convert PDFs to TXTs

Analyze the text data

Linking certified products to vulnerabilities

Linking certified products to vulnerabilities

- For every certified device, we have `` `(vendor, device name, heuristically extracted versions)` ``

Linking certified products to vulnerabilities

- For every certified device, we have `` `(vendor, device name, heuristically extracted versions)` ``

> Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010 or v1.02.013 EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 libraries and with specific IC-dedicated software

Linking certified products to vulnerabilities

- For every certified device, we have `` `(vendor, device name, heuristically extracted versions)` ``

> Infineon Technologies Security Controller M7793 A12 and G12 with optional RSA2048/4096 v1.02.010 or v1.02.013 EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 libraries and with specific IC-dedicated software

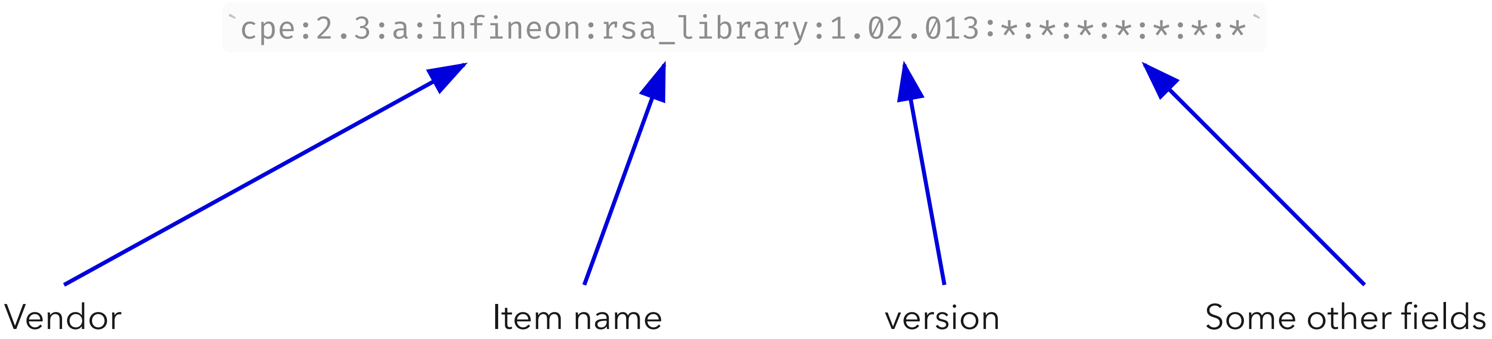
- Each vulnerability has a list of affected platform configurations specified with CPEs

What the heck is a CPE?

What the heck is a CPE?

```
`cpe:2.3:a:infineon:rsa_library:1.02.013:*:*:*:*:*:*:*`
```

What the heck is a CPE?



`cpe:2.3:a:infineon:rsa_library:1.02.013:*:*:*:*:*:*:*`

Vendor

Item name

version

Some other fields

Linking certified products to vulnerabilities

- For every certified device, we have `` `(vendor, device name, heuristically extracted versions)` ``

> **Infineon Technologies** Security Controller M7793 A12 and G12 with optional **RSA**2048/4096 v1.02.010 or **v1.02.013**, EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 **libraries** and with specific IC-dedicated software

- Each vulnerability has a list of affected platforms specified with CPE
- RoCA vulnerability has, among others: `` `cpe:2.3:a:infineon:rsa_library:1.02.013:*:*:*:*:*:*:*` ``

Linking certified products to vulnerabilities

- For every certified device, we have `` `(vendor, device name, heuristically extracted versions)` ``

> **Infineon Technologies** Security Controller M7793 A12 and G12 with optional **RSA**2048/4096 v1.02.010 or **v1.02.013**, EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 **libraries** and with specific IC-dedicated software

- Each vulnerability has a list of affected platforms specified with CPE
- RoCA vulnerability has, among others: `` `cpe:2.3:a:infineon:rsa_library:1.02.013:*:*:*:*:*:*:*` ``

💡 **Idea: Measure string similarity between certificate name and CPEs**

Linking certified products to vulnerabilities

- For every certified device, we have `(vendor, device name, heuristically extracted versions)`

> **Infineon Technologies** Security Controller M7793 A12 and G12 with optional **RSA**2048/4096 v1.02.010 or **v1.02.013**, EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 **libraries** and with specific IC-dedicated software

- Each vulnerability has a list of affected platforms specified with CPE
- RoCA vulnerability has, among others: `cpe:2.3:a:infineon:rsa_library:1.02.013:*:*:*:*:*:*:*`

💡 **Idea: Measure string similarity between certificate name and CPEs**

```
for vuln in vulnerabilities:
  affected_cpes = vuln.get_affected_cpes()
  for certificate in cc_dataset:
    for cpe in affected_cpes:
      if string_similarity(certificate.title, cpe) > 0.9:
        certificate.related_cves = vuln
```

Linking certified products to vulnerabilities

- For every certified device, we have `` `(vendor, device name, heuristically extracted versions)` ``

> **Infineon Technologies** Security Controller M7793 A12 and G12 with optional **RSA**2048/4096 v1.02.010 or **v1.02.013**, EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 **libraries** and with specific IC-dedicated software

- Each vulnerability has a list of affected platforms specified with CPE
- RoCA vulnerability has, among others: `` `cpe:2.3:a:infineon:rsa_library:1.02.013:*:*:*:*:*:*:*` ``

💡 **Idea: Measure string similarity between certificate name and CPEs**

```
for vuln in vulnerabilities:
  affected_cpes = vuln.get_affected_cpes()
  for certificate in cc_dataset:
    for cpe in affected_cpes:
      if string_similarity(certificate.title, cpe) > 0.9:
        certificate.related_cves = vuln
```

Linking certified products to vulnerabilities

- For every certified device, we have `` `(vendor, device name, heuristically extracted versions)` ``

> **Infineon Technologies** Security Controller M7793 A12 and G12 with optional **RSA**2048/4096 v1.02.010 or **v1.02.013**, EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 **libraries** and with specific IC-dedicated software

- Each vulnerability has a list of affected platforms specified with CPE
- RoCA vulnerability has, among others: `` `cpe:2.3:a:infineon:rsa_library:1.02.013:*:*:*:*:*:*:*` ``

💡 **Idea: Measure string similarity between certificate name and CPEs**

```
for vuln in vulnerabilities:
  affected_cpes = vuln.get_affected_cpes()
  for certificate in cc_dataset:
    for cpe in affected_cpes:
      if string_similarity(certificate.title, cpe) > 0.9:
        certificate.related_cves = vuln
```

Linking certified products to vulnerabilities

- For every certified device, we have `` `(vendor, device name, heuristically extracted versions)` ``

> **Infineon Technologies** Security Controller M7793 A12 and G12 with optional **RSA**2048/4096 v1.02.010 or **v1.02.013**, EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 **libraries** and with specific IC-dedicated software

- Each vulnerability has a list of affected platforms specified with CPE
- RoCA vulnerability has, among others: `` `cpe:2.3:a:infineon:rsa_library:1.02.013:*:*:*:*:*:*:*` ``

💡 Idea: Measure string similarity between certificate name and CPEs

```
for vuln in vulnerabilities:
  affected_cpes = vuln.get_affected_cpes()
  for certificate in cc_dataset:
    for cpe in affected_cpes:
      if string_similarity(certificate.title, cpe) > 0.9:
        certificate.related_cves = vuln
```

Linking certified products to vulnerabilities

- For every certified device, we have `` `(vendor, device name, heuristically extracted versions)` ``

> **Infineon Technologies** Security Controller M7793 A12 and G12 with optional **RSA**2048/4096 v1.02.010 or **v1.02.013**, EC v1.02.010 or v1.02.013 and Toolbox v1.02.010 or v1.02.013 **libraries** and with specific IC-dedicated software

- Each vulnerability has a list of affected platforms specified with CPE
- RoCA vulnerability has, among others: `` `cpe:2.3:a:infineon:rsa_library:1.02.013:*:*:*:*:*:*:*` ``

💡 **Idea: Measure string similarity between certificate name and CPEs**

```
for vuln in vulnerabilities:
  affected_cpes = vuln.get_affected_cpes()
  for certificate in cc_dataset:
    for cpe in affected_cpes:
      if string_similarity(certificate.title, cpe) > 0.9:
        certificate.related_cves = vuln
```

DIRTY DATA YOU HAVE
CLEAN IT UP YOU MUST
memegenerator.net

Wait, is this an AI?

Wait, is this an AI?

- Technically, the problem above is an unsupervised classification problem

Wait, is this an AI?

- Technically, the problem above is an unsupervised classification problem
- Each certificate can get 800k labels

Wait, is this an AI?

- Technically, the problem above is an unsupervised classification problem
- Each certificate can get 800k labels
- All that we do is unsupervised

Wait, is this an AI?

- Technically, the problem above is an unsupervised classification problem
- Each certificate can get 800k labels
- All that we do is unsupervised

💡 **Idea: Label a managable subset of certificates to evaluate our classifiers externally.**

Wait, is this an AI?

- Technically, the problem above is an unsupervised classification problem
- Each certificate can get 800k labels
- All that we do is unsupervised

💡 **Idea: Label a managable subset of certificates to evaluate our classifiers externally.**

Results

- For 853 of 4892 certificates, we have >0 CPEs
- For 616 certificates, we have >0 CVEs

Wait, is this an AI?

- Technically, the problem above is an unsupervised classification problem
- Each certificate can get 800k labels
- All that we do is unsupervised

💡 **Idea: Label a managable subset of certificates to evaluate our classifiers externally.**

Results

- For 853 of 4892 certificates, we have >0 CPEs
- For 616 certificates, we have >0 CVEs
- When we say that certificate is affected by CVE, we’re right in **89%** of cases

Wait, is this an AI?

- Technically, the problem above is an unsupervised classification problem
- Each certificate can get 800k labels
- All that we do is unsupervised

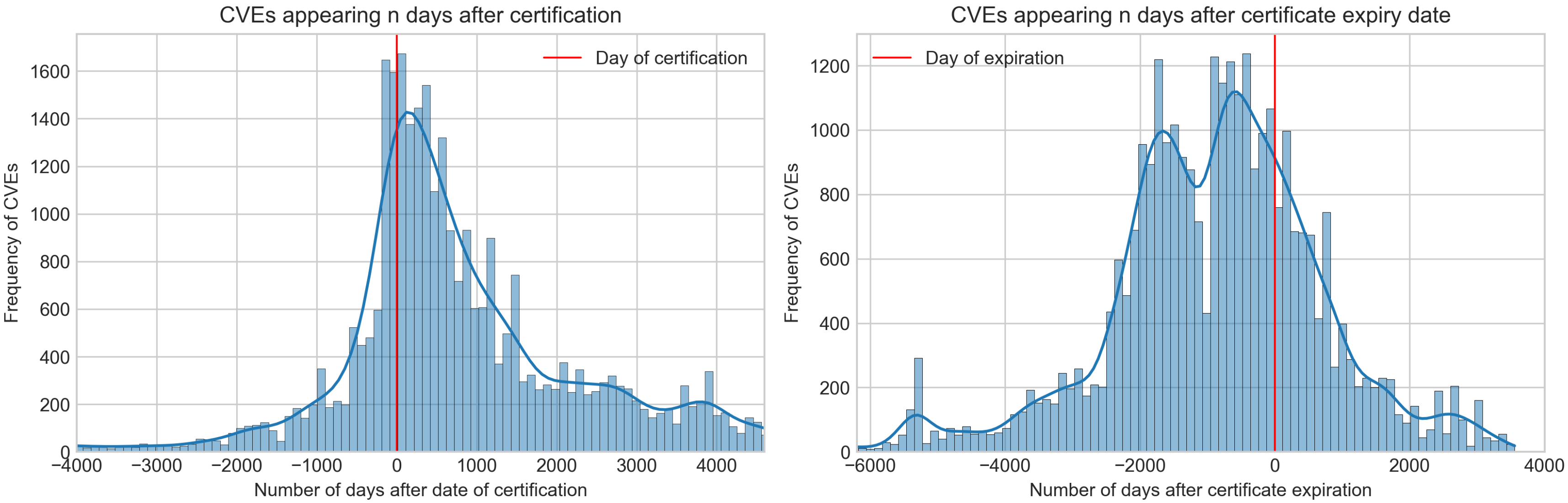
💡 **Idea: Label a managable subset of certificates to evaluate our classifiers externally.**

Results

- For 853 of 4892 certificates, we have >0 CPEs
- For 616 certificates, we have >0 CVEs
- When we say that certificate is affected by CVE, we're right in **89%** of cases
- Are we?

Very secure smartcards

When to watch out for vulnerabilities



Lifecycle of a product vs. its vulnerabilities

What to do to have fewer vulns?

What to do to have fewer vulns?

- Different Evaluation Assurance Levels (1-7) imply different Security Assurance Requirements

What to do to have fewer vulns?

- Different Evaluation Assurance Levels (1-7) imply different Security Assurance Requirements
- Several classes of SAR, e.g., `` `ATE` `` – product tests
  - `` `ATE_COV` `` - test coverage, levels, 3 levels
  - `` `ATE_IND` `` - independent testing, 3 levels

What to do to have fewer vulns?

- Different Evaluation Assurance Levels (1-7) imply different Security Assurance Requirements
- Several classes of SAR, e.g., `` `ATE` `` – product tests
  - `` `ATE_COV` `` - test coverage, levels, 3 levels
  - `` `ATE_IND` `` - independent testing, 3 levels
- We collected SARs from the certificates

What to do to have fewer vulns?

- Different Evaluation Assurance Levels (1-7) imply different Security Assurance Requirements
- Several classes of SAR, e.g., `` `ATE` `` – product tests
  - `` `ATE_COV` `` - test coverage, levels, 3 levels
  - `` `ATE_IND` `` - independent testing, 3 levels
- We collected SARs from the certificates

💡 **Idea: Level of SARs (EALs) should be negatively correlated to number and severity of vulnerabilities**

Are SARs correlated with vulnerabilities?

Are SARs correlated with vulnerabilities?

- Sometimes sketchy numbers, support of ~500 certificates, weak correlations

Are SARs correlated with vulnerabilities?

- Sometimes sketchy numbers, support of ~500 certificates, weak correlations
- EAL negatively correlated to the number of vulns ($\rho = -0.12$), and their avg. severity ($\rho = -0.15$)

Are SARs correlated with vulnerabilities?

- Sometimes sketchy numbers, support of ~500 certificates, weak correlations
- EAL negatively correlated to the number of vulns ($\rho = -0.12$), and their avg. severity ($\rho = -0.15$)
- With high test coverage (`` `ATE_COV` ``) and good functional tests (`` `ATE_FUN` ``) you can acually expect *more* vulns ($\rho = 0.14$)

Are SARs correlated with vulnerabilities?

- Sometimes sketchy numbers, support of ~500 certificates, weak correlations
- EAL negatively correlated to the number of vulns ($\rho = -0.12$), and their avg. severity ($\rho = -0.15$)
- With high test coverage (`` `ATE_COV` ``) and good functional tests (`` `ATE_FUN` ``) you can acually expect *more* vulns ($\rho = 0.14$)
- Highest negative correlation with severity: `` `ALC_DVS` ``
    - Life-cycle support - development security, $\rho = -0.19$

Are SARs correlated with vulnerabilities?

- Sometimes sketchy numbers, support of ~500 certificates, weak correlations
- EAL negatively correlated to the number of vulns ($\rho = -0.12$), and their avg. severity ($\rho = -0.15$)
- With high test coverage (`` `ATE_COV` ``) and good functional tests (`` `ATE_FUN` ``) you can acually expect *more* vulns ($\rho = 0.14$)
- Highest negative correlation with severity: `` `ALC_DVS` ``
  - Life-cycle support - development security, $\rho = -0.19$
- Highest negative correlation with vuln. number: `` `ATE_IND` ``
  - Independent tests, $\rho = -0.12$

Are SARs correlated with vulnerabilities?

- Sometimes sketchy numbers, support of ~500 certificates, weak correlations
- EAL negatively correlated to the number of vulns ($\rho = -0.12$), and their avg. severity ($\rho = -0.15$)
- With high test coverage (`` `ATE_COV` ``) and good functional tests (`` `ATE_FUN` ``) you can acually expect *more* vulns ($\rho = 0.14$)
- Highest negative correlation with severity: `` `ALC_DVS` ``
  - Life-cycle support - development security, $\rho = -0.19$
- Highest negative correlation with vuln. number: `` `ATE_IND` ``
  - Independent tests, $\rho = -0.12$
- Only 25 SARs evaluated, so no "extreme by random" results expected

Live demo

Some links worth exploring

- CVE profile: seccerts.org/vuln/cve/CVE-2017-15361
- Certificate profile: seccerts.org/cc/5efe98a1ba4df4d7/
- Fulltext search: seccerts.org/cc/ftsearch/
- Notebook: seccerts.org/docs/notebooks/examples/common_criteria.html

Conclusions

Conclusions

- Certification schemes are a mess

Conclusions

- Certification schemes are a mess
- We try to make sense of them with data analysis

Conclusions

- Certification schemes are a mess
- We try to make sense of them with data analysis
- This is a long term project

Conclusions

- Certification schemes are a mess
- We try to make sense of them with data analysis
- This is a long term project
- CVE analysis: not great, not terrible

Conclusions

- Certification schemes are a mess
- We try to make sense of them with data analysis
- This is a long term project
- CVE analysis: not great, not terrible
- References analysis: Many complex dependencies, complicated to analyze

Conclusions

- Certification schemes are a mess
- We try to make sense of them with data analysis
- This is a long term project
- CVE analysis: not great, not terrible
- References analysis: Many complex dependencies, complicated to analyze
- Future work: massive potential for NLP

Conclusions

- Certification schemes are a mess
- We try to make sense of them with data analysis
- This is a long term project
- CVE analysis: not great, not terrible
- References analysis: Many complex dependencies, complicated to analyze
- Future work: massive potential for NLP
- Contributions are welcomed!

Learn More

Web | Documentation | GitHub Repo

Slides: ajanovsky.cz/europen.pdf

Open-source development at university

- Junior developers
- Prepare the project for your leave
  - So that issues can be fixed semi-automagically
- When they finally learn it, they leave for different project

Some advice

- Let them do what they want to do
- Constraint their space for errors as much as possible
  - Use linters, enforce code style, protect branches, enforce tests, …