The Million-Key Question

Investigating the Origins of RSA Public Keys

Petr Švenda, Matúš Nemec, Peter Sekan, Rudolf Kvašňovský,
David Formánek, David Komárek and Vashek Matyáš

svenda@fi.muni.cz

Faculty of Informatics, Masaryk University, Czech Republic

CROCS
Centre for Research on Cryptography and Security

Crypto library code

$$P \times Q = N$$

RSA public key

N = 9782D7123C330444C88E279BF321EE84AC39524F1D84026327B04F32E1E930FC81588010178
DC75FCBF8258A068071317245D08817988813C4173495A922A41DA429A964F738020076EFFE7ED
5811088873C6E58EEF1CDC90059668[illegible]BE72368B51A821FC699E9C3FD66B377E2DF2485DC4
01DD99CC125890E5D969A6AC8B

e = 10001

OpenSSL

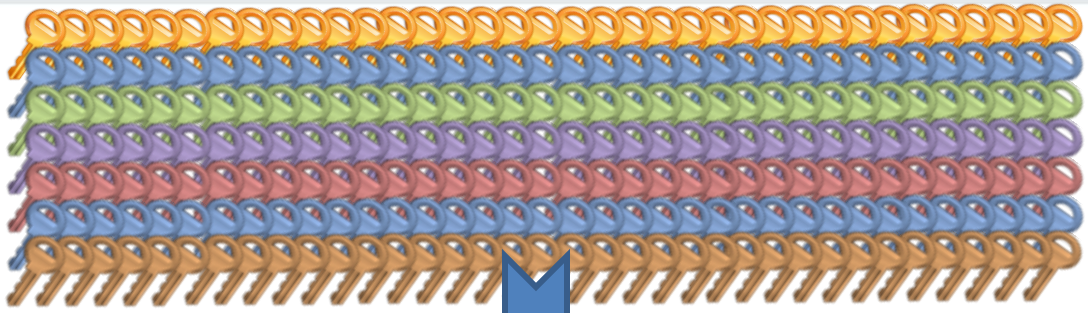
Infineon Crypto Java Ca[illegible]

60+ million fresh RSA keypairs

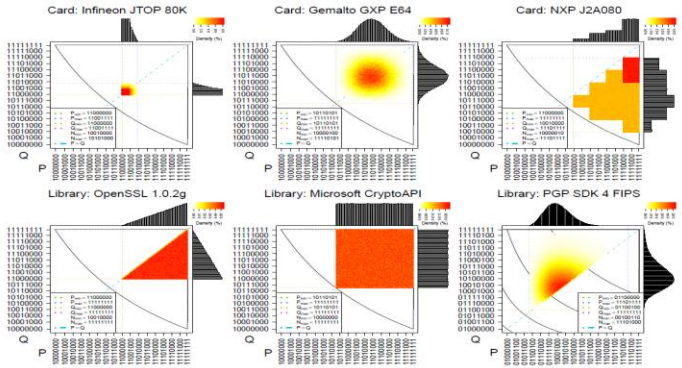
ARM mbed · FlexiProvider · Nettle · GnuPG · OpenSSL · cryptlib · OpenJDK · Microsoft · wolfSSL · LibTom · Botan · Crypto++ · PGP · FIPS 140-2

G&D Crypto Java Card · Infineon Crypto Java Card · Gemalto Crypto Java Card · NXP Crypto Java Card · Oberthur Crypto Java Card · Feitian Crypto Java Card

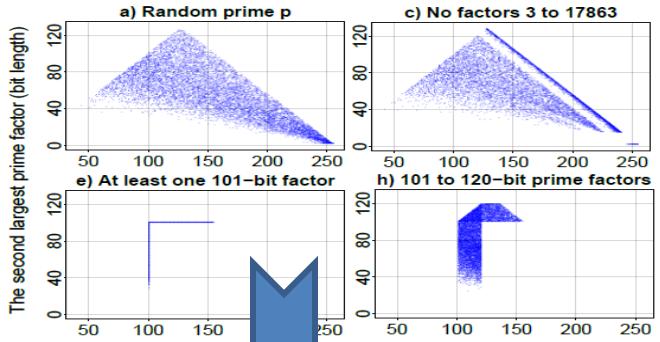
22 sw. libraries
16 smart cards



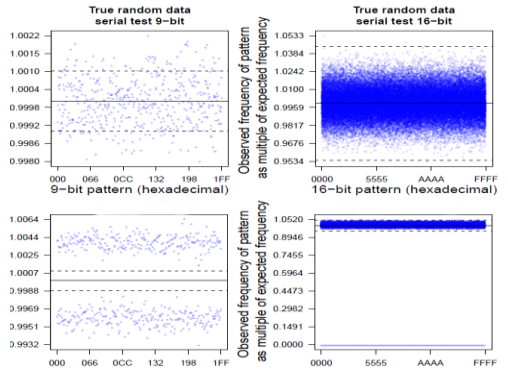
Distribution of primes (MSB)



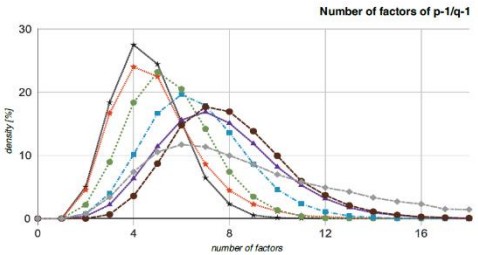
Large factors of p-1 / p+1



Bit stream statistics



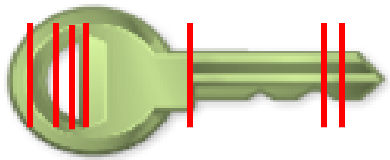
Number of factors



and more…

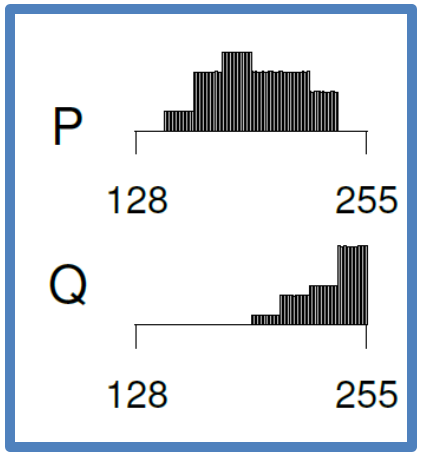
7 implementation choices observable in public keys

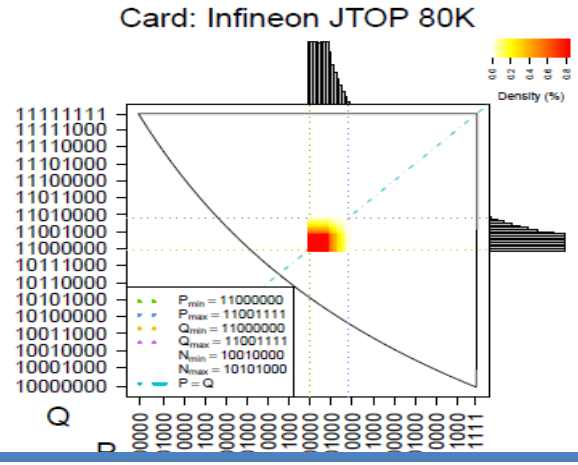
(biased bits of public modulus, “mask”)

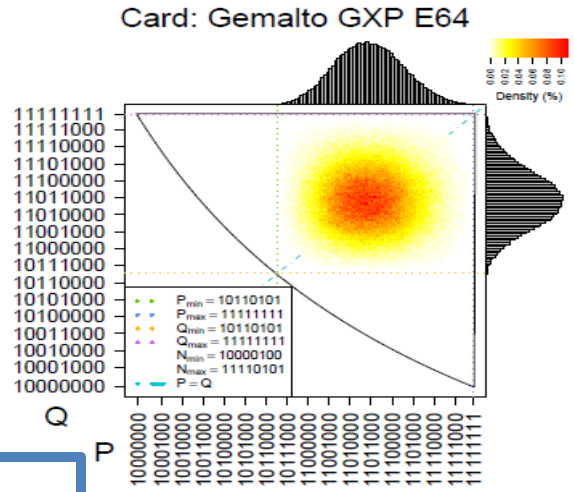


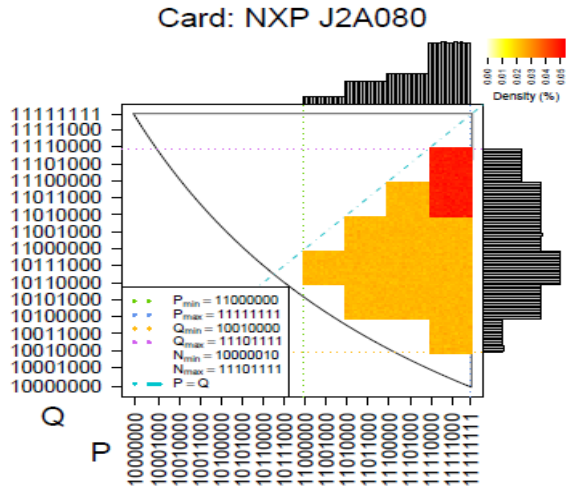
Heatmap of primes' most significant byte

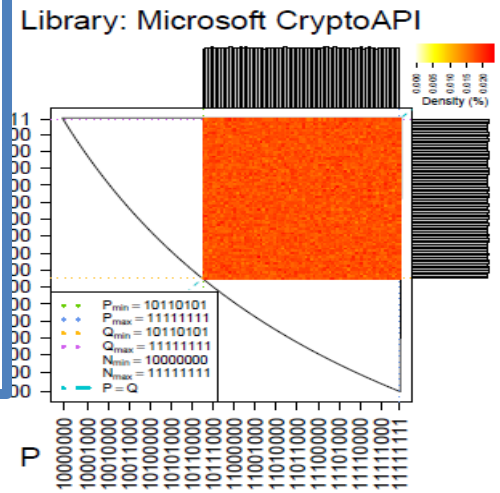
$$P \times Q = N$$

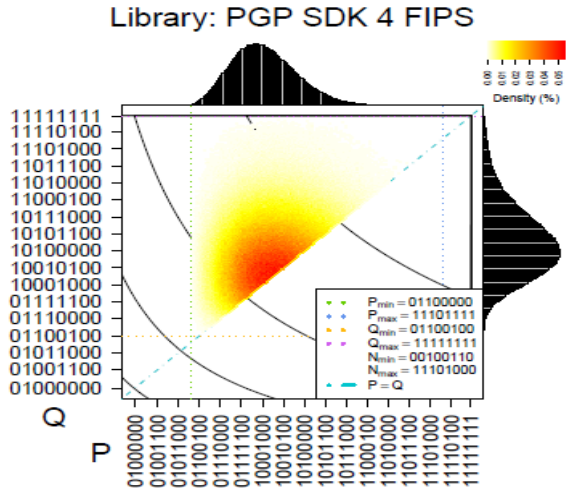






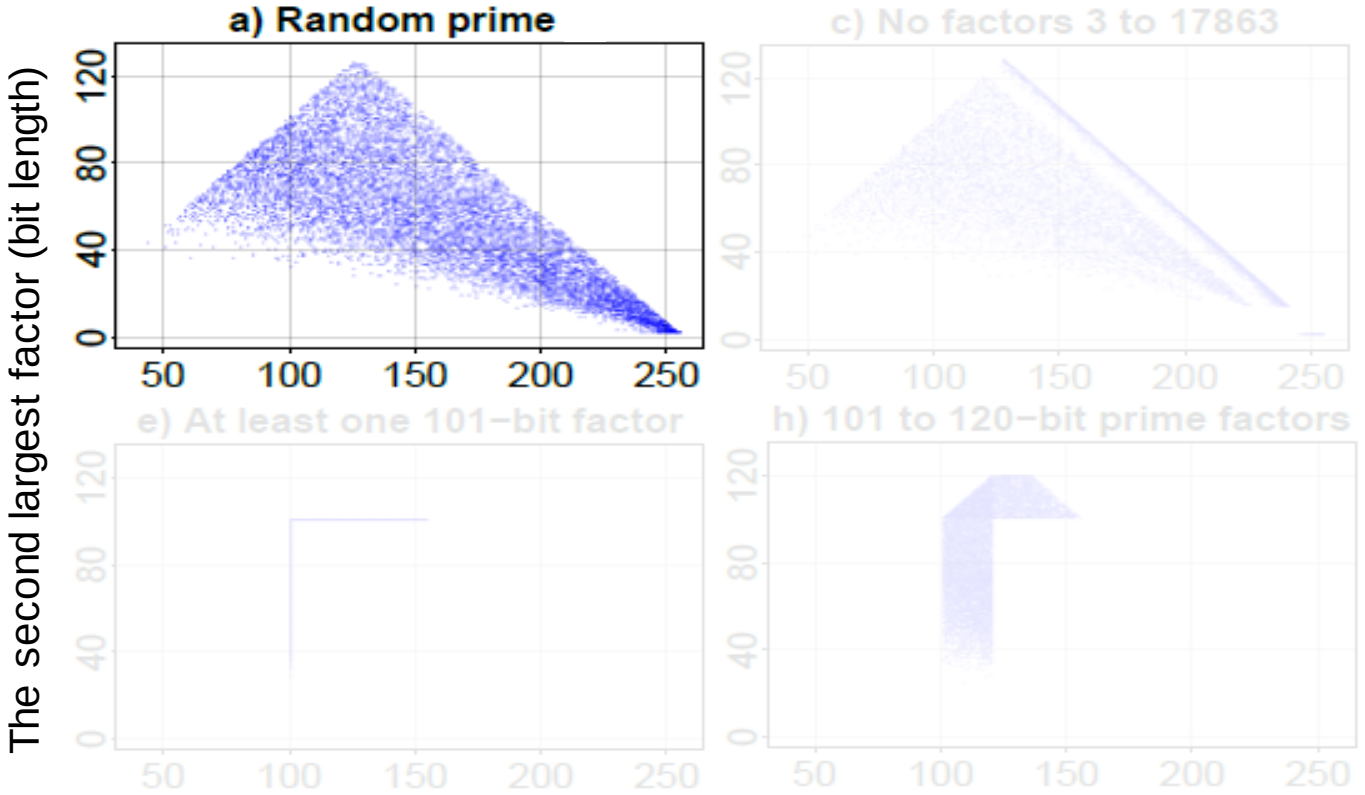






Factors of `P-1/Q-1` (and its impact on modulus N)

- For RSA512b, length of prime is 256bits => **`P-1/Q-1`** can be factorized
- We factorized 10k primes for every source with YAFU and…
- Small factors avoided
  - Significant bias on lower bits of N
  - Used by I. Mironov (OpenSSL)
- FIPS primes (specific range)
  - Not observable in modulus N



MIRONOV, I. Factoring RSA Moduli II. https://windowsontheory.org/2012/05/17/factoring-rsa-modulipart-ii/

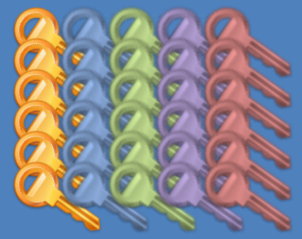
7 implementation choices observable in public key

Significance

1. Direct manipulation of the primes' highest bits 
2. Avoidance of small factors in P-1 and Q-1 
3. Requirement for moduli to be Blum integers 
4. Restriction of the primes' bit length 
5. Specific method to construct strong or provable primes 
6. Use of another non-traditional algorithm – functionally unknown, but statistically observable 
7. Type of action after candidate prime rejection 

Building classification matrix

Harvest keys from known sources (learning set)

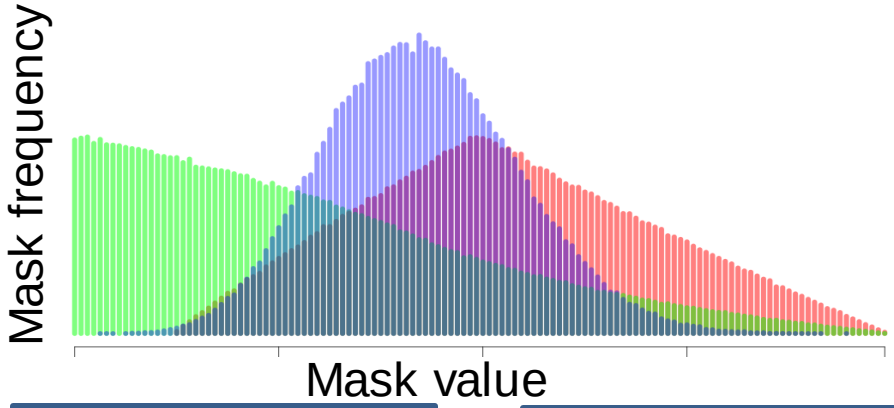


E.g., `101101110`

15x `000000000`
175x `000000001`
…
83x `111111121`

Apply mask to learning set

Count mask frequency



Group sources with very similar frequencies

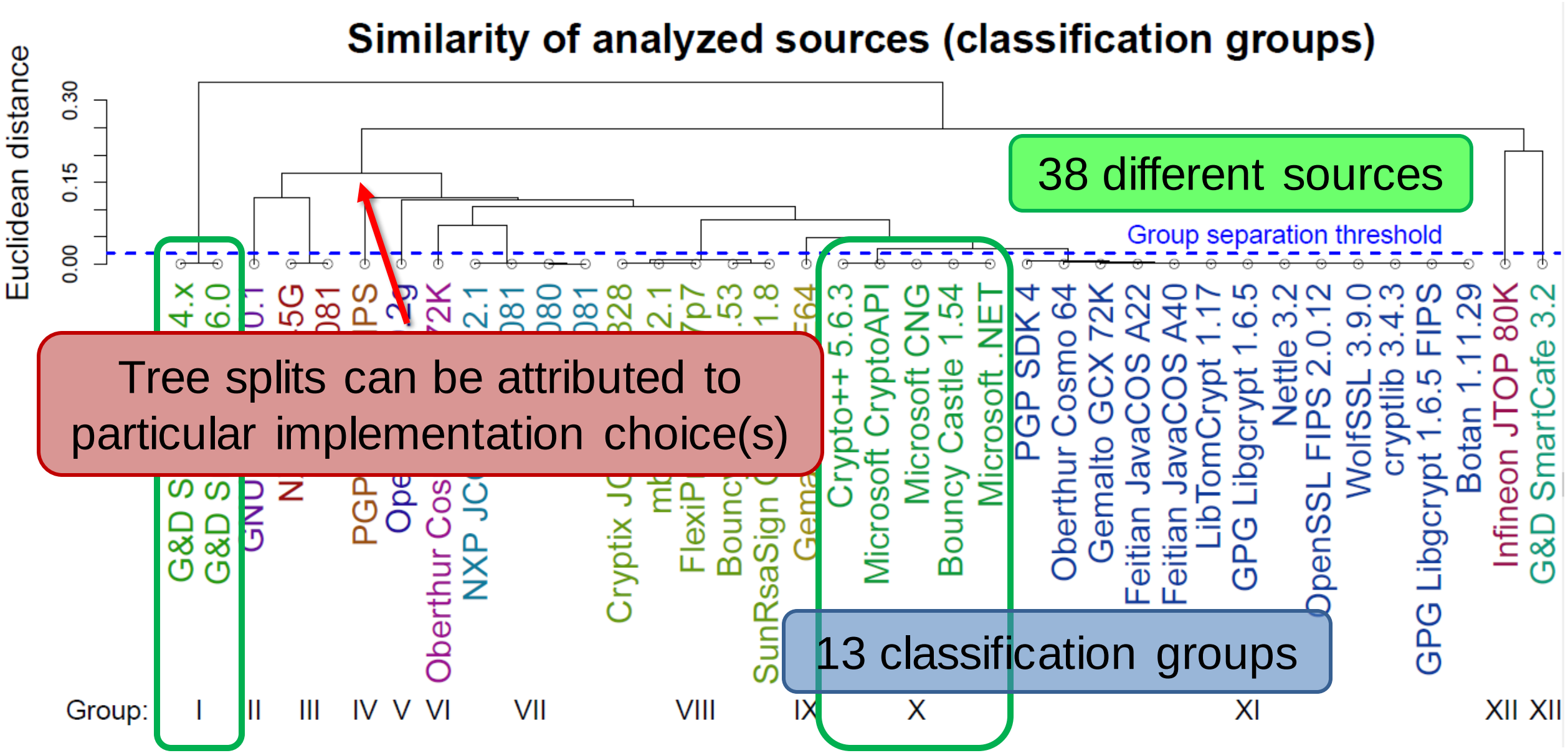
Normalize mask vectors of groups

Identification of biased modulus bits (mask, 9bits)

2nd-7th MSb | 2nd LSb | modulus mod 3 | len(modulus) mod 2

Classification matrix

Mask value	Group I	Group II	…	Group XII	Group XIII
000000000	0.124	0.347		0.105	0.012
000000001	0.004	0.038		0.236	0.454
000000011	0.046	0.002		0.447	0.112
…					
111111110	0.394	0.044		0.320	0.002
111111111	0.046	0.347		0.015	0.312


Similarity of analyzed sources (classification groups)
Euclidean distance
0.00
0.15
0.30
38 different sources
Group separation threshold
Tree splits can be attributed to particular implementation choice(s)
13 classification groups
Group:
I
II
III
IV
V
VI
VII
VIII
IX
X
XI
XII
XII
Crypto++ 5.6.3
Microsoft CryptoAPI
Microsoft CNG
Bouncy Castle 1.54
Microsoft .NET
PGP SDK 4
Oberthur Cosmo 64
Gemalto GCX 72K
Feitian JavaCOS A22
Feitian JavaCOS A40
LibTomCrypt 1.17
GPG Libgcrypt 1.6.5
Nettle 3.2
OpenSSL FIPS 2.0.12
WolfSSL 3.9.0
cryptlib 3.4.3
GPG Libgcrypt 1.6.5 FIPS
Botan 1.11.29
Infineon JTOP 80K
G&D SmartCafe 3.2

Input key

-----BEGIN CERTIFICATE-----
MIIG9zCCBd+gAwIBAgIIJOR2wFUwc20wDQYJKoZI
hvcNAQELBQAwSTELMAkGA1UEBhMCVVMxEzAR
BgNVBAoTCkdvb2dsZ...BJ...mMxJTAjBgNVBAMTHE
dvb2d...cm5idCB...dXRob3JpdHkgRzIwHhc
NMTY...NzA...YwOTI4MDgwMzA
wk2zl...qn...NRoC...D9zPk/rEp4miQ9aVgC6k7i
bLukl4c...c0kCOr8kNUBhH25DS6HpekTmO1s
9q81KbtS2E7+4Q/57xgdghBLiaTEv7O7+gskLQ/qJa
TouwiDPM6SHIVU6X2Ca1lNKg2wbx8h2Q63SDIwFJ
52HsNAClKp4ADvjvvImYoWVitcLlhpXogOAzbLz3Hls
6Jk=
-----END CERTIFICATE-----

Precomputed matrix

Mask value	Group I	Group II	…	Group XII	Group XIII
000000000	0.124	0.347		0.105	0.012
000000001	0.004	0.038		0.236	0.454
000000011	0.046	0.002		0.447	0.112
…					
111111110	0.394	0.044		0.320	0.002
111111111	0.046	0.347		0.015	0.312

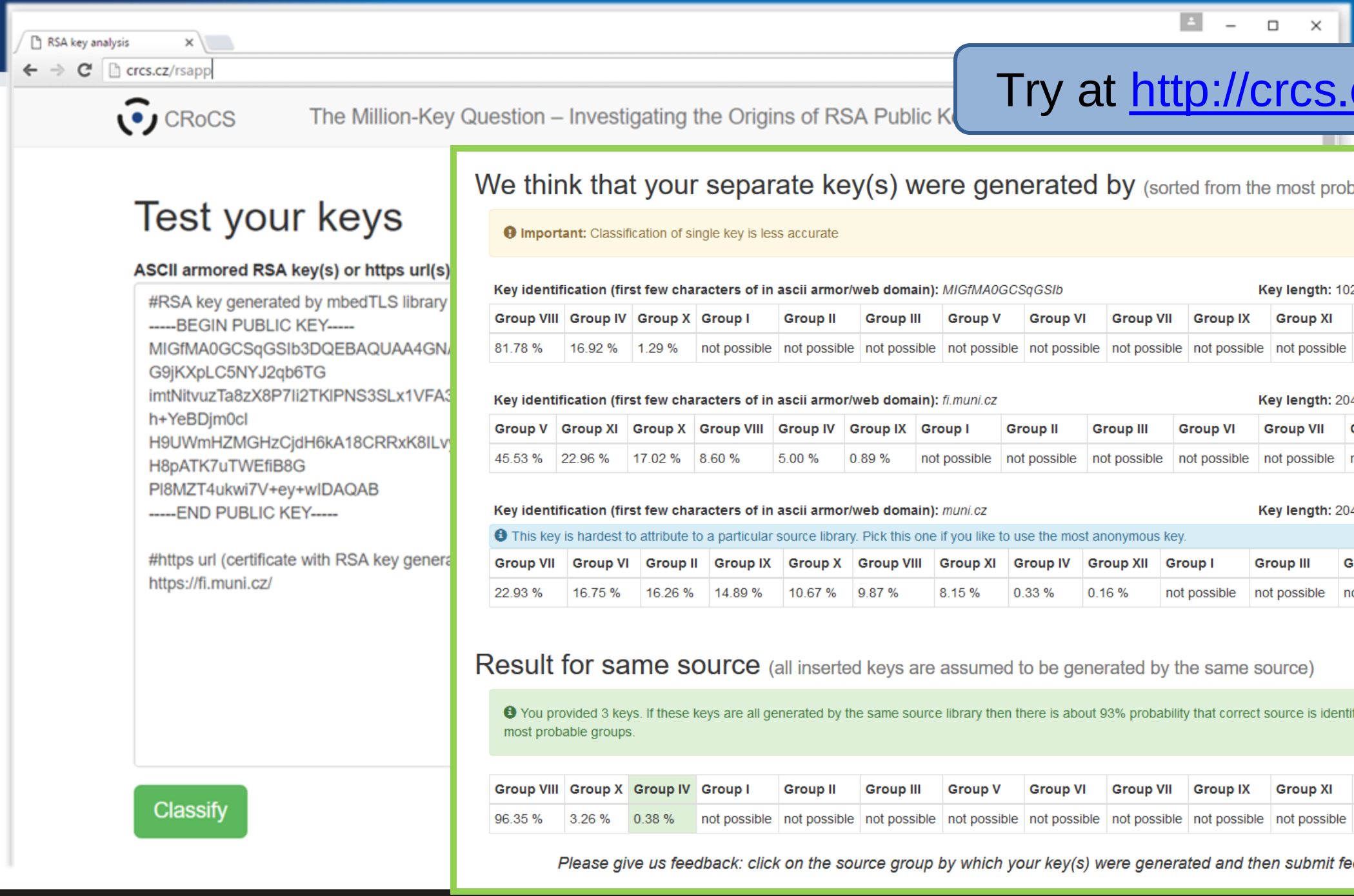
Classification

44% OpenSSL 's group

11% POLAR SSL 's group

9% PGP 's group

…



Try at http://crcs.cz/rsapp

We think that your separate key(s) were generated by (sorted from the most probable)

Important: Classification of single key is less accurate

Key identification (first few characters of in ascii armor/web domain): *MIGfMA0GCSqGSIb* **Key length:** 1024 **Exponent:** 65537

Group VIII	Group IV	Group X	Group I	Group II	Group III	Group V	Group VI	Group VII	Group IX	Group XI	Group XII	Group XIII
81.78 %	16.92 %	1.29 %	not possible	not possible	not possible	not possible	not possible	not possible	not possible	not possible	not possible	not possible

Key identification (first few characters of in ascii armor/web domain): *fi.muni.cz* **Key length:** 2048 **Exponent:** 65537

Group V	Group XI	Group X	Group VIII	Group IV	Group IX	Group I	Group II	Group III	Group VI	Group VII	Group XII	Group XIII
45.53 %	22.96 %	17.02 %	8.60 %	5.00 %	0.89 %	not possible	not possible	not possible	not possible	not possible	not possible	not possible

Key identification (first few characters of in ascii armor/web domain): *muni.cz* **Key length:** 2048 **Exponent:** 65537

This key is hardest to attribute to a particular source library. Pick this one if you like to use the most anonymous key.

Group VII	Group VI	Group II	Group IX	Group X	Group VIII	Group XI	Group IV	Group XII	Group I	Group III	Group V	Group XIII
22.93 %	16.75 %	16.26 %	14.89 %	10.67 %	9.87 %	8.15 %	0.33 %	0.16 %	not possible	not possible	not possible	not possible

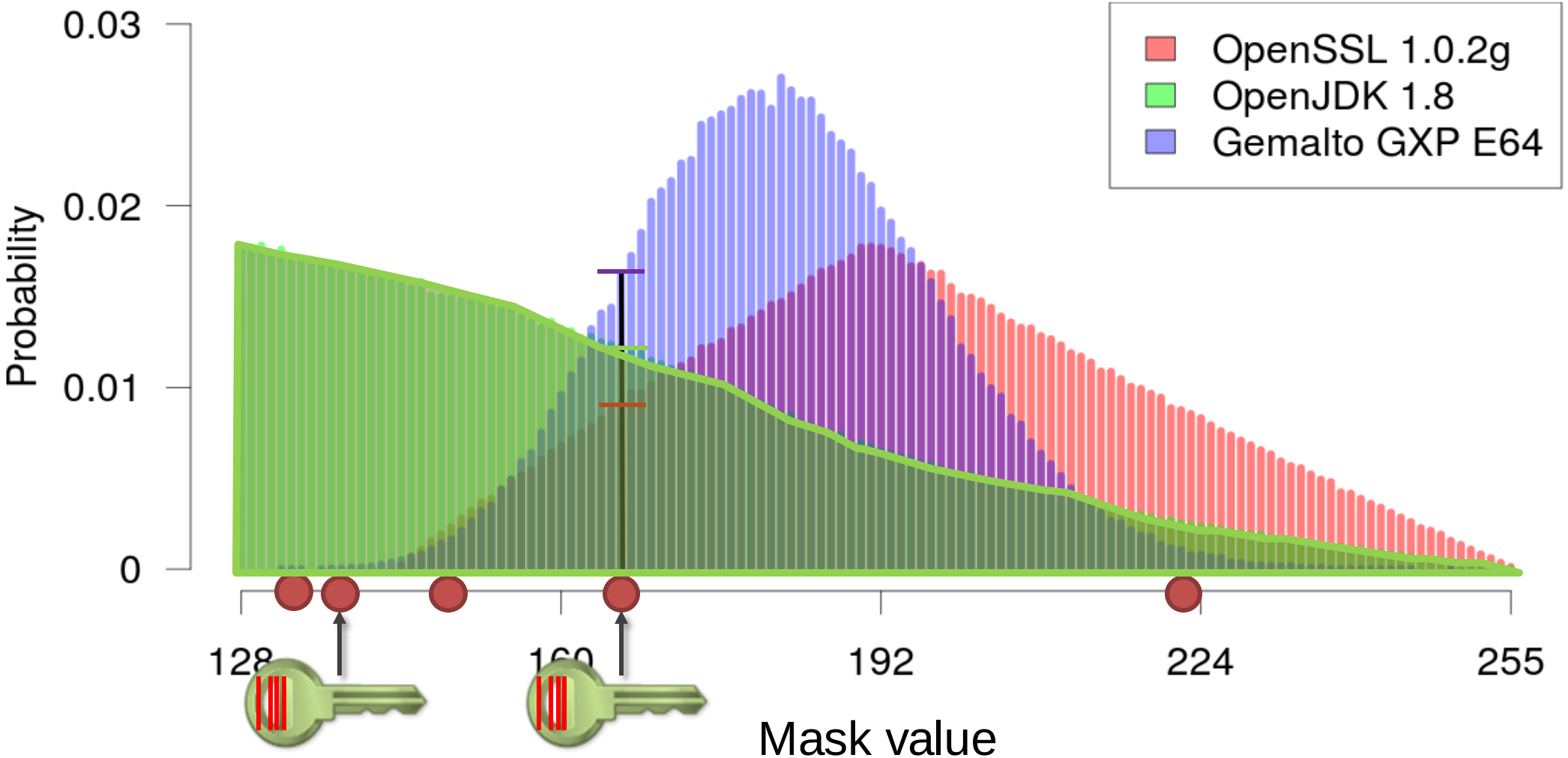
Result for same source (all inserted keys are assumed to be generated by the same source)

You provided 3 keys. If these keys are all generated by the same source library then there is about 93% probability that correct source is identified within the first three most probable groups.

Group VIII	Group X	Group IV	Group I	Group II	Group III	Group V	Group VI	Group VII	Group IX	Group XI	Group XII	Group XIII
96.35 %	3.26 %	0.38 %	not possible	not possible	not possible	not possible	not possible	not possible	not possible	not possible	not possible	not possible

Please give us feedback: click on the source group by which your key(s) were generated and then submit feedback form.

Classification accuracy



Classification accuracy (test set, 10k keys/source)

# keys in batch	Top 1 match 1	2	5	10	Top 3 match 1	2	5	10	Top 5 match 1	2	5	10
Group I	95.39%	98.42%	99.38%	99.75%								
Group II	17.75%	32.50%	58.00%	69.50%								
Group III	45.36%	72.28%	93.17%	98.55%								
Group IV	90.14%	97.58%	99.80%	100.00%								
Group V	63.38%	81.04%	97.50%	99.60%								
Group VI	54.68%	69.22%	88.45%	94.60%								
Group VII	7.58%	31.69%	64.21%	82.35%								
Group VIII	15.65%	40.30%	68.46%	76.60%								
Group IX	22.22%	45.12%	76.35%	83.00%	83.00%	54.57%	71.86%	85.25%	86.80%	88.00%	61.77%	81.96%	94.35%	95.00%	99.00%
Group X	0.63%	6.33%	27.42%	42.74%								
Group XI	11.77%	28.40%	55.56%	65.28%								
Group XII	60.36%	79.56%	97.20%	99.40%								
Group XIII	39.56%	70.32%	96.20%	99.70%								
Average	**40.34%**	**57.90%**	**78.59%**	**85.47%**								

1 key

Top 1: avg. **40.34%**, min. 0.63%, max. 95.36%

Top 3: avg. **73.09%**, min. 39.32%, max. 98.41%

5 keys

Top 1: avg. **78.59%**, min. 27.42%, max. 99.38%

Top 3: avg. **97.48%**, min. 91.45%, max. 100.00%

10 keys

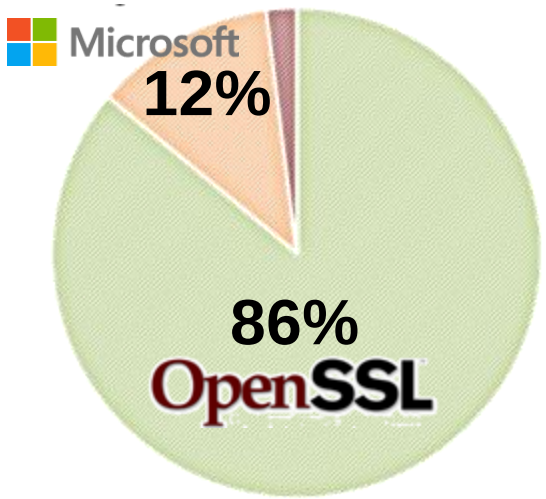
Top 1: avg. **85.47%**, min. 42.74%, max. 100.00%

Top 3: avg. **99.27%**, min. 95.00%, max. 100.00%

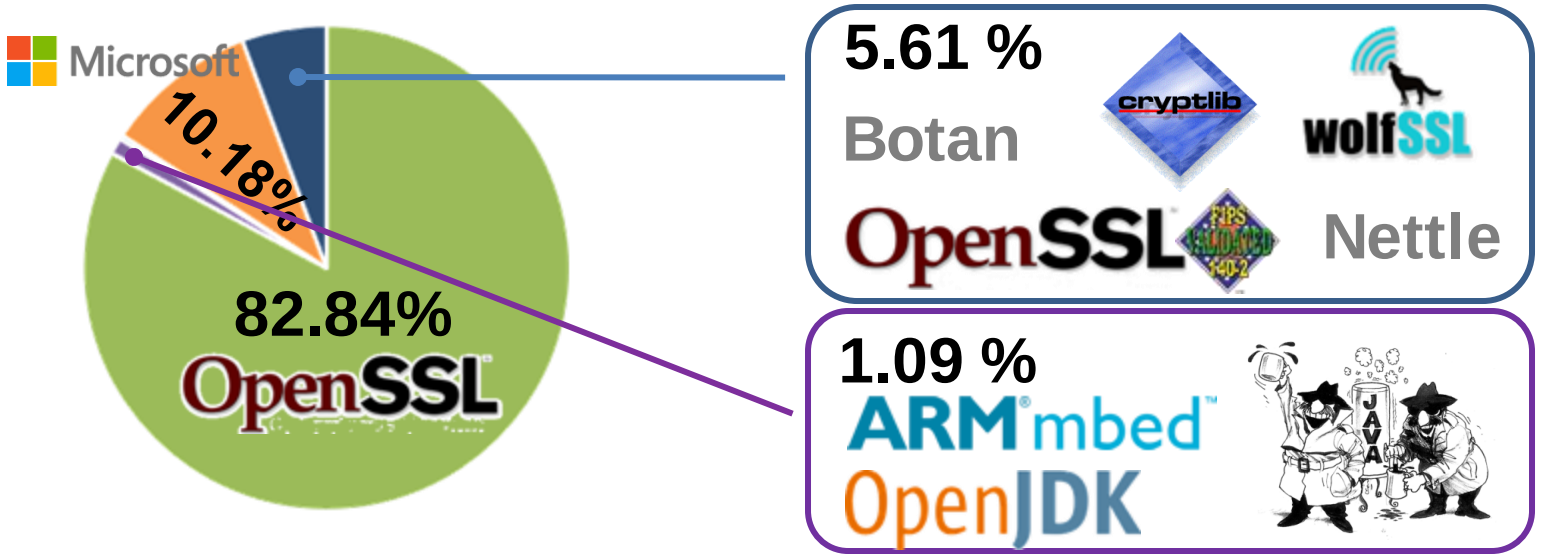
Sanity check with real world keys: IPv4 TLS dataset

- Datasets: IPv4 TLS scan(10M), PGP(1.4M), Cert. Transparency(13M)…
  - Problem: keys in these datasets are not annotated with source library
- Web servers market share => OpenSSL (~86%), Microsoft (~12%)

Expected



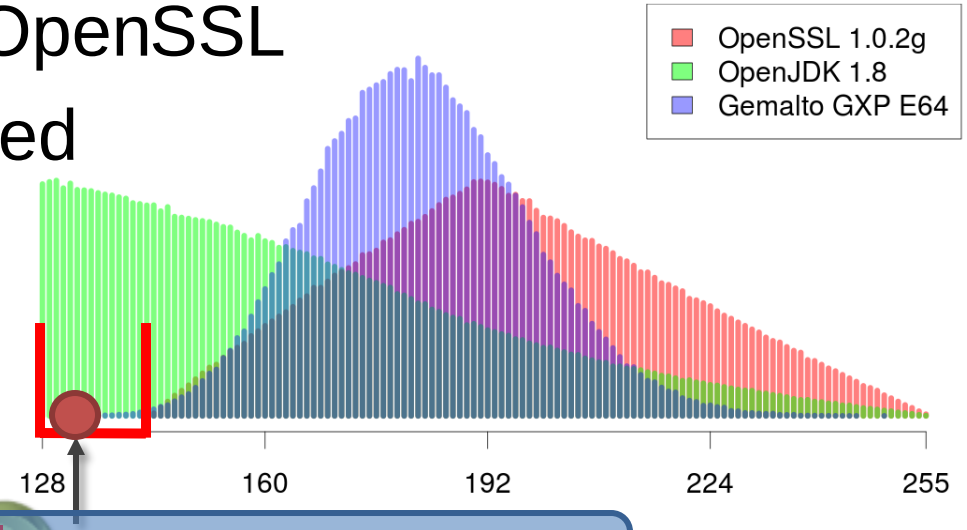
Classified (10-99 keys with same subject and issue date)



Sanity check: keys which *cannot* be from OpenSSL

- Keys with mask value never generated by OpenSSL
- Advantage: all keys from dataset can be used

Dataset	!OpenSSL
Cert. Transparency [16]	11.80%
PGP keyset [54]	47.35%
TLS IPv4 [15]	18.91%
Let's Encrypt [15]	1.83%



OpenSSL rare in PGP

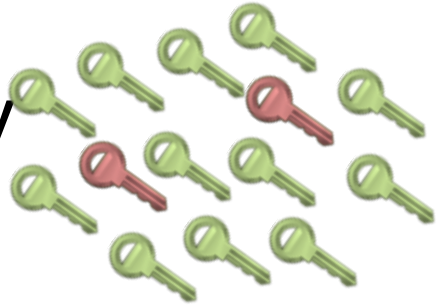
Leaves ~81 % for OpenSSL

OpenSSL is default client

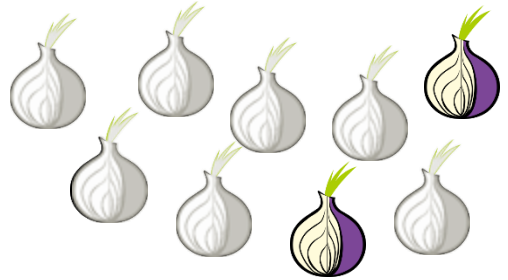
Impact (of the possibility) of public key classification

- Information leakage vulnerability

- Quick search for other keys from vulnerable library



- Linking related Tor hidden services operators



- Verify Crypto-as-a-Service use of secure hardware

OpenSSL

How to defend against public key classification?

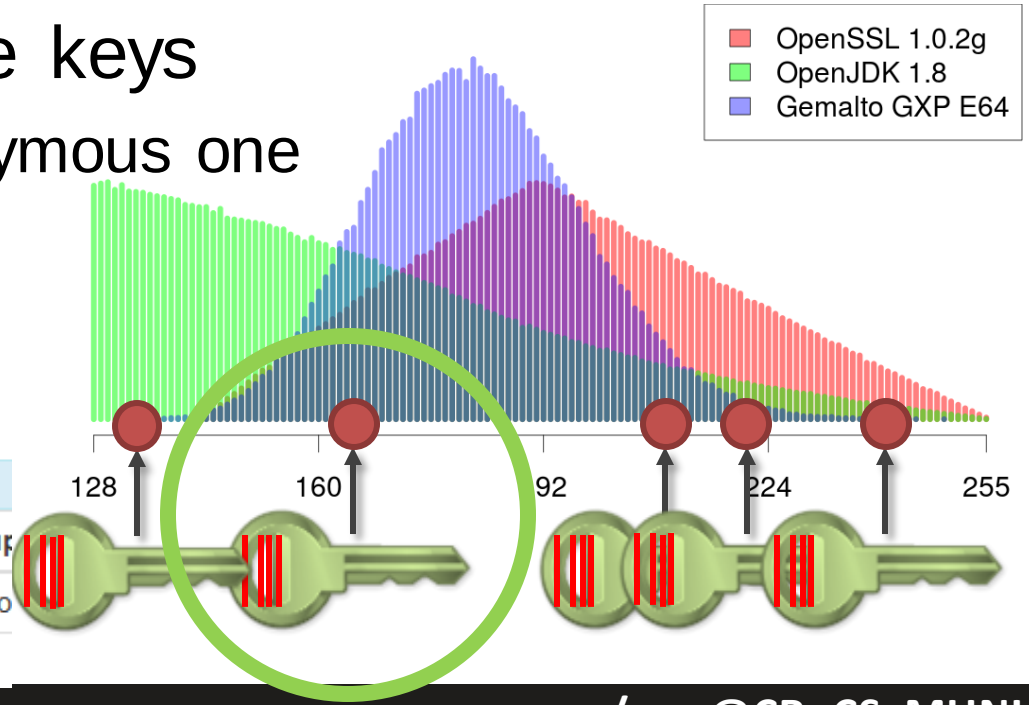
1. Developers of libraries - unify RSA key generation
   – Unlikely to happen soon, changes in critical part of code, legacy binaries…

2. Users of libraries – select from multiple keys
   – Generate multiple keys, pick the most anonymous one
   – Only about 5 keys required on average
   – http://crcs.cz/rsapp

Key identification (first few characters of in ascii armor/web domain): *muni.cz*

This key is hardest to attribute to a particular source library. Pick this one if you like to use the most anonymous key.

Group VII	Group VI	Group II	Group IX	Group X	Group VIII	Group XI	Group IV	Group XII	Group
22.93 %	16.75 %	16.26 %	14.89 %	10.67 %	9.87 %	8.15 %	0.33 %	0.16 %	not po



What else?

- More in the paper and technical report http://crcs.cz/rsa
  - Summary RSA generation techniques used by libraries and cards
  - Analysis of random data streams from smart cards (bias detected)
  - Systematic defect responsible for generating weak RSA keys
  - Time and power analysis of key generation on smart cards
- Download datasets and tools at http://crcs.cz/rsa

Limitations of the current work

1. Lower accuracy with single key only (40% on avg.)
2. Can't distinguish all libraries mutually (groups)
3. Some sources missing (HSMs…)
   - Will be misclassified at the moment

Conclusions

- RSA keypair generation observably bias public keys
  - Different libraries use different implementation choices
- Source library can be probabilistically estimated from RSA public key
  - Accuracy more than 85 % with 10 keys (>99 % within top three matches)
  - For some sources, even a single key is enough
- Information disclosure vulnerability
  - Forensics, de-anonymization, vulnerability scans, compliancy testing…
- Not easy to fix, will stay for longer time

Questions ?

Get tech. report and datasets at http://crcs.cz/rsa, try classification at http://crcs.cz/rsapp

BACKUP SLIDES

Similarity of analyzed sources (classification groups) with annotated differences