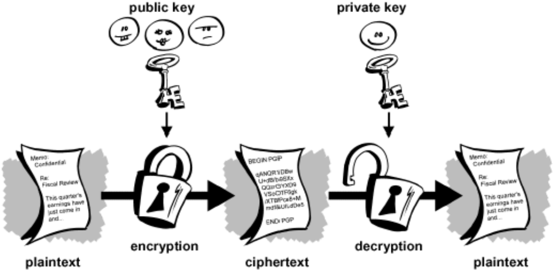
RSA Keys with Common Factors

Joppe W. Bos

Cryptography group
eXtreme Computing Group, Microsoft Research

Outline

Public-Key Cryptography



Cryptanalysis of Public-Key Cryptography

Popular Public-Key Cryptography

$\swarrow$ $\searrow$

RSA (EC)DSA / ElGamal

$\downarrow$ $\downarrow$

$\boxed{\text{Mathematical problem}}$

$\downarrow$ $\downarrow$

Integer factorization

$n = p \times q$

Find p or q, given n

(EC) Discrete Logarithm

$g = b^k \in G$

Find $\log_b(g) = k$, given b, g and G

Sanity check I

There have been many sanity checks of certificates and PKI

- Analyzing RSA Standards

  D. Loebenberger and M. Nüsken. Analyzing standards for RSA integers. In Africacrypt, 2011

- Analyzing X.509

  R. Holz, L. Braun, N. Kammenhuber, and G. Carle. The SSL landscape: a thorough analysis of the X.509 PKI using active and passive measurements. In ACM SIGCOMM, 2011

  N. Vratonjic, J. Freudiger, V. Bindschaedler, and J.-P. Hubaux. The inconvenient truth about web certificates. In The Workshop on Economics of Information Security, 2011

- Debian OpenSSL vulnerability

  S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage. When private keys are public: results from the 2008 Debian OpenSSL vulnerability. In Internet Measurement Conference, 2009

Sanity check I

There have been many sanity checks of certificates and PKI

- Analyzing RSA Standards

  D. Loebenberger and M. Nüsken. Analyzing standards for RSA integers. In Africacrypt, 2011

  > *The entropy of the output distribution [of standardized RSA key generation] is always almost maximal, ... and the outputs are hard to factor if factoring in general is hard.*

- Analyzing X.509

  R. Holz, L. Braun, N. Kammenhuber, and G. Carle. The SSL landscape: a thorough analysis of the X.509 PKI using active and passive measurements. In ACM SIGCOMM, 2011

  N. Vratonjic, J. Freudiger, V. Bindschaedler, and J.-P. Hubaux. The inconvenient truth about web certificates. In The Workshop on Economics of Information Security, 2011

- Debian OpenSSL vulnerability

  S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage. When private keys are public: results from the 2008 Debian OpenSSL vulnerability. In Internet Measurement Conference, 2009

Sanity check II

We look at things from a computational crypto point of view...

Our work

A. K. Lenstra, J. P. Hughes, M. Augier, J. W. Bos, T. Kleinjung, and C. Wachter.
Public-Keys. In CRYPTO 2012, LNCS vol. 7417, pp 626-642.
Full-version: Ron was wrong, Whit is right. In Cryptology ePrint Archive

At the same time...

N. Heninger, Z. Durumeric, E. Wustrow, J. A. Halderman.
Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices.
USENIX Security Symposium 2012

Sanity check II

We look at things from a computational crypto point of view...

Our work

A. K. Lenstra, J. P. Hughes, M. Augier, J. W. Bos, T. Kleinjung, and C. Wachter.
Public-Keys. In CRYPTO 2012, LNCS vol. 7417, pp 626-642.
Full-version: Ron was wrong, Whit is right. In Cryptology ePrint Archive

At the same time...

N. Heninger, Z. Durumeric, E. Wustrow, J. A. Halderman.
Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices.
USENIX Security Symposium 2012

Data collection – Summer 2009 - November 2011

Aug. '10: Download all publicly-visible SSL certificates on the IPv4 Internet

6 185 372	X.509 certificates
5 481 332	PGP keys
11 666 704	public keys

Data collection – Summer 2009 - November 2011

Aug. '10: Download all publicly-visible SSL certificates on the IPv4 Internet

6 185 372	X.509 certificates
5 481 332	PGP keys
11 666 704	public keys

X.509

6 185 230	RSA
141	DSA
1	ECDSA

47.6%: expiration date $>$ 2011
77.7%: use $\geq$ SHA-1
33.4%: satisfy both requirements

PGP keys

2 546 752	ElGamal
2 536 959	DSA
397 621	RSA

Ron Rivest, Adi Shamir and Leonard Adleman

RSA is the most widely used approach to achieve public-key cryptography

Keys

- Secret information: exponent d, prime factors p, q
- Public information: modulus n and the exponent e

$$n = p \times q \text{ with } p \approx q$$
$$gcd(e, (p-1)(q-1)) = 1 \text{ and } d \equiv e^{-1} \bmod (p-1)(q-1)$$

- Encryption: $c = m^e \bmod n$
- Decryption: $m = c^d \bmod n$

Check the public exponent

X.509		PGP		Combined	
e	%	e	%	e	%
65537	98.4921	65537	48.8501	65537	95.4933
17	0.7633	17	39.5027	17	3.1035
3	0.3772	41	7.5727	41	0.4574
35	0.1410	19	2.4774	3	0.3578
5	0.1176	257	0.3872	19	0.1506
7	0.0631	23	0.2212	35	0.1339
11	0.0220	11	0.1755	5	0.1111
47	0.0101	3	0.0565	7	0.0596
13	0.0042	21	0.0512	11	0.0313
65535	0.0011	$2^{127}+3$	0.0248	257	0.0241
other	0.0083	other	0.6807	other	0.0774

Note: 8 times $e = 1$ was used!

Check moduli sizes

Moduli sizes

%	bits	%	bits
0.01	384	0.04	3072
1.6	512	1.5	4096
0.8	768	0.01	8192
73.9	1024	0.003	16384
21.7	2048		

Primality and small factors

- 2 moduli are prime
- 171 have a factor $< 2^{24}$
- (68 are even)

These RSA keys were discarded.

Debian moduli

- 30 097 (21 459 distinct) blacklisted keys

Identical keys I — $n_1 = n_2$

Implications

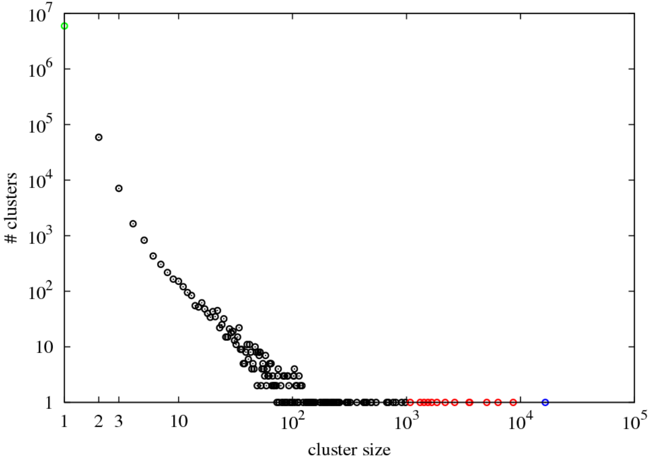
User 1 can decrypt all messages from user 2 (and vice versa)

- Most of the time harmless: renewal of key
- Possible explenation: Low-entropy when generating keys

```
seed(initial_randomness);
do { p=random(); } while( isprime(p) != true );
do { q=random(); } while( isprime(q) != true );
n = p*q;
```

Identical keys II

Cluster: certs/keys with the same modulus



Note: One cluster of size 16 489 4.3% of the RSA moduli are shared

Moduli with shared factors

$$\boxed{K_1 : a \times b} \quad \boxed{K_2 : c \times d}$$

- User 1 and user 2 have secure keys

Moduli with shared factors

$$\boxed{K_1 : a \times b} \quad \boxed{K_2 : c \times d}$$
$$\boxed{K_3 : b \quad \times \quad c}$$

- User 1 and user 3 share a factor and User 2 and 3 share a factor

Moduli with shared factors

$$\boxed{K_1 : a \times b} \quad \boxed{K_2 : c \times d}$$
$$\boxed{K_3 : b \quad \times \quad c}$$

- User 1 and user 3 share a factor and User 2 and 3 share a factor
- Greatest common divisor: **everyone** can break these keys!

Moduli with shared factors

Given two RSA moduli n_1 and n_2,

$$n_1 \neq n_2 \wedge gcd(n_1, n_2) \neq 1,$$

results in a complete loss of security for these moduli.

Checking all RSA keys for shared factors

- Straight-forward approach: $\approx$ ten core-years
- Smarter approach: $\approx$ ten core-hours

Moduli with shared factors

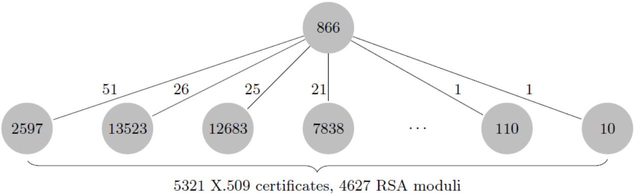
Given two RSA moduli n_1 and n_2,

$$n_1 \neq n_2 \wedge gcd(n_1, n_2) \neq 1,$$

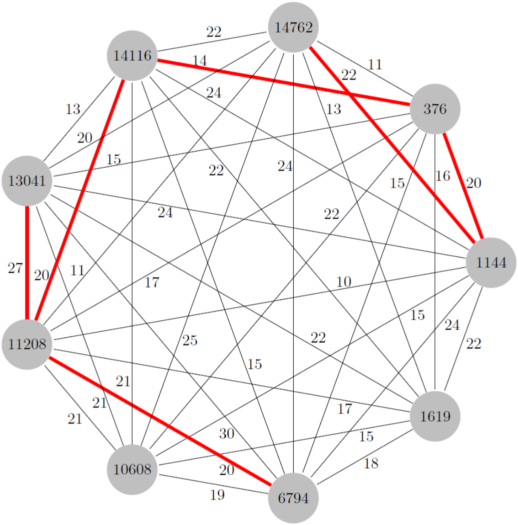
results in a complete loss of security for these moduli.

Checking all RSA keys for shared factors

- Straight-forward approach: $\approx$ ten core-years
- Smarter approach: $\approx$ ten core-hours



RSA keys V - K9


14762
14116
376
13041
1144
11208
1619
10608
6794
22
14
11
22
24
13
13
20
15
22
24
15
16
20
24
22
27
20
11
17
10
15
24
22
25
22
15
21
21
21
17
30
15
20
18
19

RSA keys VI

Affected keys

We found 14 901 distinct primes factoring 12 934 distinct moduli
21 419 X.509 certs and PGP keys are affected
None of these are blacklisted

Primes

#	bits
307	256
2	257
14 592	512

Moduli

#	bits
214	512
12 720	1024

- **3 201** 1024-bit RSA moduli occur in 5 250 certificates which are not-expired and use SHA-1

RSA keys VII - Discussion

RSA requires generating two random prime numbers
These primes must not be selected by anyone else before

NIST recommends: size(random seed) = $2\times$ size(security level)

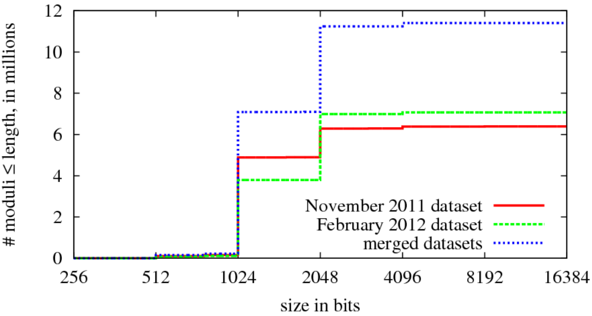
Possible explanations:

- Poor random initial seeding $\rightarrow$ duplicate keys
- Using local entropy after each guess
  - "poor initial guess" p_1, with prob $1/\log(p_1)$ this is prime
  - next guesses use the local entropy

```
seed(initial_randomness);
do { p=random(); } while( isprime(p) != true );
do { q=random(); } while( isprime(q) != true );
n = p*q;
```

February 2012, new scan by EFF

7.2M distinct X.509 certs (up from 6.2M)



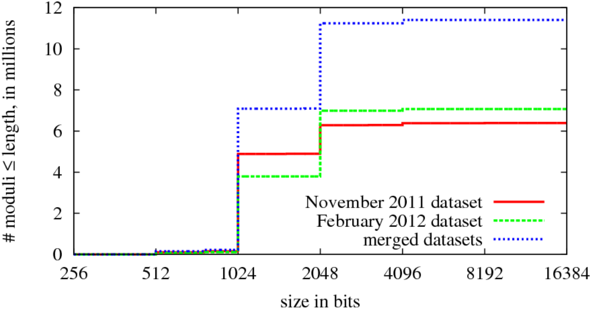
February 2012, new scan by EFF

7.2M distinct X.509 certs (up from 6.2M)

RSA-1024

- 4.7M $\rightarrow$ 3.7M keys
- $>$ 5000 affected keys are no longer present
- 13 019 new keys affected

New: 10 RSA-2048 keys are affected, two have not expired and use SHA-1



Conclusions

Multi-secret systems (RSA) vs. single-secret systems (ElGamal, (EC)DSA)

Conclusions

Multi-secret systems (RSA) vs. single-secret systems (ElGamal, (EC)DSA)

Possible remedy

Moduli pq for k-bit primes p chosen such that

$$q = \left\lceil \frac{2^{2k-1} + p - (2^{2k-1} \bmod p)}{p} \right\rceil \text{ is prime}$$

A. Lenstra. Generating RSA Moduli with a Predetermined Portion. In Asiacrypt 1998

Conclusions

Multi-secret systems (RSA) vs. single-secret systems (ElGamal, (EC)DSA)

Possible remedy

Moduli pq for k-bit primes p chosen such that

$$q = \left\lceil \frac{2^{2k-1} + p - (2^{2k-1} \bmod p)}{p} \right\rceil \text{ is prime}$$

A. Lenstra. Generating RSA Moduli with a Predetermined Portion. In Asiacrypt 1998

Misinterpretations in the Media

- "*This is simply the Debian PRNG bug*"
  All our results exclude the blacklisted Debian keys.
- "*RSA is insecure*"
  When properly generating random primes then RSA is still secure.
- "*Only embedded devices are affected*"
  We have multiple examples of affected keys between users.