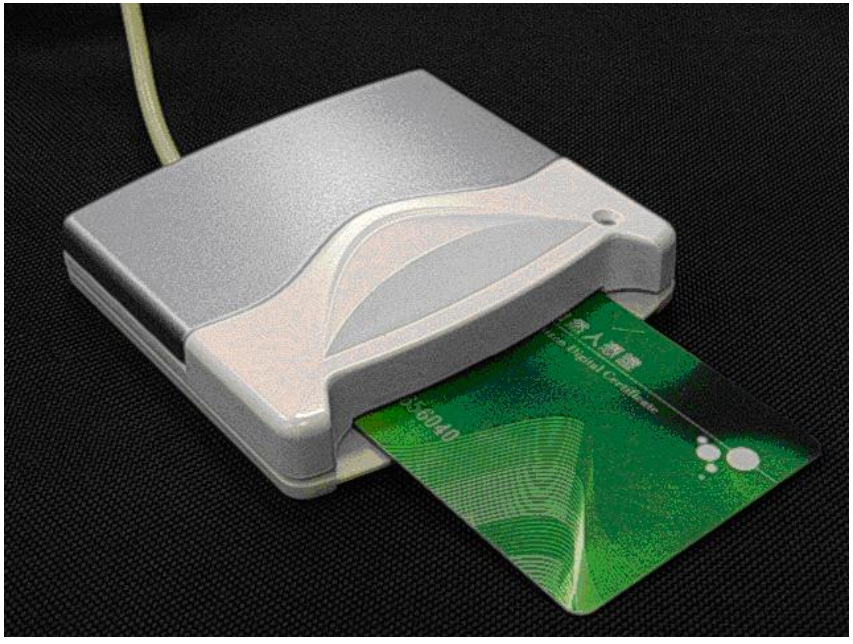
Factoring RSA keys from certified smart cards: Coppersmith in the wild

Daniel J. Bernstein, Yun-An Chang,
Chen-Mou Cheng, Li-Ping Chou,
Nadia Heninger, Tanja Lange,
Nicko van Someren

Taiwan Citizen Digital Certificate

Government-issued smart cards allow citizens to

- file income taxes,
- update car registrations,
- transact with government agencies,
- interact with companies (e.g. Chunghwa Telecom) online.



FIPS-140 and Common Criteria Level 4+ certified.

Taiwan Citizen Digital Certificate

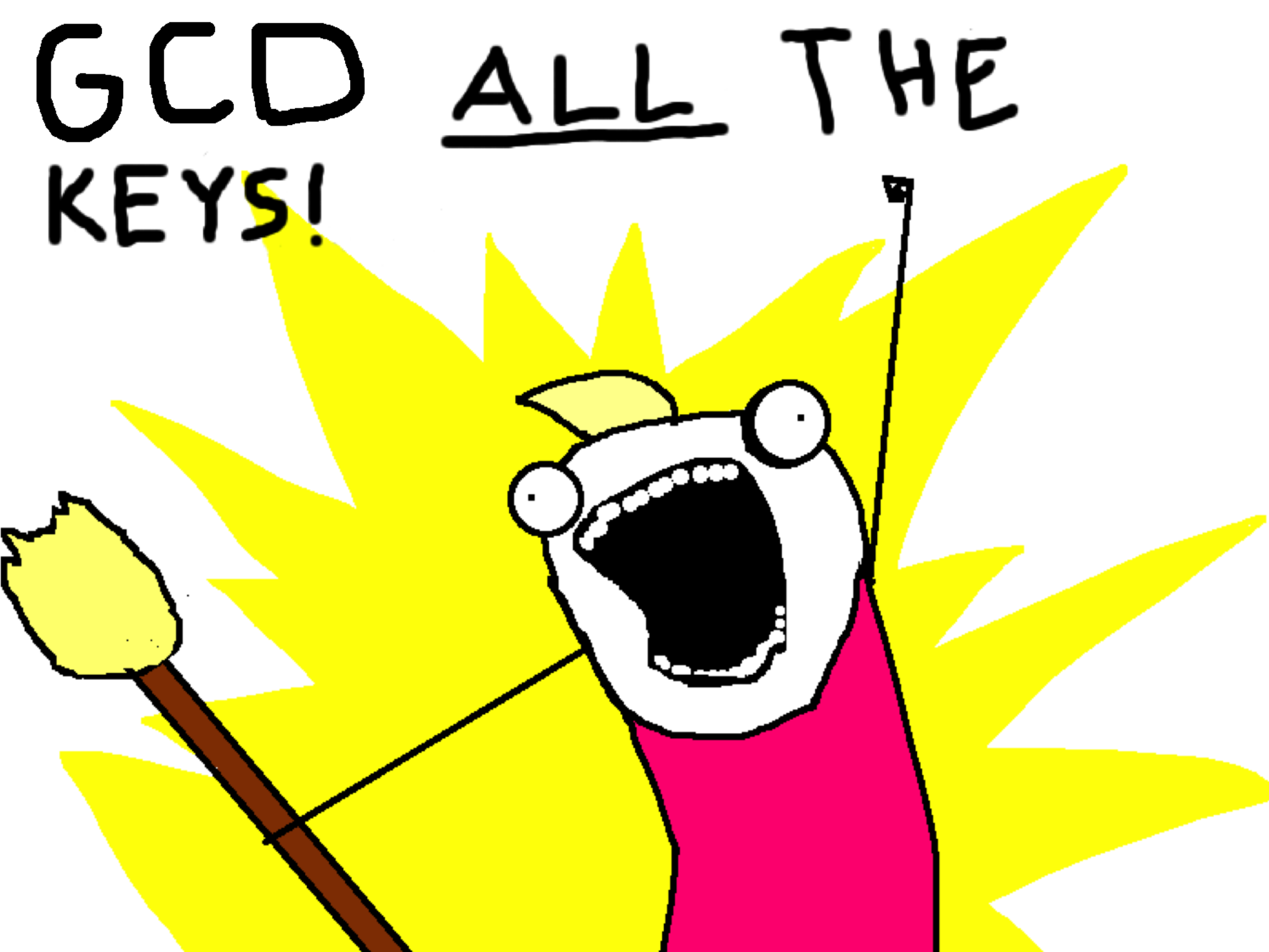
Collected 3,002,000 certificates (all using RSA keys) from national LDAP directory.

2.3 million distinct 1024-bit RSA moduli, 700,000 2048-bit moduli.

Certificate of Chen-Mou Cheng

```
Data: Version: 3 (0x2)
Serial Number: d7:15:33:8e:79:a7:02:11:7d:4f:25:b5:47:e8:ad:38
Signature Algorithm: sha1WithRSAEncryption
Issuer: C=TW, O=XXX
Validity
     Not Before: Feb 24 03:20:49 2012 GMT
     Not After : Feb 24 03:20:49 2017 GMT
Subject: C=TW, CN=YYY serialNumber=0000000112831644
Subject Public Key Info:
     Public Key Algorithm: rsaEncryption
Public-Key: (2048 bit) Modulus:
     00:bf:e7:7c:28:1d:c8:78:a7:13:1f:cd:2b:f7:63:
     2c:89:0a:74:ab:62:c9:1d:7c:62:eb:e8:fc:51:89:
     b3:45:0e:a4:fa:b6:06:de:b3:24:c0:da:43:44:16:
     e5:21:cd:20:f0:58:34:2a:12:f9:89:62:75:e0:55:
     8c:6f:2b:0f:44:c2:06:6c:4c:93:cc:6f:98:e4:4e:
     3a:79:d9:91:87:45:cd:85:8c:33:7f:51:83:39:a6:
     9a:60:98:e5:4a:85:c1:d1:27:bb:1e:b2:b4:e3:86:
     a3:21:cc:4c:36:08:96:90:cb:f4:7e:01:12:16:25:
     90:f2:4d:e4:11:7d:13:17:44:cb:3e:49:4a:f8:a9:
     a0:72:fc:4a:58:0b:66:a0:27:e0:84:eb:3e:f3:5d:
     5f:b4:86:1e:d2:42:a3:0e:96:7c:75:43:6a:34:3d:
     6b:96:4d:ca:f0:de:f2:bf:5c:ac:f6:41:f5:e5:bc:
     fc:95:ee:b1:f9:c1:a8:6c:82:3a:dd:60:ba:24:a1:
     eb:32:54:f7:20:51:e7:c0:95:c2:ed:56:c8:03:31:
     96:c1:b6:6f:b7:4e:c4:18:8f:50:6a:86:1b:a5:99:
     d9:3f:ad:41:00:d4:2b:e4:e7:39:08:55:7a:ff:08:
     30:9e:df:9d:65:e5:0d:13:5c:8d:a6:f8:82:0c:61:
     c8:6b
Exponent: 65537 (0x10001)

        .
        .
        .
```


GCD ALL THE KEYS!

All-pairs GCD algorithm factors

103 keys.

Most commonly shared factor appears 46 times

```
c000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
0000000000000000000000000002f9
```

Next most common factor appears 7 times

c9242492249292499249492449242492
2492924992494924492424922492924 9
9249492449242492249292499249492 4
4924249224929249924949244924 24e5

Hypothesized key generation process for weak primes:

1. Choose a bit pattern of length 1, 3, 5, or 7 bits.
2. Repeat it to cover 512 bits.
3. For every 32-bit word, swap the lower and upper 16 bits.
4. Fix the most significant two bits to 11.
5. Find the next prime greater than or equal to this number.

Factoring by trial division

1. Generate all primes of this form.

2. Trial division.

Factoring by trial division

1. Generate all primes of this form.

2. Trial division.

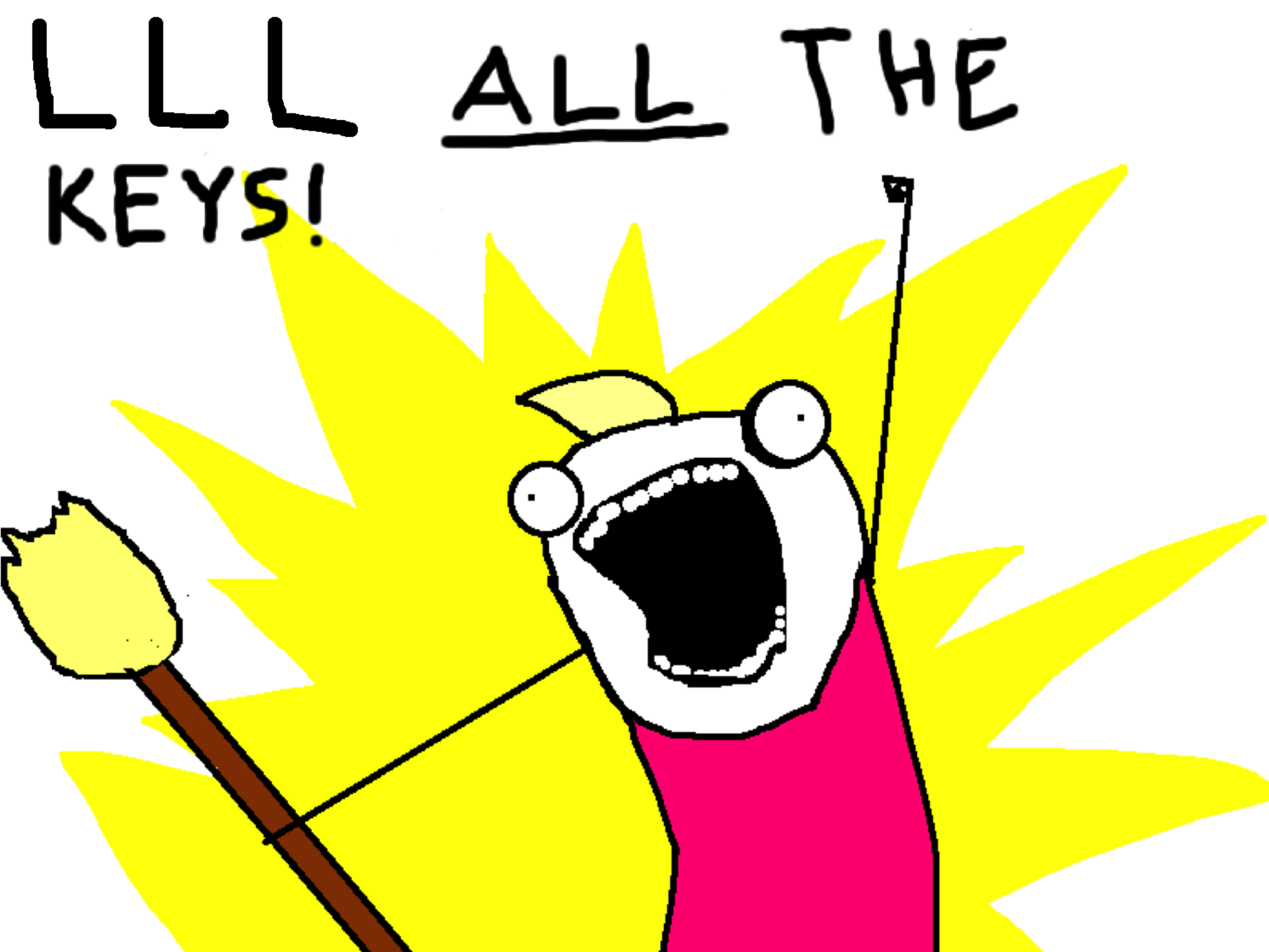
Enumerating all patterns factored **18 new keys**.

Extending to patterns of length 9: **4 more keys**.

Some more prime factors

```
c0000000000000000000000000000000
0000000000000000000000000000000000
0000000000000000000000000000000000
00000000000000000000000000101ff

c0000000000000000000000000000000
0000000000000000000000000000000000
0000000000000000000000000000000000
000000000000000000000100000177
```


LLL ALL THE
KEYS!

Factoring with Coppersmith

1. For all patterns a and moduli N, run LLL on

$$\begin{bmatrix} X^2 & Xa & 0 \\ 0 & X & a \\ 0 & 0 & N \end{bmatrix}$$

2. Hope $a + x$ factors N.

- For 1024-bit N, X as large as 170 bits.
- Factored **39 new keys**

```
ffffaa55fffffffff3cd9fe3ffff676
fffffffffffe0000000000000000000
0000000000000000000000000000000
000000000000000000000000000009d

c000b80000000000000000000000000
0000000000000000000000000000000
0000068000000000000000000000000
000000000000000000000000000251
```

Factoring with Bivariate Coppersmith

Search for prime factors of the form

$$p = a + 2^t x + y$$

- Works with 6, 10, or 15-dimensional lattices.
- Ran on 20 most common patterns and factored **13 more keys**.

Why are government-issued smartcards generating weak keys?

Card behavior very clearly not FIPS-compliant.

Why are government-issued smartcards generating weak keys?

Card behavior very clearly not FIPS-compliant.

Hypothesized failure:

- Hardware ring oscillator gets stuck in some conditions.
- Card software not post-processing RNG output.

Lessons:

Nontrivial GCD is not the only way RSA can fail with bad RNG.

Lessons:

Nontrivial GCD is not the only way RSA can fail with bad RNG.

Future work:

Lessons:

Nontrivial GCD is not the only way RSA can fail with bad RNG.

Future work:

- Breaking RSA-1024 with Fermat factoring.

Lessons:

Nontrivial GCD is not the only way RSA can fail with bad RNG.

Future work:

- Breaking RSA-1024 with Fermat factoring.
- Breaking RSA-1024 using Adi Shamir's secret database of all primes.

Lessons:

Nontrivial GCD is not the only way RSA can fail with bad RNG.

Future work:

- Breaking RSA-1024 with Fermat factoring.
- Breaking RSA-1024 using Adi Shamir's secret database of all primes.
- Breaking RSA-1024 using $1024 = 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2$.

Lessons:

Nontrivial GCD is not the only way RSA can fail with bad RNG.

Future work:

- Breaking RSA-1024 with Fermat factoring.
- Breaking RSA-1024 using Adi Shamir's secret database of all primes.
- Breaking RSA-1024 using $1024 = 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2$.
- Breaking RSA-1024 using Intel's new RDRAND_NSAKEY instruction.