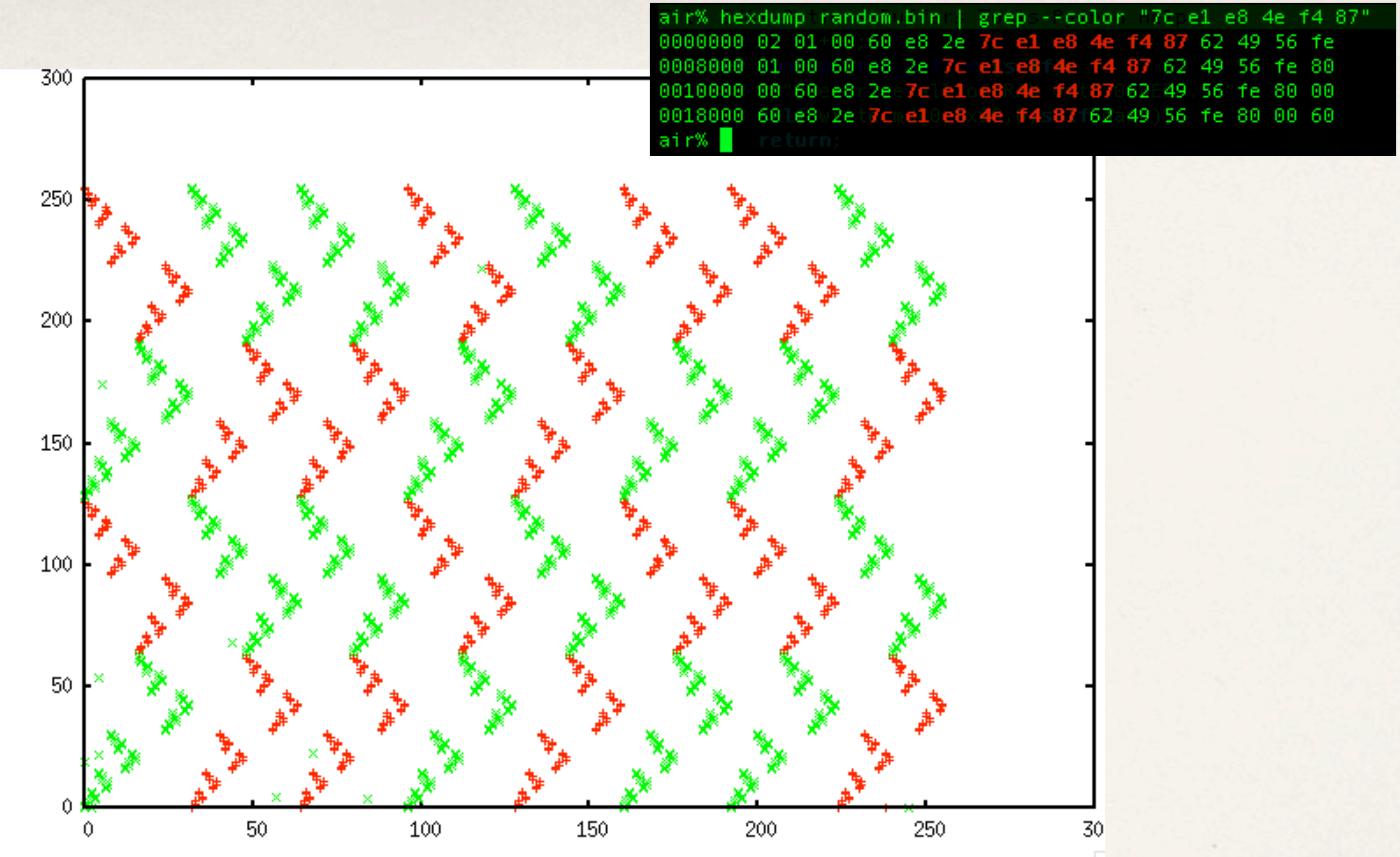
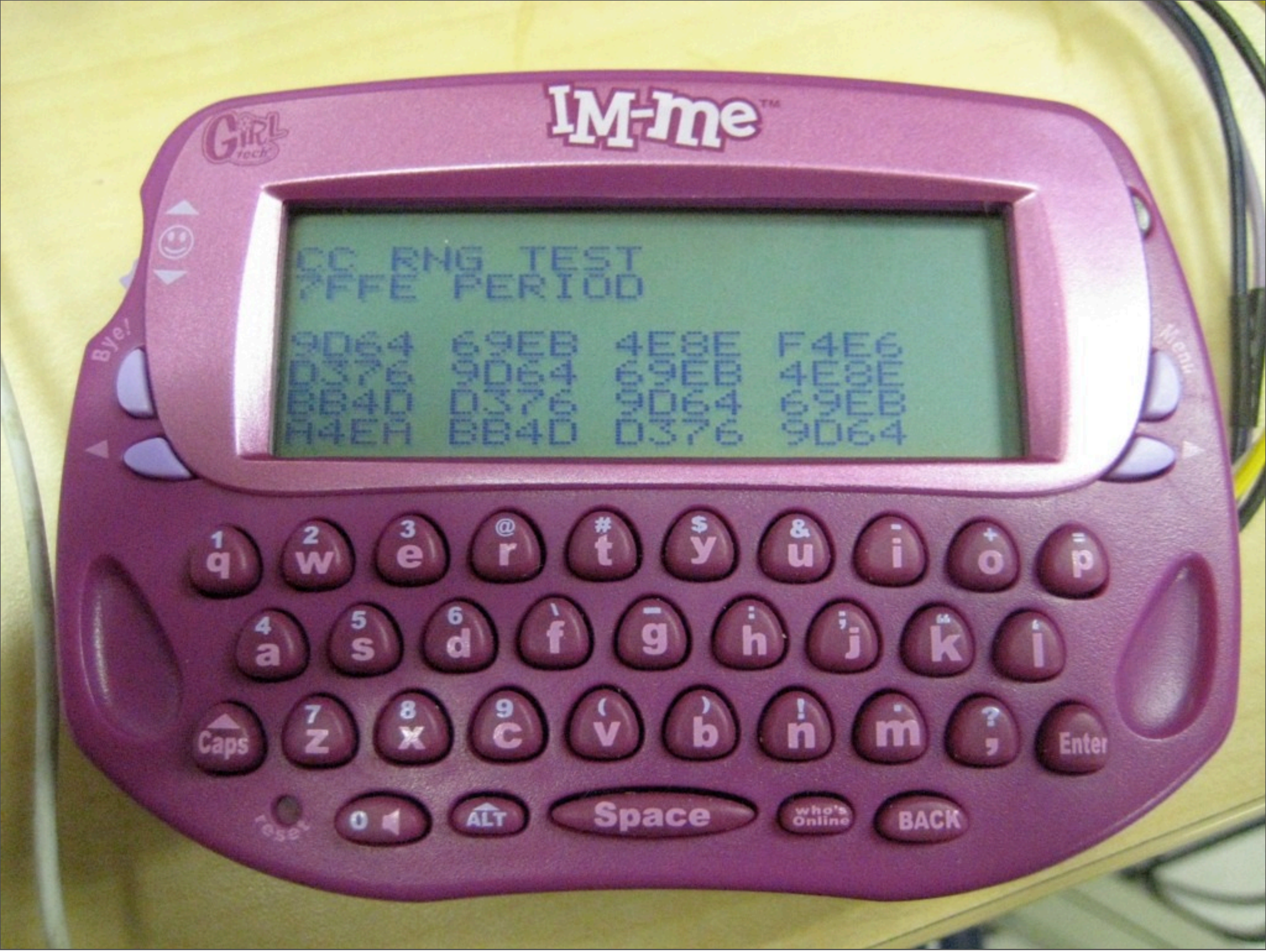


```
air% hexdump random.bin | grep --color "7c e1 e8 4e f4 87"
0000000 02 01 00 60 e8 2e 7c e1 e8 4e f4 87 62 49 56 fe
0008000 01 00 60 e8 2e 7c e1 e8 4e f4 87 62 49 56 fe 80
0010000 00 60 e8 2e 7c e1 e8 4e f4 87 62 49 56 fe 80 00
0018000 60 e8 2e 7c e1 e8 4e f4 87 62 49 56 fe 80 00 60
air%
```

# The ZStack PRNG Vulnerability

Travis Goodspeed, <travis@radiantmachines.com>

*March 11, 2010*



```
CC RNG TEST
7FFE PERIOD

9D64 69EB 4E8E F4E6
D376 9D64 69EB 4E8E
BB4D D376 9D64 69EB
A4EA BB4D D376 9D64
```

# Disclaimer

- This is a lecture about a specific vulnerability, not a survey of vulnerabilities in general.

- Chipcon's ZStack implementation of the ZigBee Smart Energy Profile is the subject of this lecture, but it should not be assumed that competing products do not have similar problems.

# Outline

- The PRNG method of key extraction.
- Other Methods
- Repair

# The vulnerability itself.

- PRNG is 16-bit LFSR
- Ephemeral Key is Predictable
    - Static Key is Random
- ECMQV is Vulnerable
    - If the Ephemeral Key is Known, then
    - the Static Key can be extracted.

# ECQMV

- Formerly part of NSA Suite B.
- Responsible for key validation AND shared secret generation.
    - In a single pass, saving computation time.
    - In a single pass, reducing security under some circumstances.
- Sold by Certicom, might be patented.

# ECQMV Variables

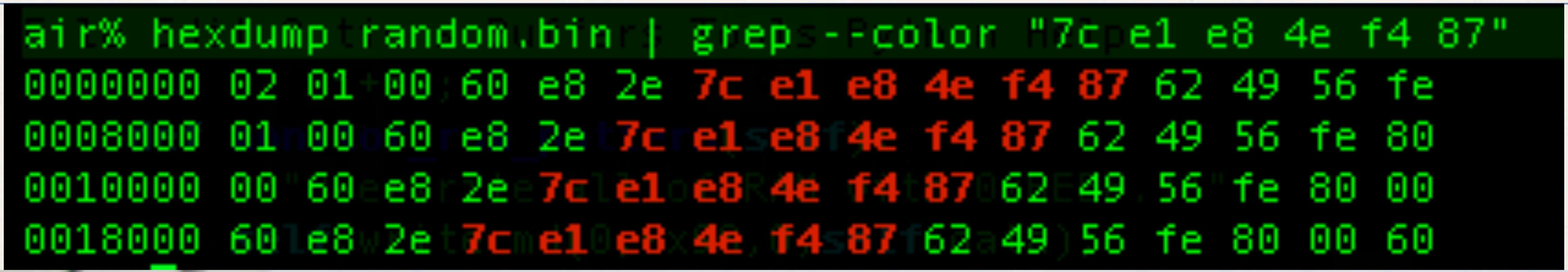
- For Alice and Bob,
  - Static Key Pair
  - Ephemeral Key Pair
- Alice Knows Bob's
  - Static Public Key
  - Ephemeral Public Key

- Public Static Keys are Signed by CA
- Ephemeral Keys come from PRNG
- Shared Secret is generated,
  - Used as AES key.
  - Always looks random.

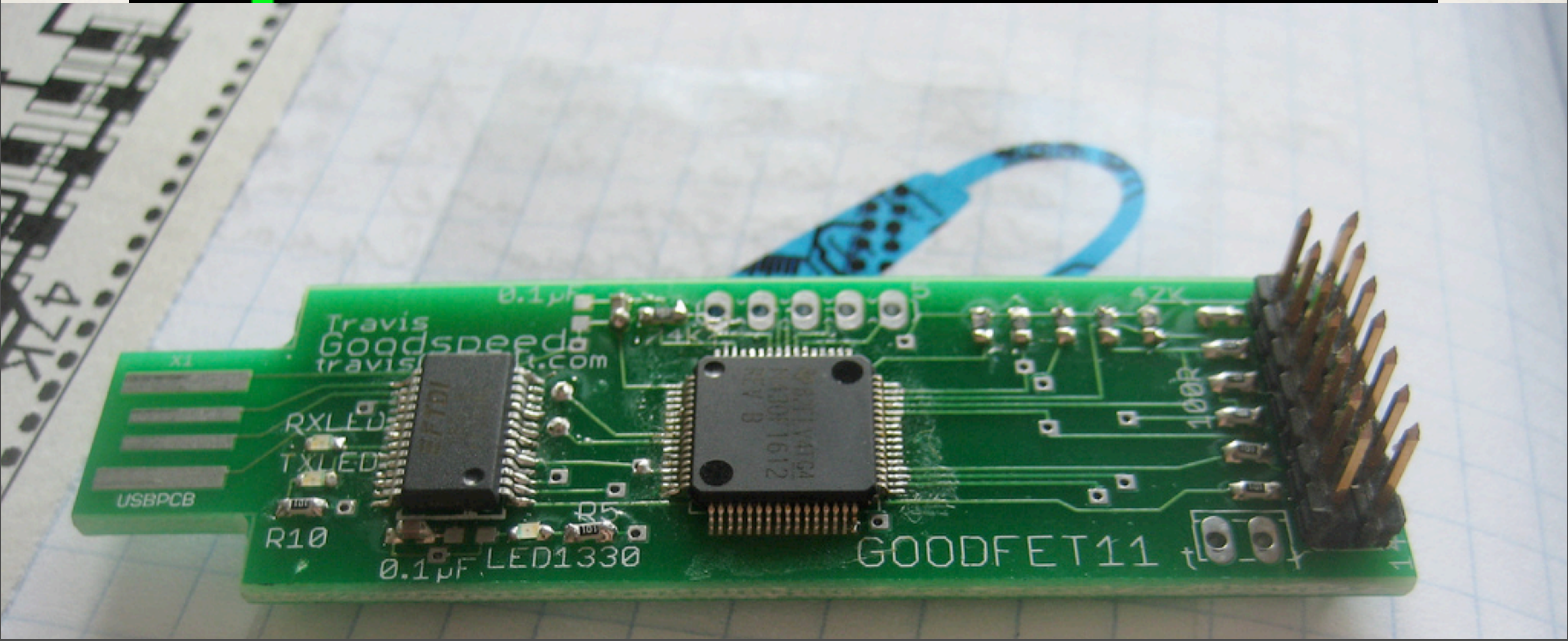
# ECMQV Break

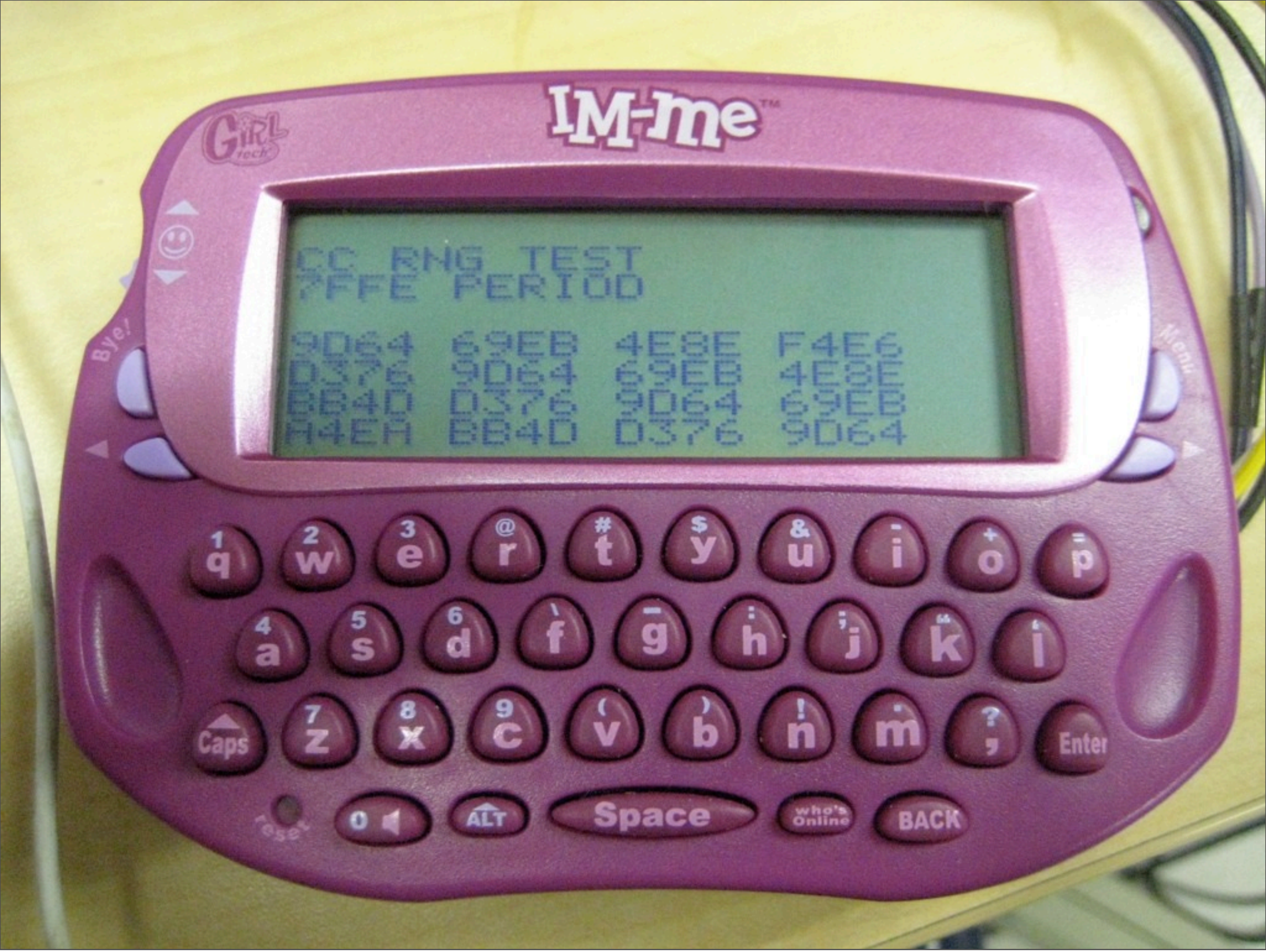
- ``Analysis of the Insecurity of ECMQV with Partially Known Nonces”
  - Peter Leadbitter and Nigel Smart, ISC 2003
- ``We have shown MQV to be insecure with keys of size less than $2^{320}$ if the attacker is able to obtain a small number of bits of each ephemeral secret generated by his victim. For a standard keylength of $2^{160}$, the 4 most significant bits were sufficient to recover the key.”

# PRNG Quality



```
air% hexdump random.bin | grep --color "7c e1 e8 4e f4 87"
0000000 02 01 00 60 e8 2e 7c e1 e8 4e f4 87 62 49 56 fe
0008000 01 00 60 e8 2e 7c e1 e8 4e f4 87 62 49 56 fe 80
0010000 00 60 e8 2e 7c e1 e8 4e f4 87 62 49 56 fe 80 00
0018000 60 e8 2e 7c e1 e8 4e f4 87 62 49 56 fe 80 00 60
```

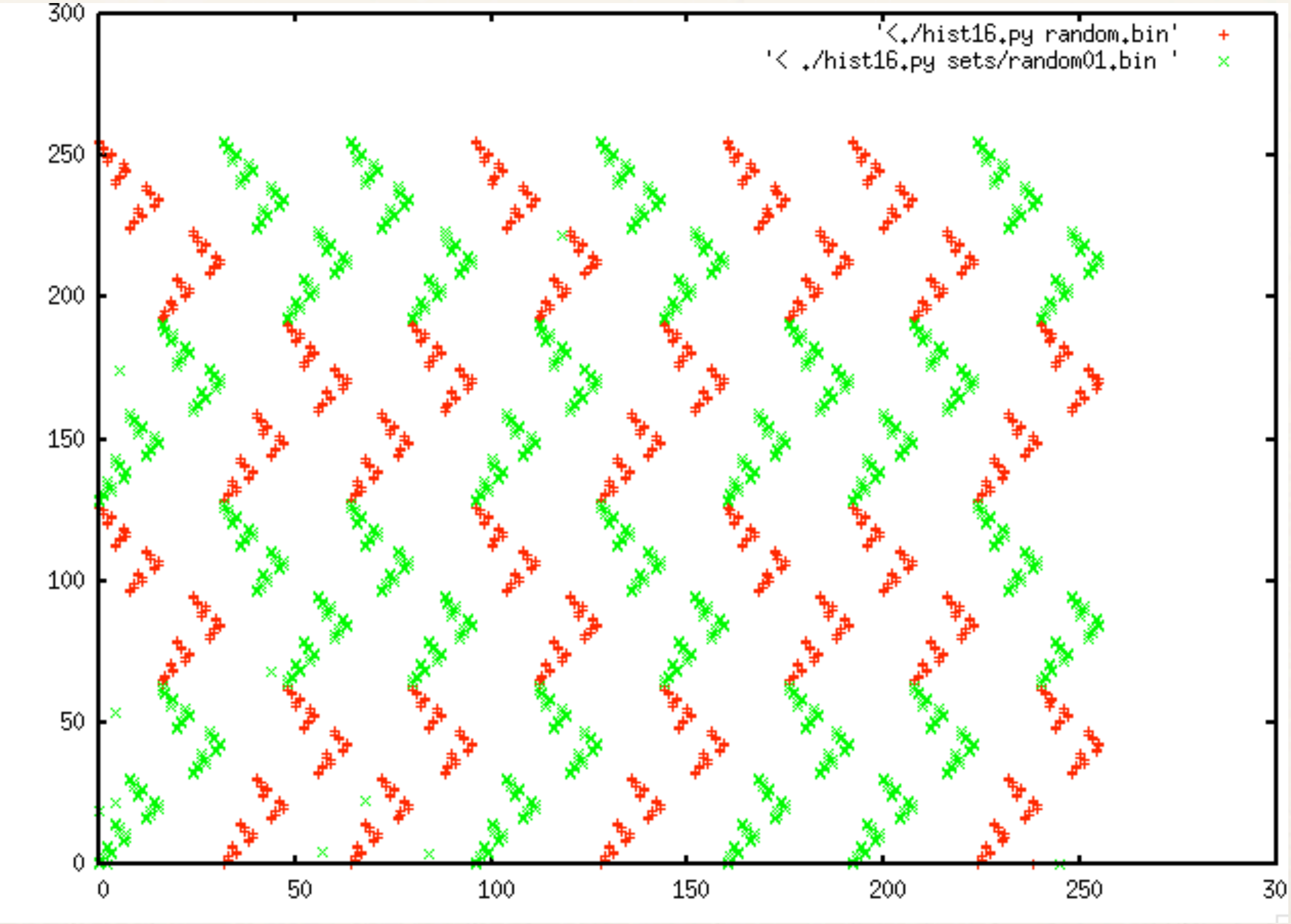




```
CC RNG TEST
7FFE PERIOD

9D64 69EB 4E8E F4E6
D376 9D64 69EB 4E8E
BB4D D376 9D64 69EB
A4EA BB4D D376 9D64
```

# PRNG Quality



Google for ``Strange Attractors"

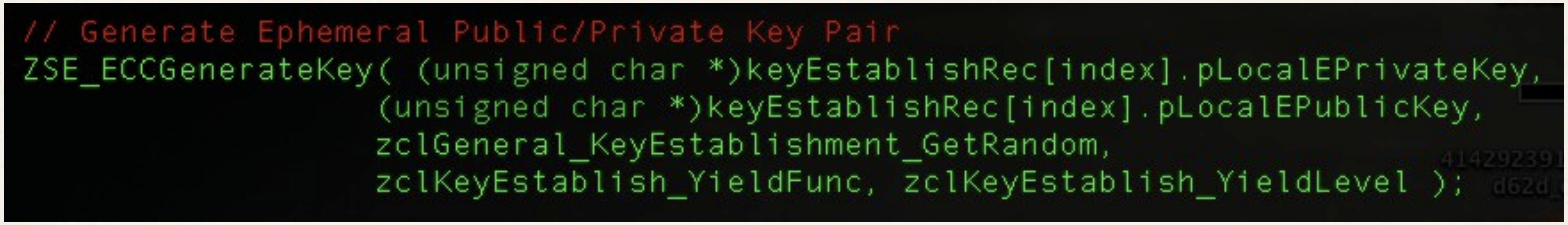
# PRNG Quality

- 16-bit LFSR is a very short sequence.
- Only half the states are exercised in any circuit.
- Only $2^{16}$ ephemeral keys will be generated.
  - 65,536 Ephemeral Key Pairs
  - Position exposed by random delays.
  - 20MB Lookup Table

# PRNG History

- CC1110, CC2430
  - Collision avoidance, random delay.
- CC2530, ZStack
  - ``The random-number generator uses a 16-bit LFSR to generate pseudorandom numbers, which can be read by the CPU or used directly by the command strobe processor. The random numbers can, e.g., be used to generate random keys used for security. ''
  - Fixed in Feb. 2010

# ZSE_ECCGenerateKey()



```
// Generate Ephemeral Public/Private Key Pair
ZSE_ECCGenerateKey( (unsigned char *)keyEstablishRec[index].pLocalEPrivateKey,
                    (unsigned char *)keyEstablishRec[index].pLocalEPublicKey,
                    zclGeneral_KeyEstablishment_GetRandom,
                    zclKeyEstablish_YieldFunc, zclKeyEstablish_YieldLevel );
```

- Zigbee Security Engine (ZSE) requires
    - Pointers to Ephemeral Keypair
    - GetRandom() function pointer

# Exploit

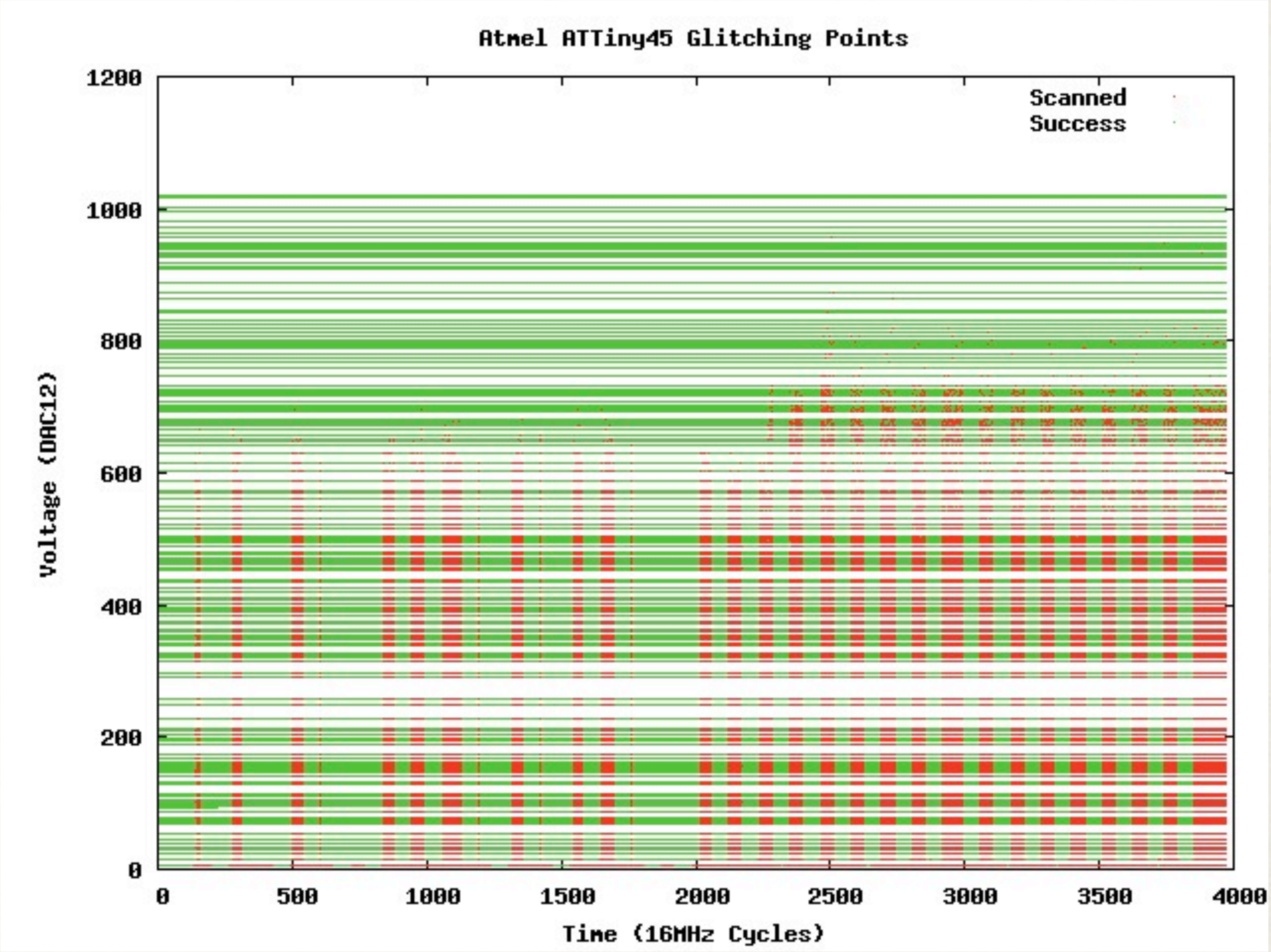
- Mallory authenticated with Alice.
- Alice provides public keys.
- Mallory’s key fails, Alice breaks authentication.
- Mallory looks up Alice’s Ephemeral Private Key.
- Mallory calculates Alice’s Static Private Key by Leadbitter technique.
- Mallory can now impersonate Alice.

# Static Key Extraction by Debugger

- ZStack stores keys in RAM.
- Chipcon 8051 devices defend Flash, not RAM.
- Exploit in two lines,
    - goodfet.cc erase
    - goodfet.cc dumpdata Foo.hex
- Foo.hex now contains a complete image of RAM.

# Key Extraction by VCC Glitch

- Drop VCC Briefly
- Interesting failures.



# Patching Firmware

- ZStack was patched in February of 2010.
- Applications using it must also be patched,
  - by the Application vendor,
  - for every project.
- Patching without vendor cooperation requires reverse engineering.

# Conclusions

- This problem has not yet been fixed.
- Crypto is hard, and it shouldn't be assumed to work.
- One vulnerable device on a network is sufficient.

Travis Goodspeed
tgo@seas.upenn.edu