CS 380S

0x1A Great Papers in Computer Security

Vitaly Shmatikov

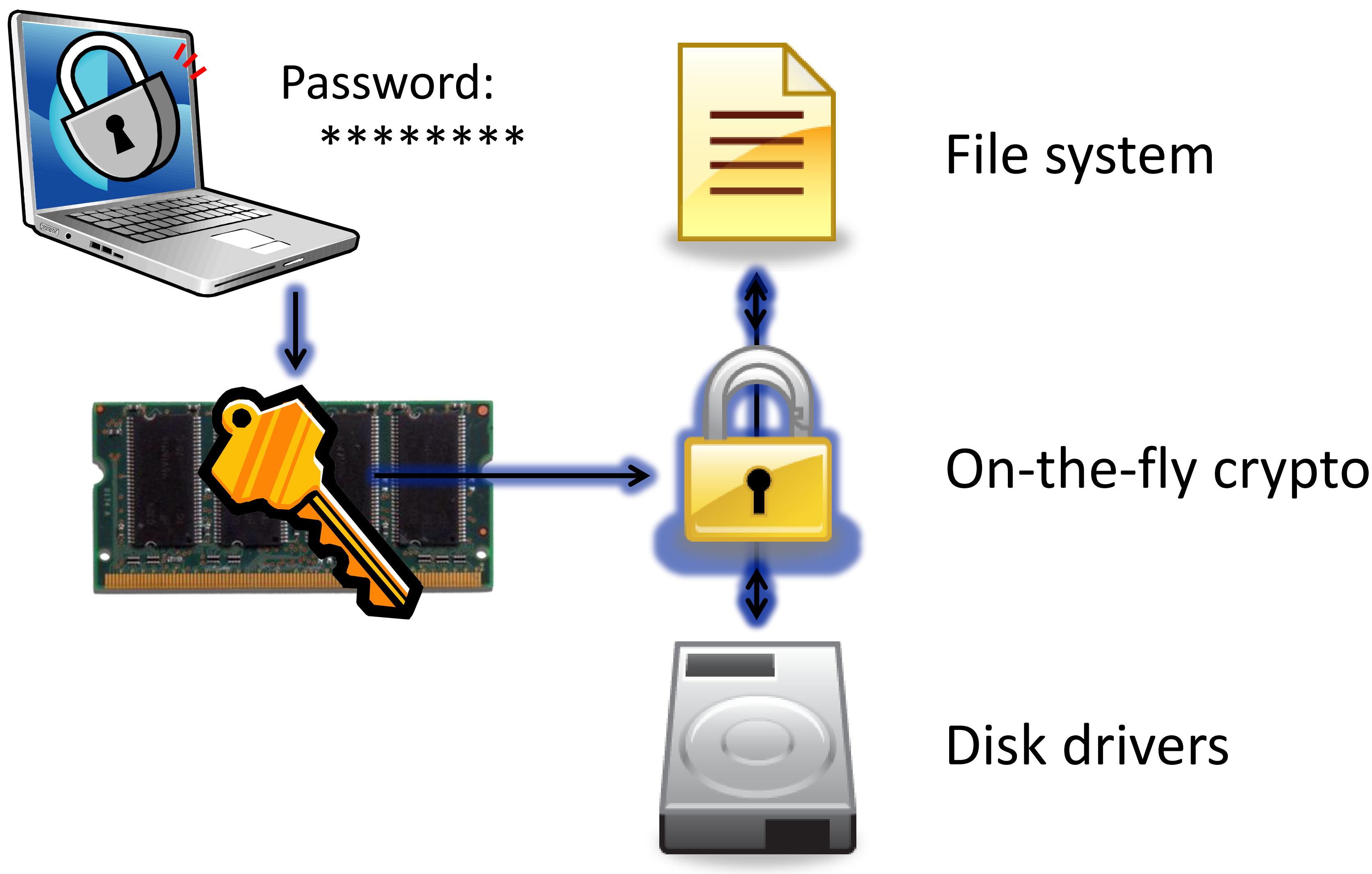
http://www.cs.utexas.edu/~shmat/courses/cs380s/

J. Alex Halderman et al.

Lest We Remember: Cold Boot Attacks on Encryption Keys

(USENIX Security 2008)

Protecting Data on a Laptop

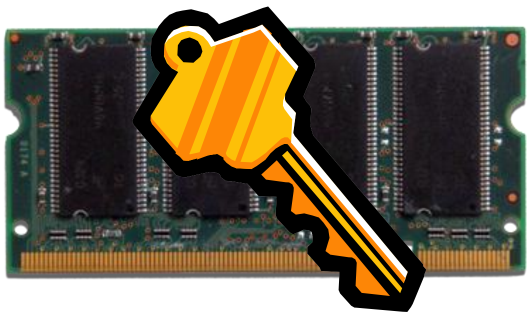


Common Attack Scenario

Security Assumptions

The encryption is strong

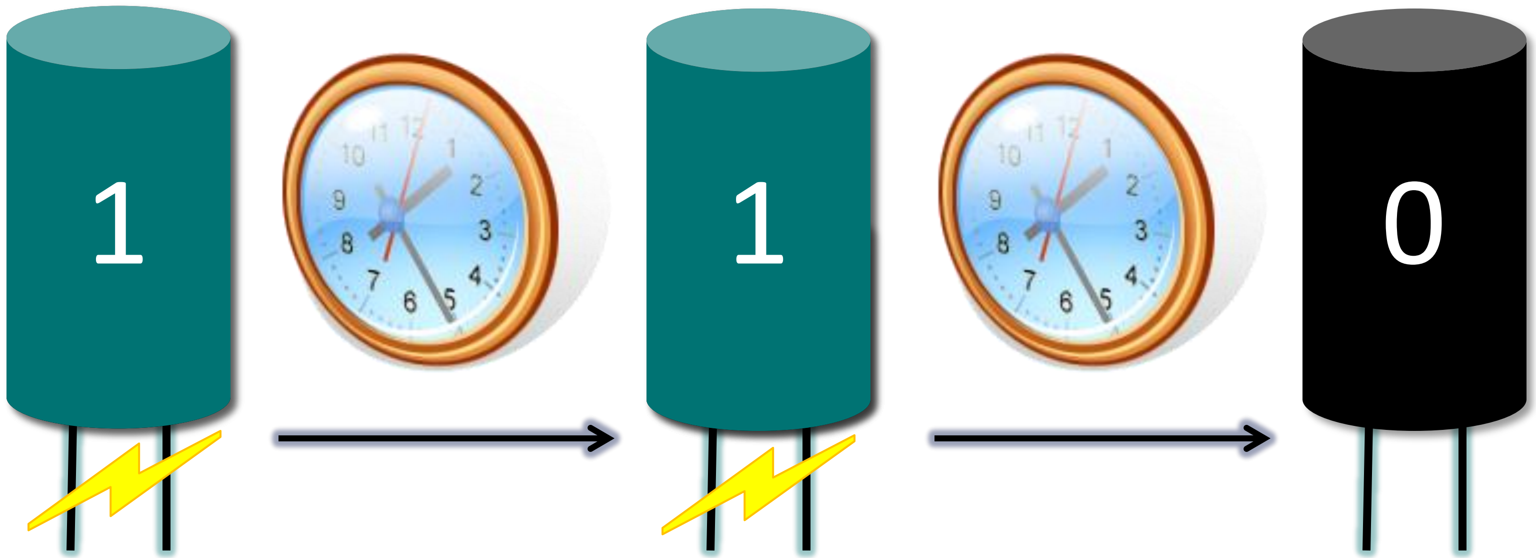
The OS protects the key in RAM



…the attacker might reboot to circumvent the OS, but since RAM is volatile, the key will be lost…

Dynamic RAM Volatility

DRAM cell
(capacitor)



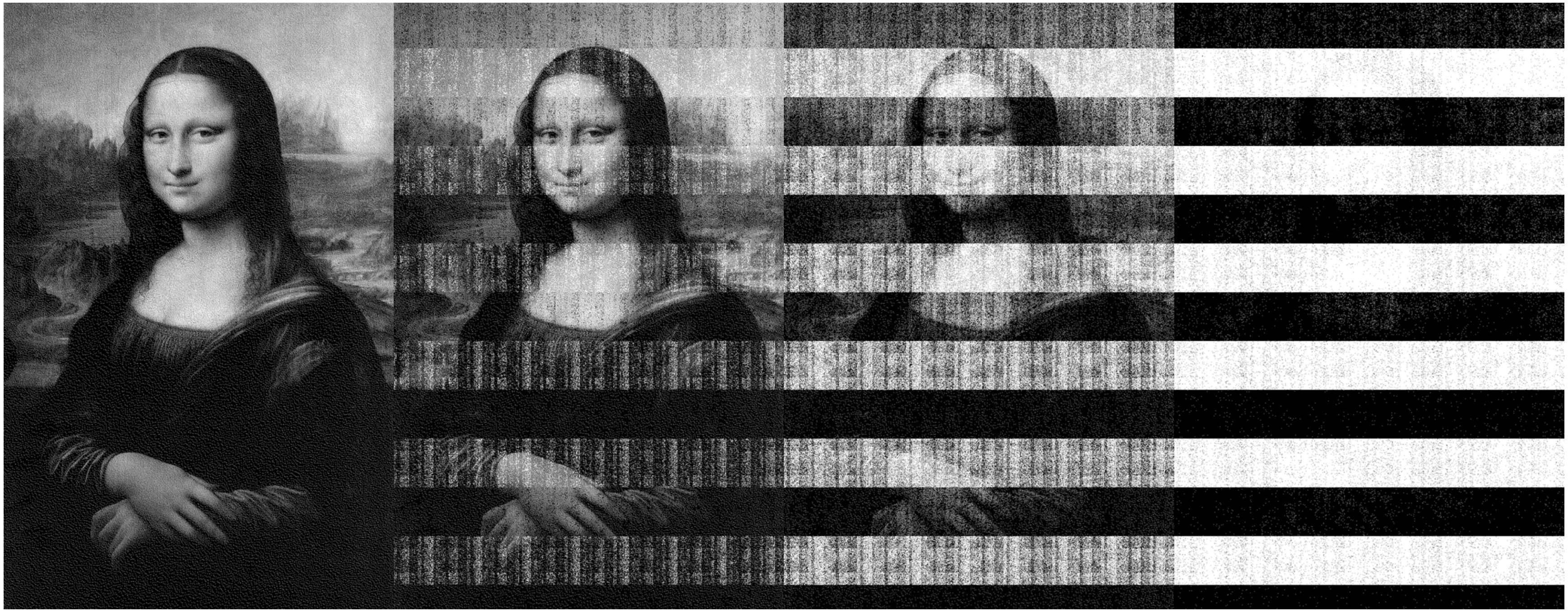
Write "1"

Refresh (read and rewrite)

Refresh interval ≈ 32 ms

What if we don't refresh?

Decay After Cutting Power

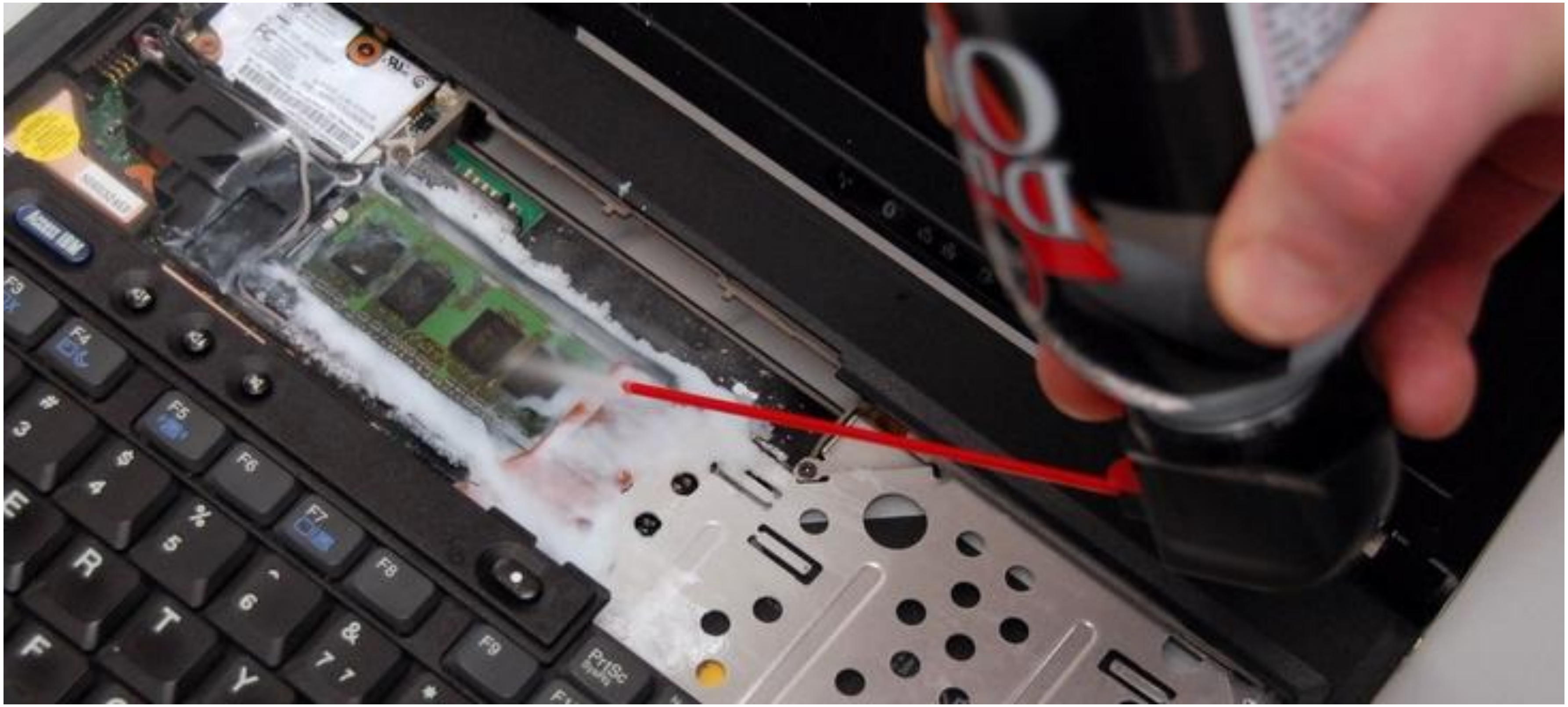


5 secs | 30 secs | 60 secs | 300 secs

Capturing Residual Data

- No special equipment needed, but …
- … booting OS overwrites large areas of RAM
- Solution: boot a small low-level program to dump out memory contents
  - PXE dump (9 KB)
  - EFI dump (10 KB)
  - USB dump (22 KB)
- What if BIOS clears RAM?
  - Common on systems with error-corrected RAM

Slowing Decay By Cooling



-50°C

< 0.2% decay after **1 minute**


Esc
F1
F2
F3
Tab
START/STOP
SET
MODE
SPLIT/RESET
SELECT
SUN MON TUE WED THU FRI SAT
6:48 29
MIN
SEC

Even Cooler

Liquid nitrogen -196°C

< 0.17% decay after **1 hour**

Not necessary in practice

Dealing with Bit Errors

Some bit errors inevitable, especially without cooling (increasing with memory density)

Naïve Approach

Given corrupted K’, find K:

Brute-force search over low Hamming distance to K’

e.g. 256-bit key with 10% error:

> 2^{56} guesses (too slow!)

Insight

Most programs store precomputed derivatives of K (e.g. key schedules)

These derivatives contain redundancy; we can treat them as error correcting codes

Correcting Bit Errors in DES

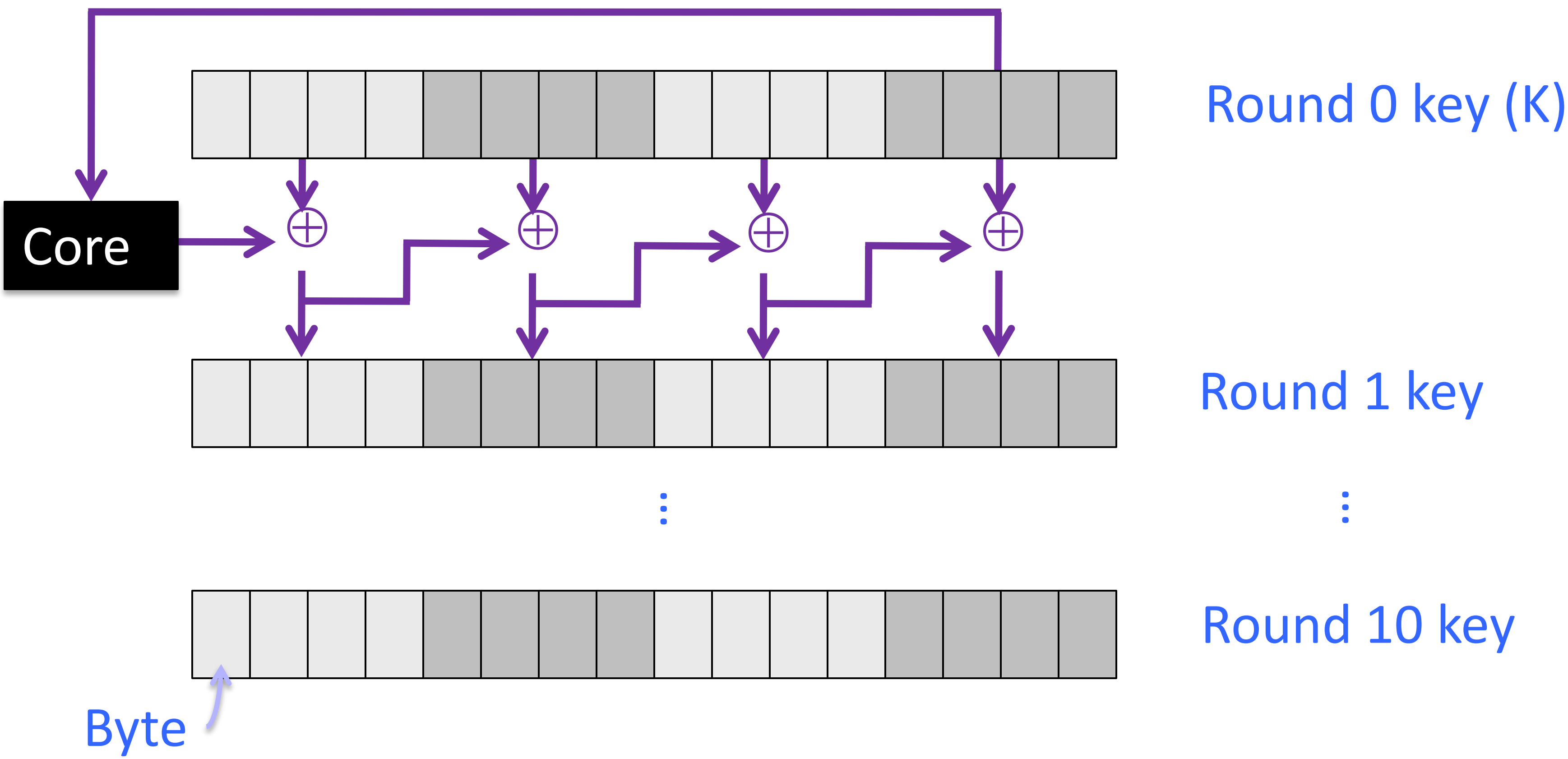
Key schedule contains ~14 redundant copies of each bit from the key

0100000010000100 ~20% error

0000000000000000 0

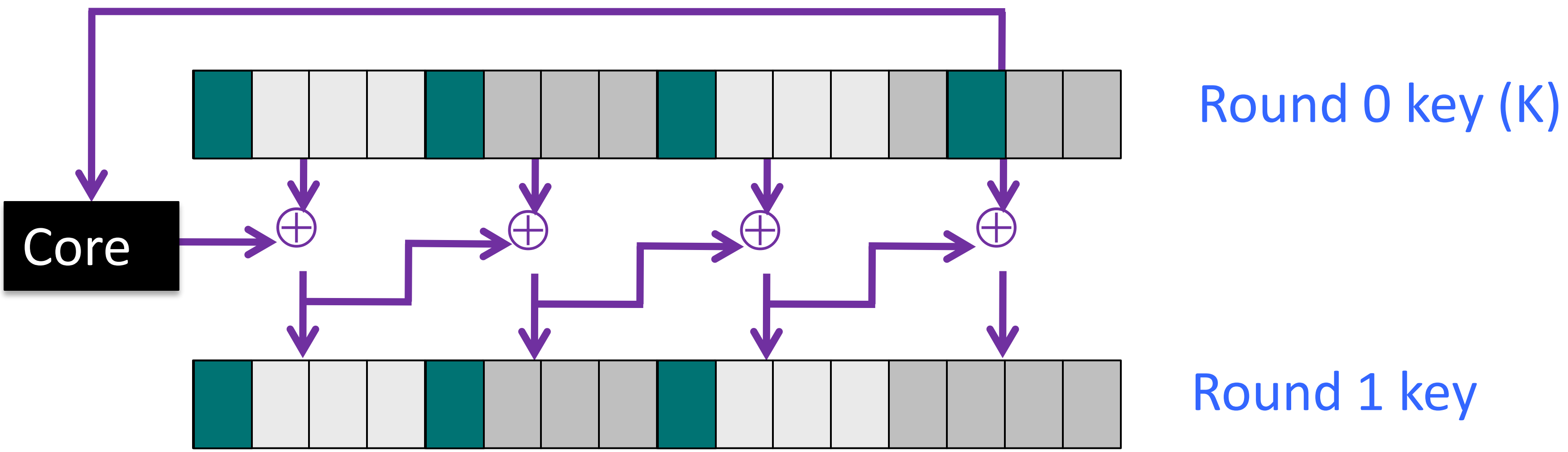
AES Key Schedule

128-bit key K → 10 more 128-bit keys for cipher rounds



Correcting Bit Errors in AES (1)

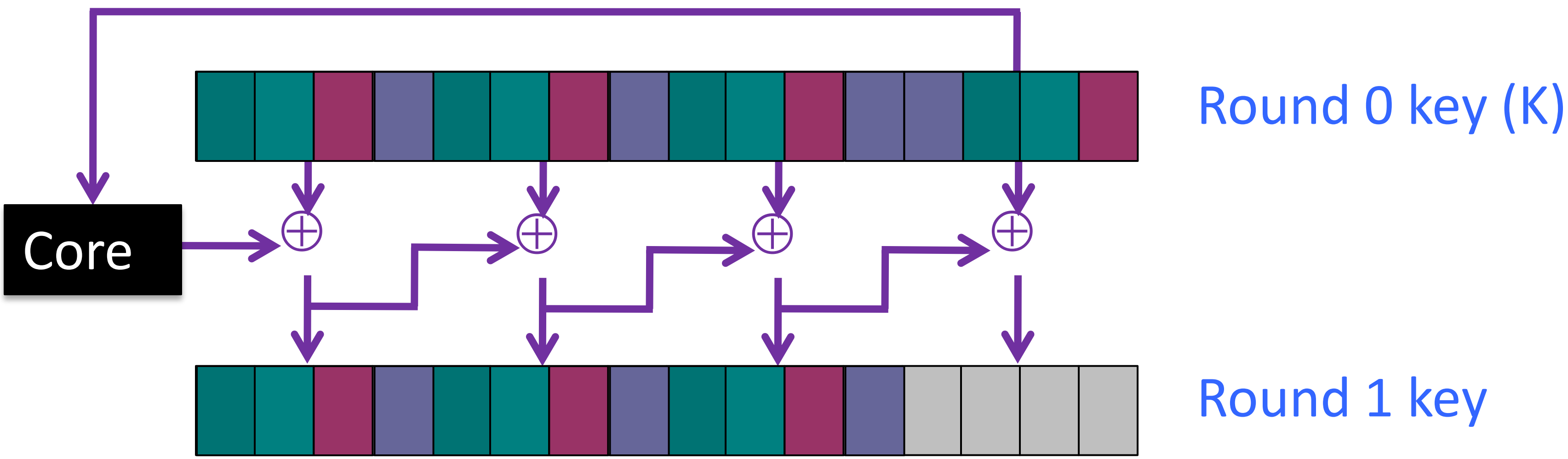
Key schedule recovered from memory (contains errors)



1. Slices: 4 bytes in Round 0 determine 3 bytes in Round 1
2. Enumerate 2^{32} possibilities for each 7 byte slice
3. Eliminate values unlikely to have decayed to observed bytes (excludes vast majority)

Correcting Bit Errors in AES (2)

Key schedule recovered from memory (contains errors)



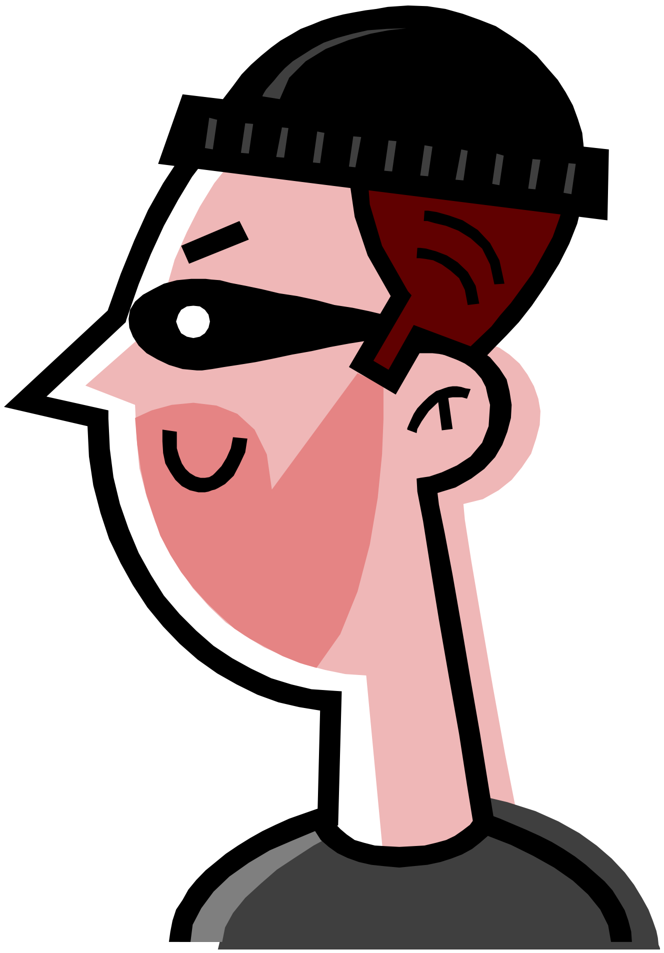
4. Repeat for each of the 4 slices
5. Combine possible slice values into candidate keys
6. Test candidates keys by expanding them into full key schedules – compare to recovered memory

Finding AES Key Schedules

- Iterate through each byte of memory
- Treat following region as an AES key schedule
- For each word in the candidate "schedule" …
  - Calculate correct value, assuming other bytes correct
  - Take Hamming distance to observed value
- If total distance is low, output the key

Demonstrated Attacks

- Windows BitLocker
- Mac OS FileVault
- Linux dm-crypt
- Linux LoopAES
- TrueCrypt



Countermeasures

- Encrypt key in memory when screen-locked
- Avoid precomputation
- Fully encrypted memory
- Trusted Platform Module (TPM)

Read paper for discussion