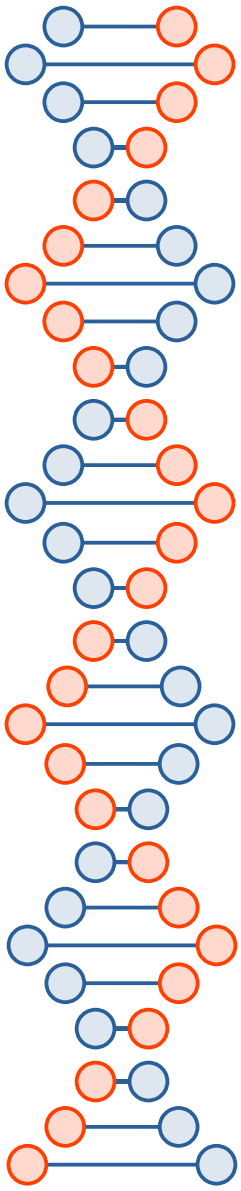
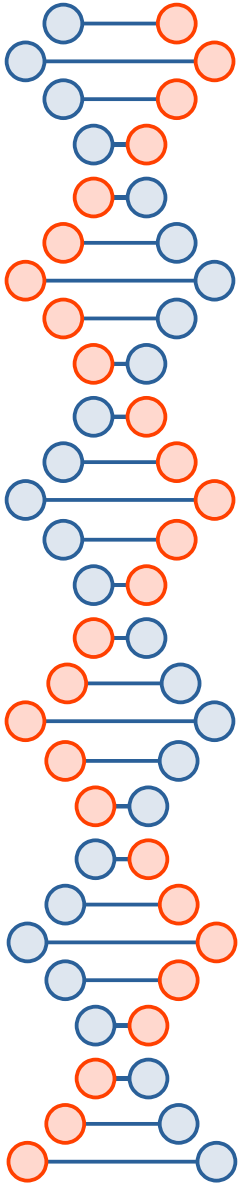


Timing and side channel attacks

CSE 539

jedimaestro@asu.edu



Outline

- What is a side channel?
- Differences in computation time
- Differences in power consumption
- Differences in cache behavior

A side channel

https://commons.wikimedia.org/wiki/File:Domino%27s_Pizza_delivery_scooter_in_Southern_Taiwan_20070220.jpg

According to Wikipedia

WIKIPEDIA
The Free Encyclopedia

Search Wikipedia

Creat

Side-channel attack

16 languages

ents [hide]

ples
termeasures
lso
ences
er reading
ks
les

Article Talk

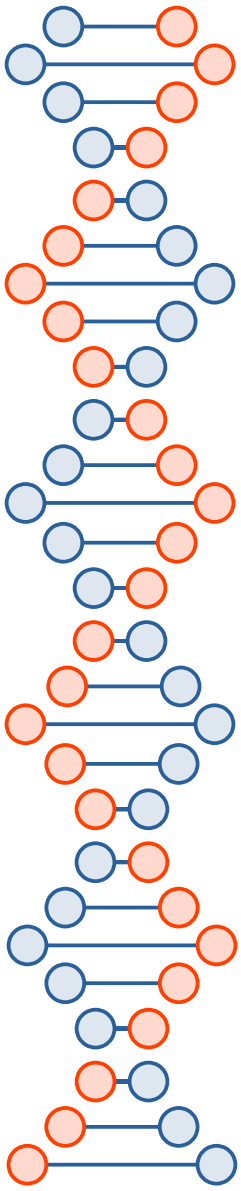
Read Edit View history

From Wikipedia, the free encyclopedia

"Side channels" redirects here. For the yoga concept, see Nadi (yoga) § Side channels. For interactive television services, see Red Button (digital television).

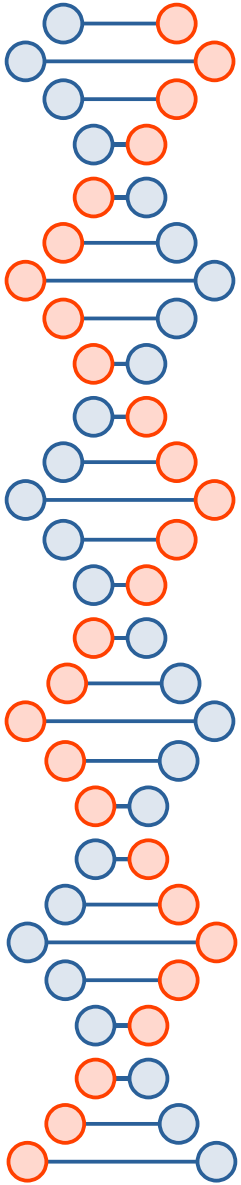
In computer security, a **side-channel attack** is any attack based on extra information that can be gathered because of the fundamental way a computer protocol or algorithm is implemented, rather than flaws in the design of the protocol or algorithm itself (e.g. flaws found in a cryptanalysis of a cryptographic algorithm) or minor, but potentially devastating, mistakes or oversights in the implementation. (Cryptanalysis also includes searching for side-channel attacks.) Timing information, power consumption, electromagnetic leaks, and sound are examples of extra information which could be exploited to facilitate side-channel attacks.

An attempt to decode RSA key bits using power analysis. The left peak represents the CPU power variations during the step of the algorithm without

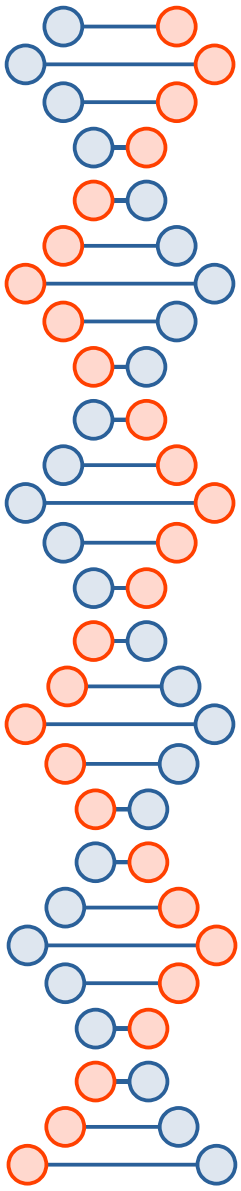


How I prefer to think about side channels...

- A covert channel is “not intended for information transfer at all” (Lampson, B.W., A Note on the Confinement Problem. Communications of the ACM, Oct.1973.16(10):p. 613-615.).
- A side channel, unlike a traditional covert channel, does not assume any collusion between sender and receiver. Rather, the sender is leaking the information on accident or is tricked into doing so.



Differences in computation time...



ELSEVIER

Available online at www.sciencedirect.com

SCIENCE DIRECT®

Computer Networks 48 (2005) 701–716

Computer Networks

www.elsevier.com/locate/comnet

Remote timing attacks are practical

David Brumley [a,*], Dan Boneh [b]

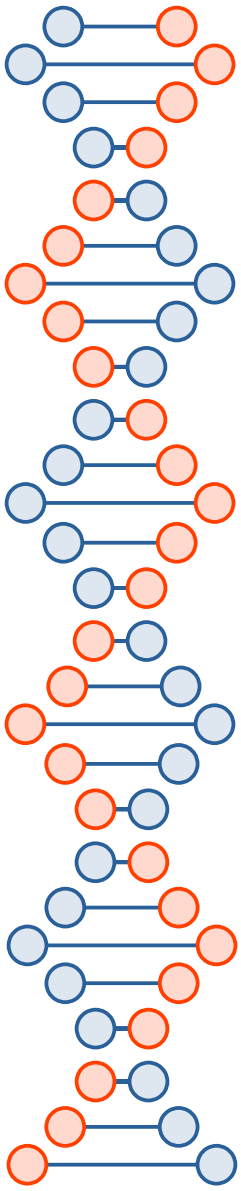
[a] *Carnegie Mellon University, 5000 Forbes Ave, Wean Hall # 8116, Pittsburgh, PA 15213, USA*
[b] *Computer Science Department, Stanford University, Gates 475, Stanford, CA 94305, USA*

Available online 11 March 2005

Abstract

Timing attacks are usually used to attack weak computing devices such as smartcards. We show that timing attacks apply to general software systems. Specifically, we devise a timing attack against OpenSSL. Our experiments show that we can extract private keys from an OpenSSL-based web server running on a machine in the local network. Our results demonstrate that timing attacks against network servers are practical and therefore security systems should defend against them.

Keywords: Timing attack; RSA; Chinese remainder; Montgomery reductions; SSL



https://en.wikipedia.org/wiki/Montgomery_modular_multiplication

```
function REDC is
    input: Integers R and N with gcd(R, N) = 1,
           Integer N′ in [0, R − 1] such that NN′ ≡ −1 mod R,
           Integer T in the range [0, RN − 1].
    output: Integer S in the range [0, N − 1] such that S ≡ TR^−1 mod N

    m ← ((T mod R)N′) mod R
    t ← (T + mN) / R
    if t ≥ N then
        return t − N
    else
        return t
    end if
end function
```

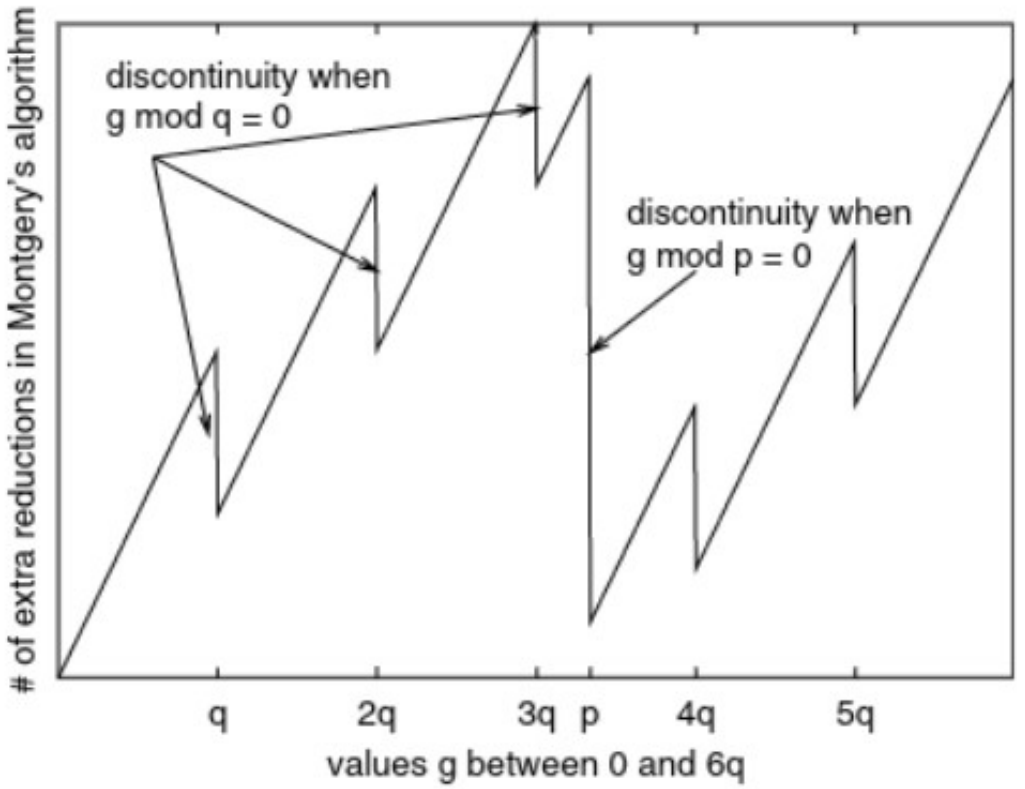
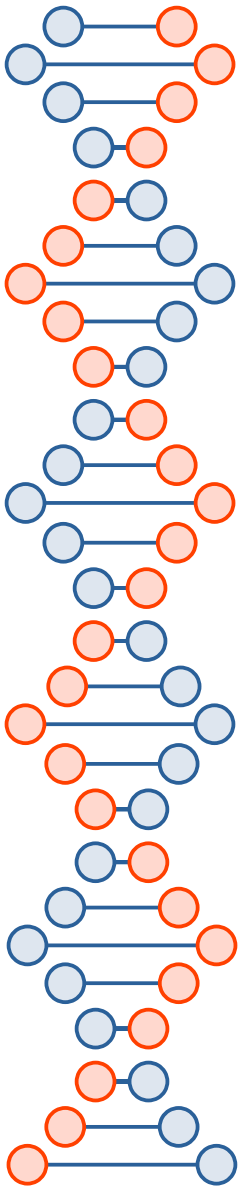


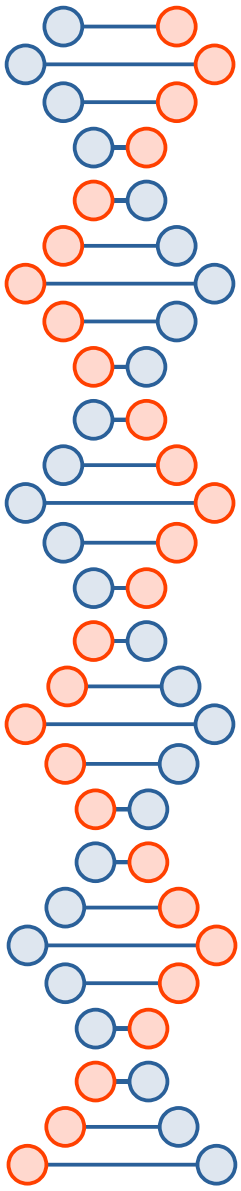
Fig. 1. Number of extra reductions in a Montgomery reduction as a function (equation 1) of the input g.



Dragonblood attacks on WPA3

- Slides plagiarized from…

  https://papers.mathyvanhoef.com/wac2019-slides.pdf



Convert password to MODP element

```
for (counter = 1; counter < 256; counter++)
    value = hash(pw, counter, addr1, addr2)
    if value >= p: continue
    P = value^((p-1)/q)
    return P
```

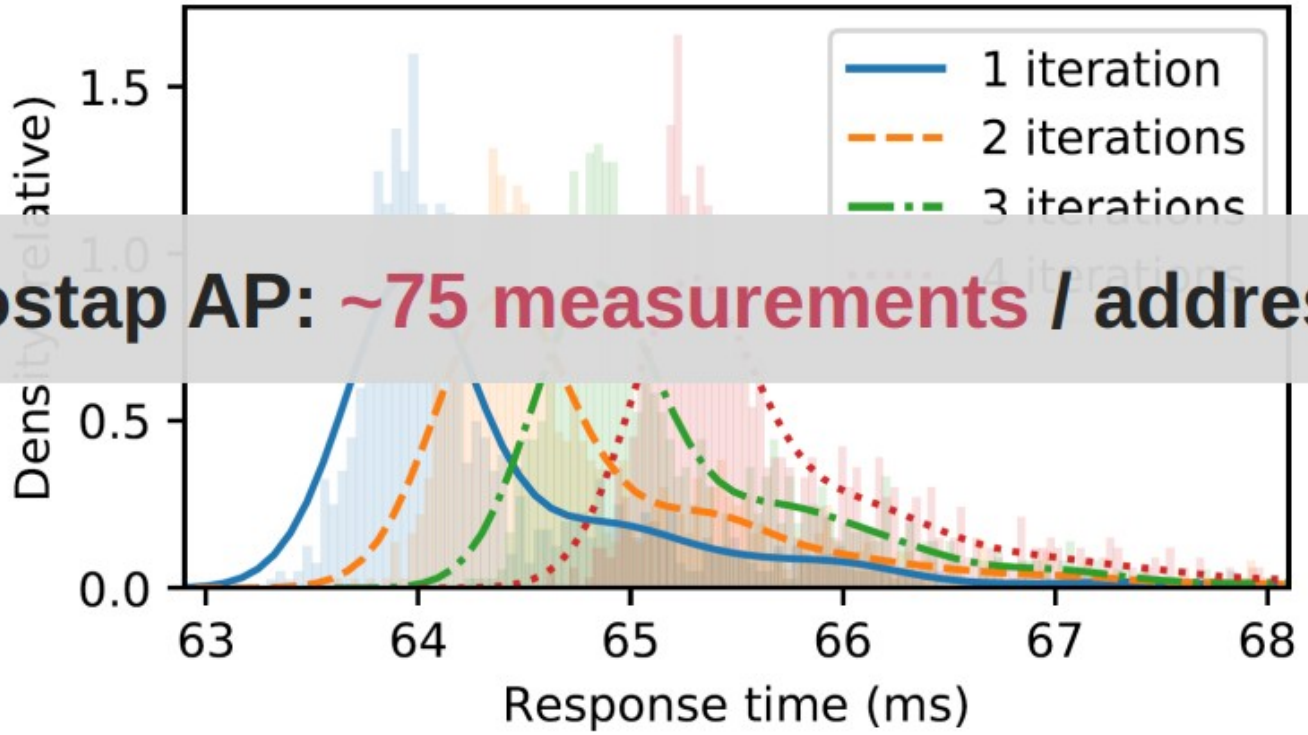
Leaked information: #iterations needed

Leaked information: #iterations needed

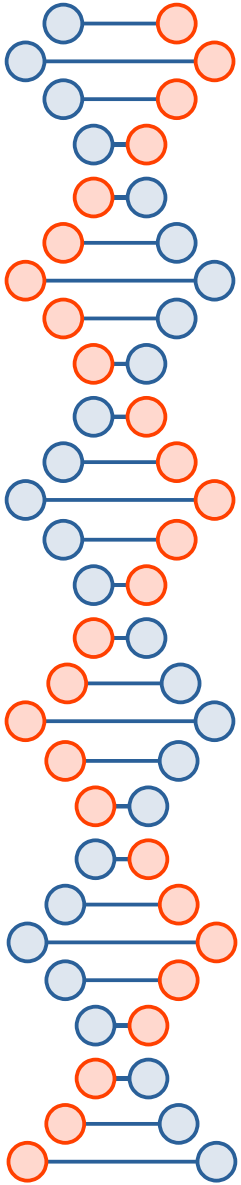
Forms a signature of the password

Need ~17 addresses to determine password in RockYou (~10^7) dump

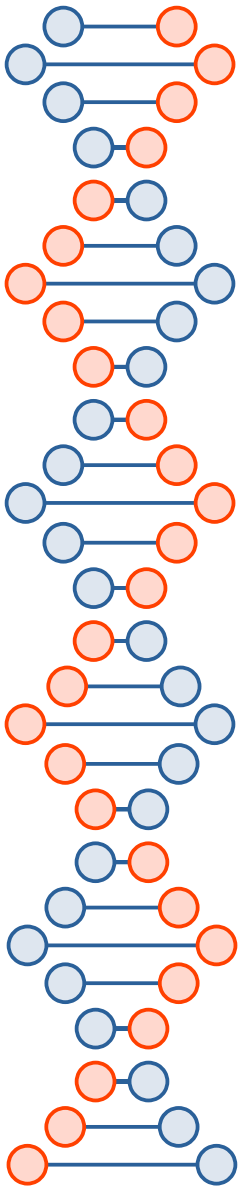
Raspberry Pi 1 B+: differences are measurable



Hostap AP: ~75 measurements / address



Differences in power consumption...



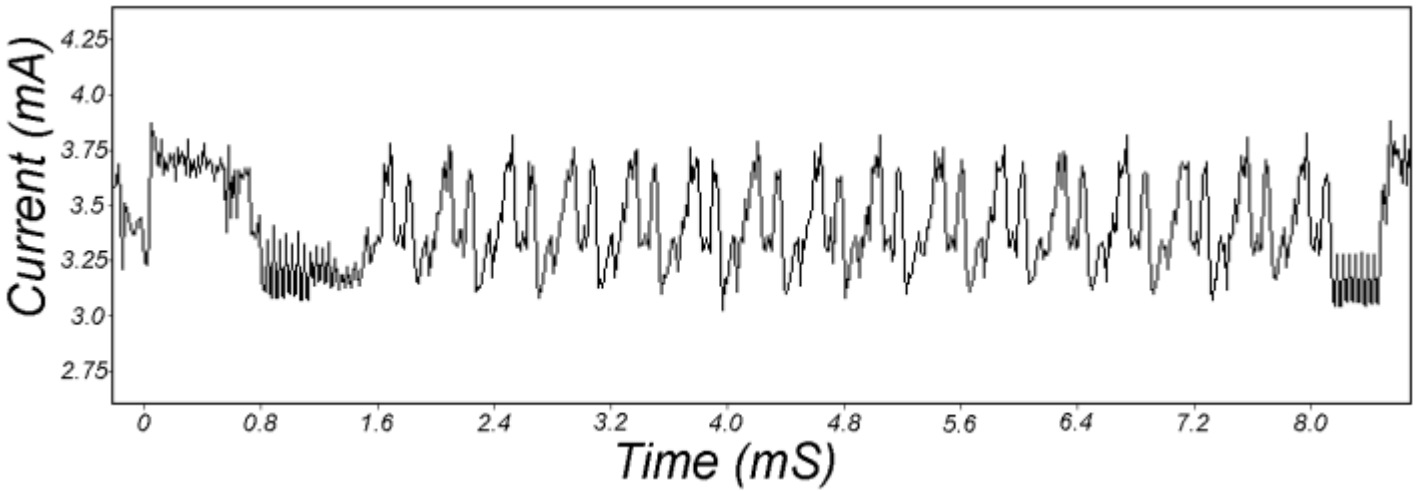
Differential Power Analysis

Paul Kocher, Joshua Jaffe, and Benjamin Jun

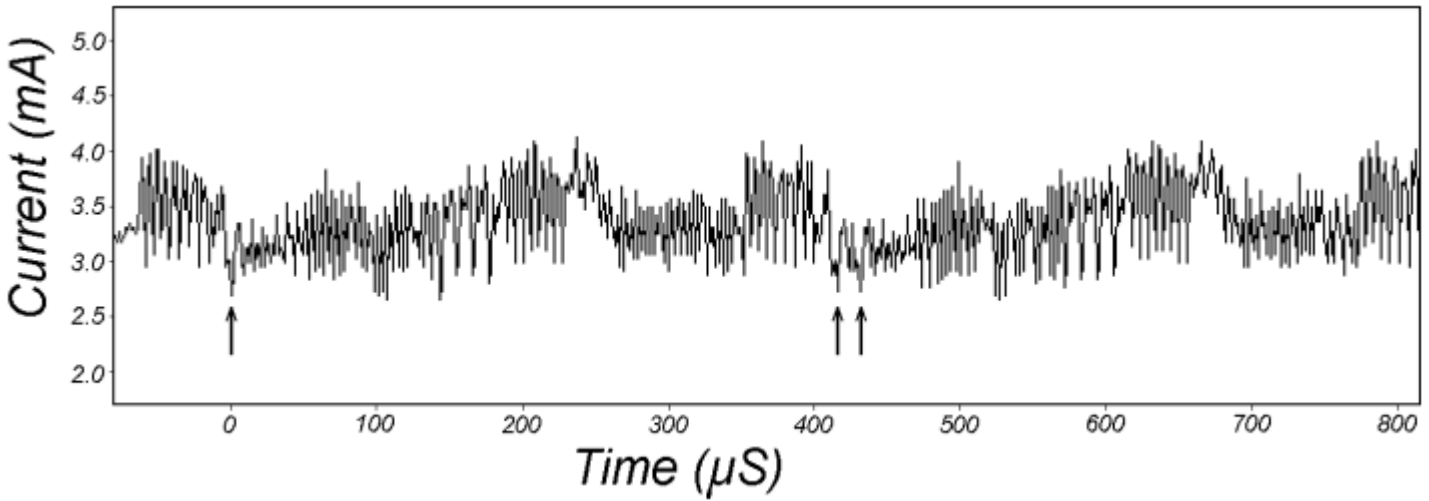
Cryptography Research, Inc.

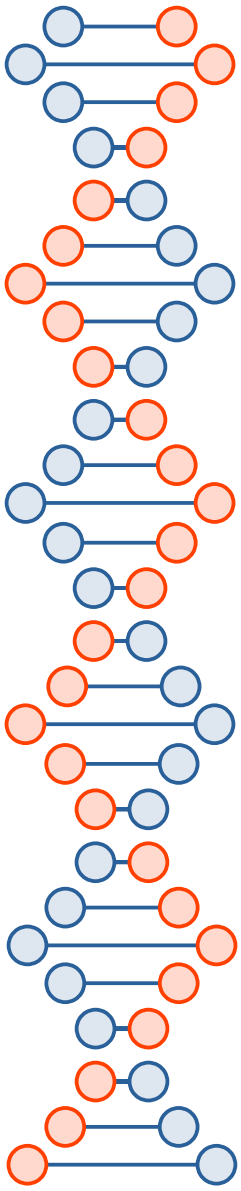
Michael Wiener (Ed.): CRYPTO'99, LNCS 1666, pp. 388-397, 1999.

16 rounds of DES (smart card)



Rounds 2 and 3 of DES





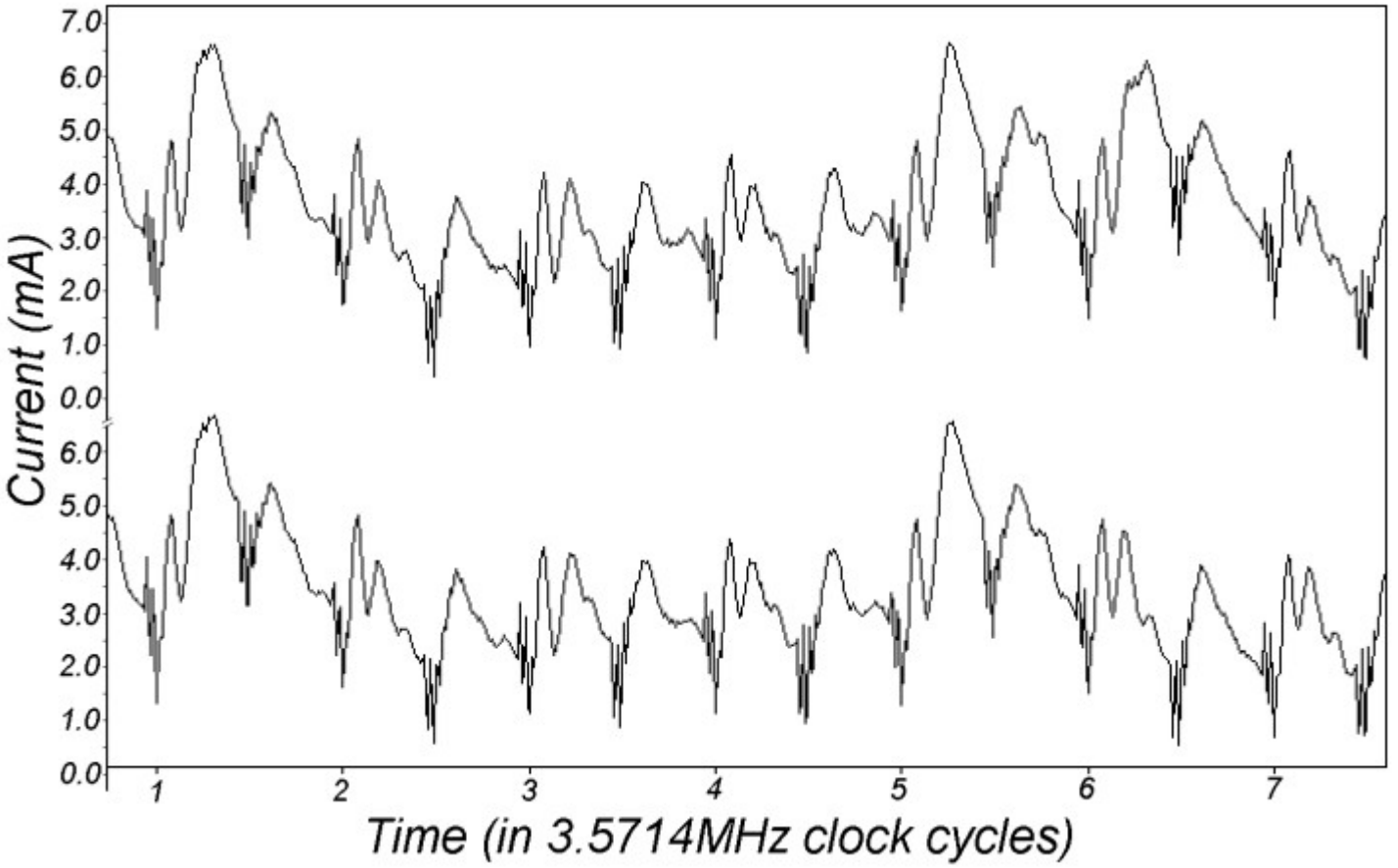
Any difference?

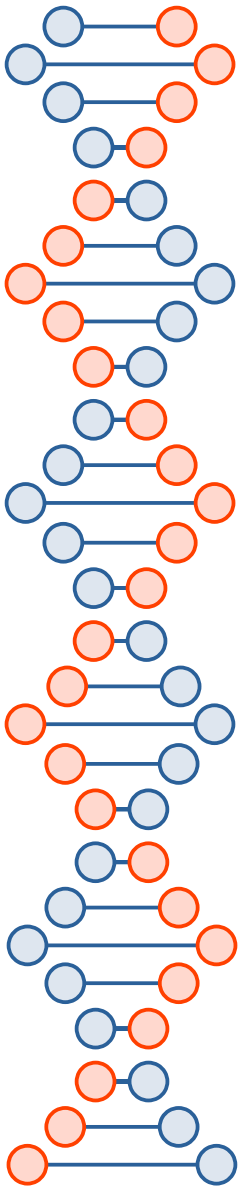
```
y = (x >> 1) | ((x & 1) << 27)
```

```
y = x >> 1
if (x & 1 == 1):
    y = y | (1 << 27)
```

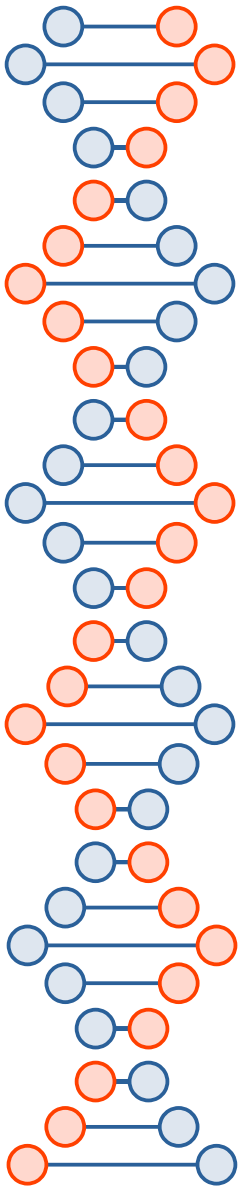
Note: this is a 28-bit rotation for DES key scheduling.

Individual clock cycles





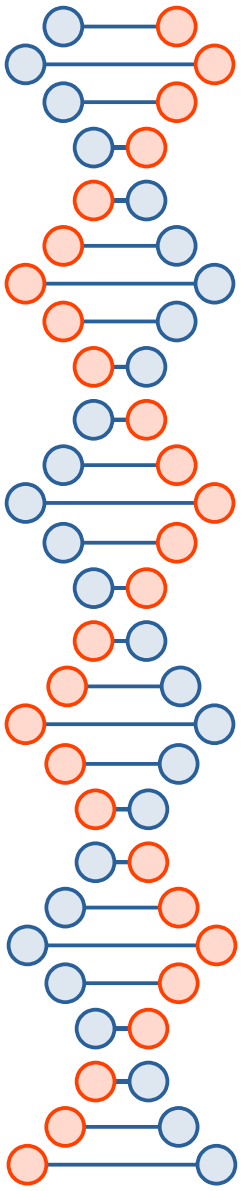
Differences in cache behavior...



Modular exponentiation

$153^{189} \pmod{251}$

Naive way: multiply 153 times itself 189 times.
Won't work for, *e.g.*, 2048-bit numbers,
especially for the exponent



Better way (all mod 251)

$153^{0} = 1$

$153^{1} = 153$

$153^{2} = 66$

$153^{4} = 89$

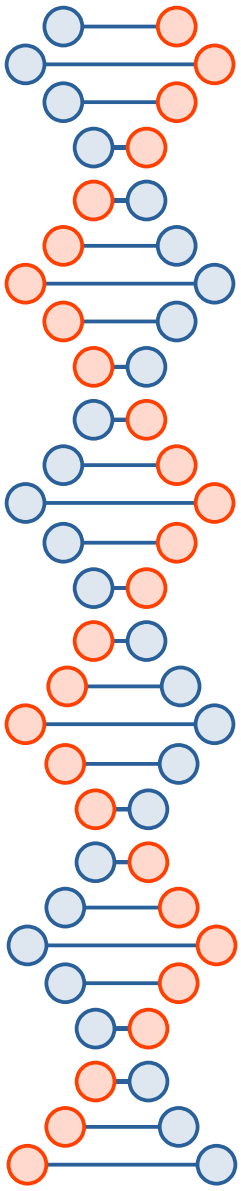
$153^{8} = 140$

$153^{16} = 22$

$153^{32} = 233$

$153^{64} = 73$

$153^{128} = 58$



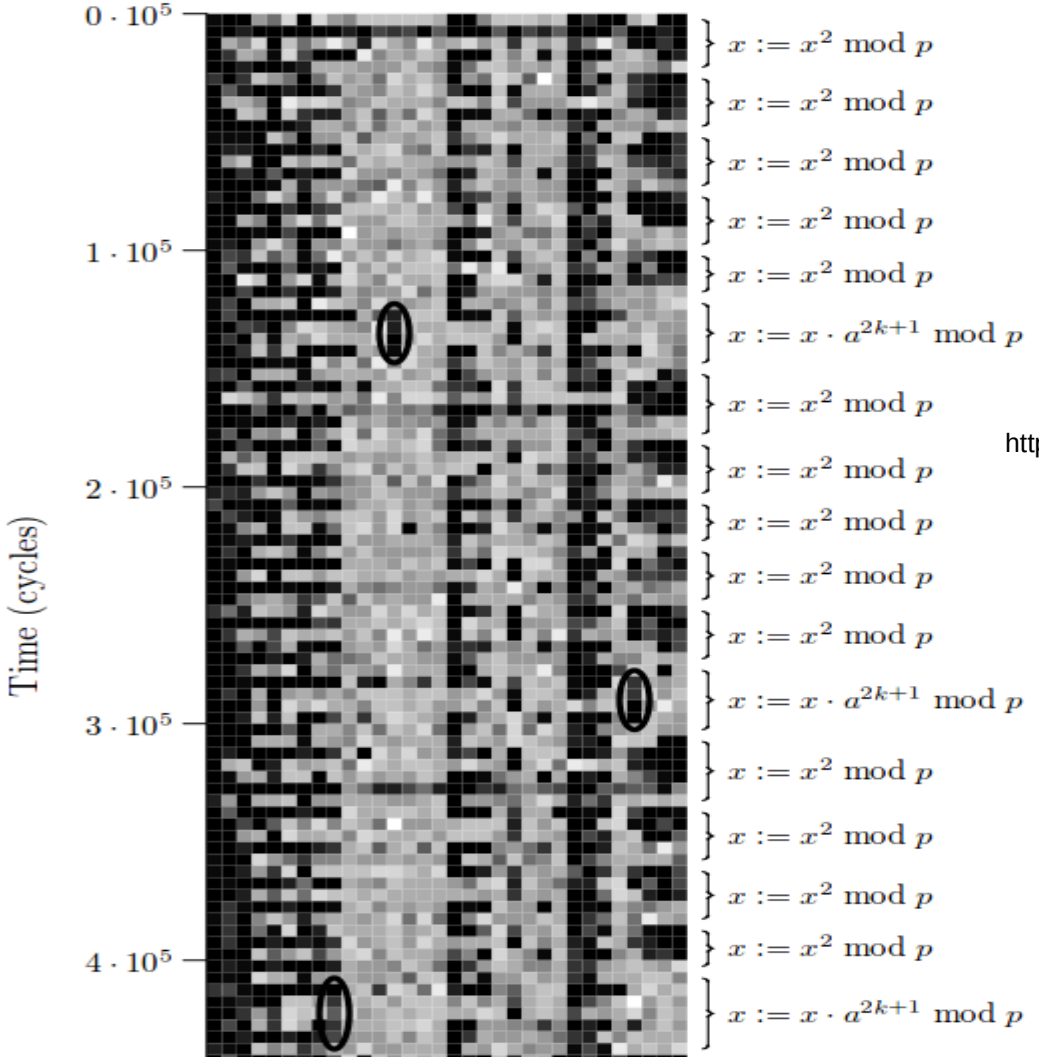
Better way

- 189 in binary is 0b10111101
- $189 = 1*2^7 + 0*2^6 + 1*2^5 + 1*2^4 + 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0$
- $153^{189} \pmod{251} = 153^{(128+0+32+16+8+4+0+1)} \pmod{251}$

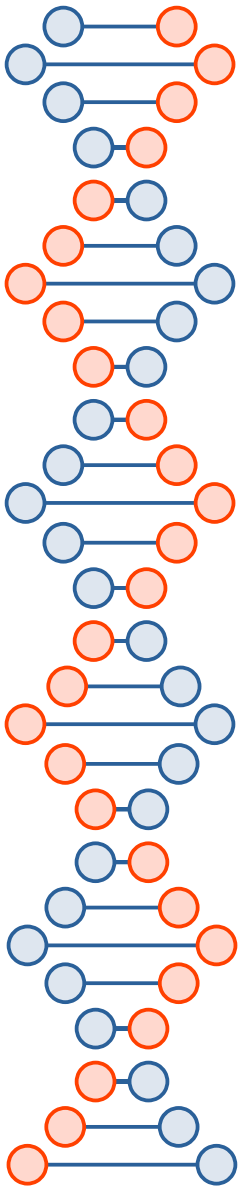
  $= 153^{128} * 153^{32} * 153^{16} * 153^{8} * 153^{4} * 153^{1} \pmod{251}$

  $= 58 * 233 * 22 * 140 * 89 * 153 \pmod{251}$

  $= 73$



http://www.daemonology.net/papers/htt.pdf



Parting thoughts

- Implementations should be constant time, constant power, *etc.*
- Elliptic curves? Quantum resistant?
- Especially hard if the attacker can modulate the power, clock, *etc.*