Advancements in the bleeding edge of cryptanalysis

PUTTING IT ALL TOGETHER

Who am I?

- David Hulton

ToorCon

- San Diego
  - 10 Years
- Seattle
  - 2 Years
- Camp
  - May 20th-25th, 2009
  - Titan-1 Missile Silo

ToorCon Foundation

- Started to help promote free thinking and hacking around the world
- Raised $2880 at ToorCon 10
- Bought 5 laptops for the first/only Montessori School in India
- Help fund scholarships for young hackers to go to conferences
- Financially promote open source project development in 3rd world countries
- Encourage global collaboration and communication

ToorCon Foundation

- We only directly benefit organizations that we know are legitimate

Hackerspaces oh my!

- Dachb0den Labs
  - San Diego: 2001-2005
- Public N3rd Area / Hackerbot Labs
  - Seattle: 2005-Present

Pico Computing

- Mad scientist lab
- Make bleeding edge supercomputing hardware
- Develop expensive hacker gadgets
- Mostly with Field Programmable Gate Arrays (FPGAs)

OpenCiphers

- What do we do with the hardware?

WEP	30x
WPA	5x
FileVault	5x
WinZip	5x
Bluetooth	100x
GSM	6000x

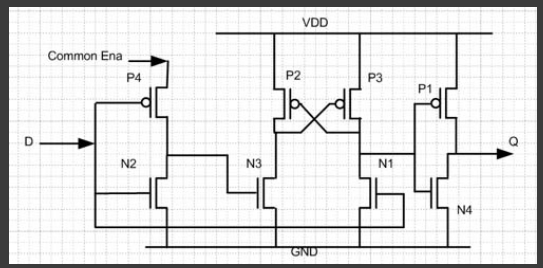
Computing 101

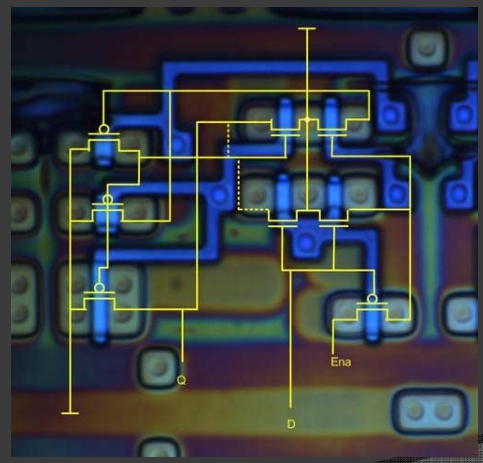
What are FPGA's good for?

- Software configurable ASIC
- Mostly used for prototyping ASIC's
- Very good at special purpose applications
- Especially good at bitwise operations
- Gains performance from parallelism
- Inherently good for crypto cracking

Computing 101

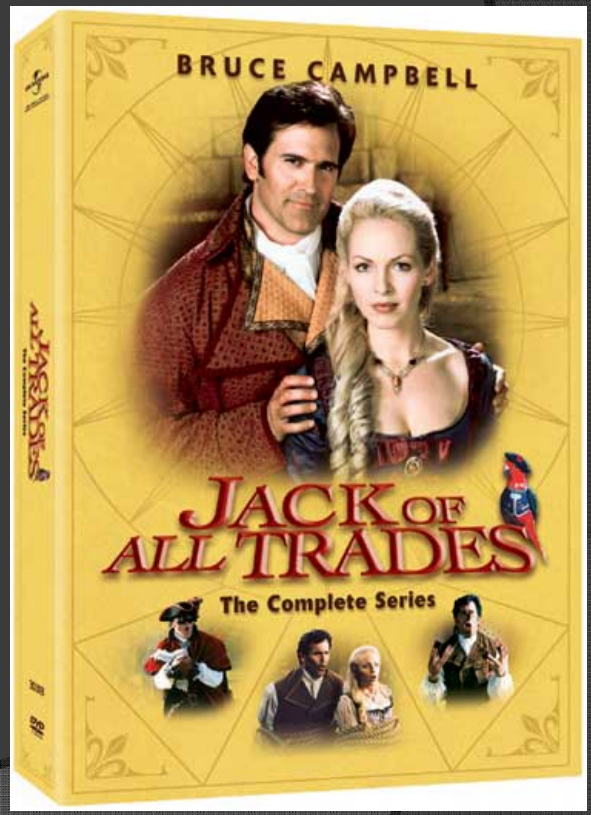
- Application Specific Integrated Circuit (ASIC)
  - Most chips out there
  - Processors, RAM, Flash, GPUs, etc





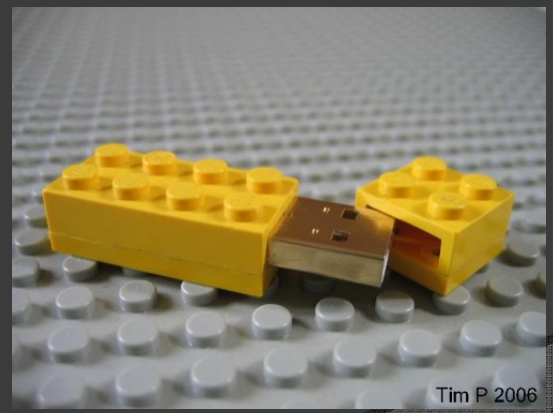
Computing 101

- Central Processing Unit (CPU)
  - Decent at doing “anything”



Computing 101

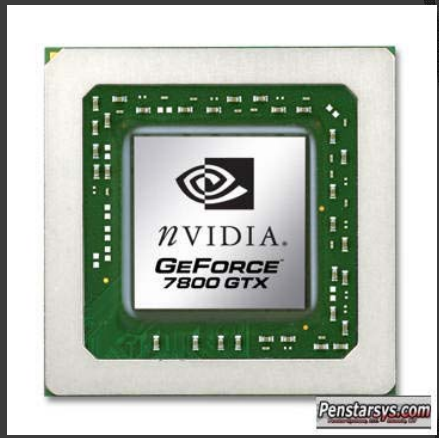
- RAM
  - Holding a lot of data
  - Providing fast access to it
  - Not good at holding data after power down

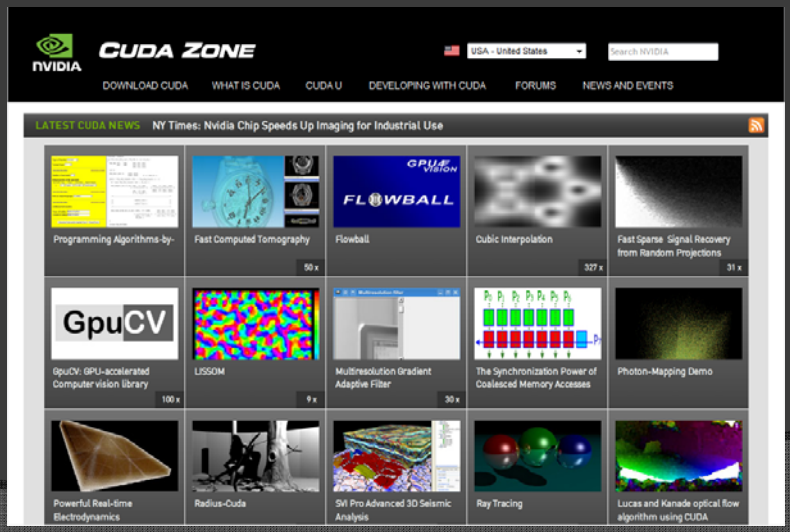


- Flash
  - Holding even more data
  - Providing quick access to it
  - Good at holding it after power down

Computing 101

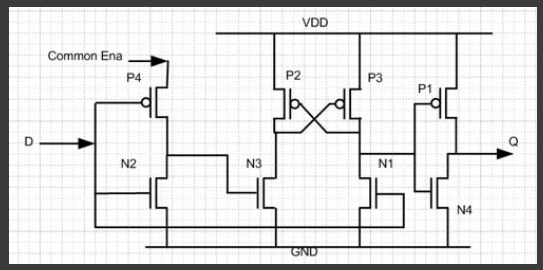
- Graphics Processing Unit (GPU)
  - Floating point operations
  - Matrix operations
  - General purpose operations
  - Parallelizable processes





Computing 101

- Field Programmable Gate Arrays (FPGAs)
  - Bitwise operations
  - Parallelizable processes
  - Prototyping ASIC designs

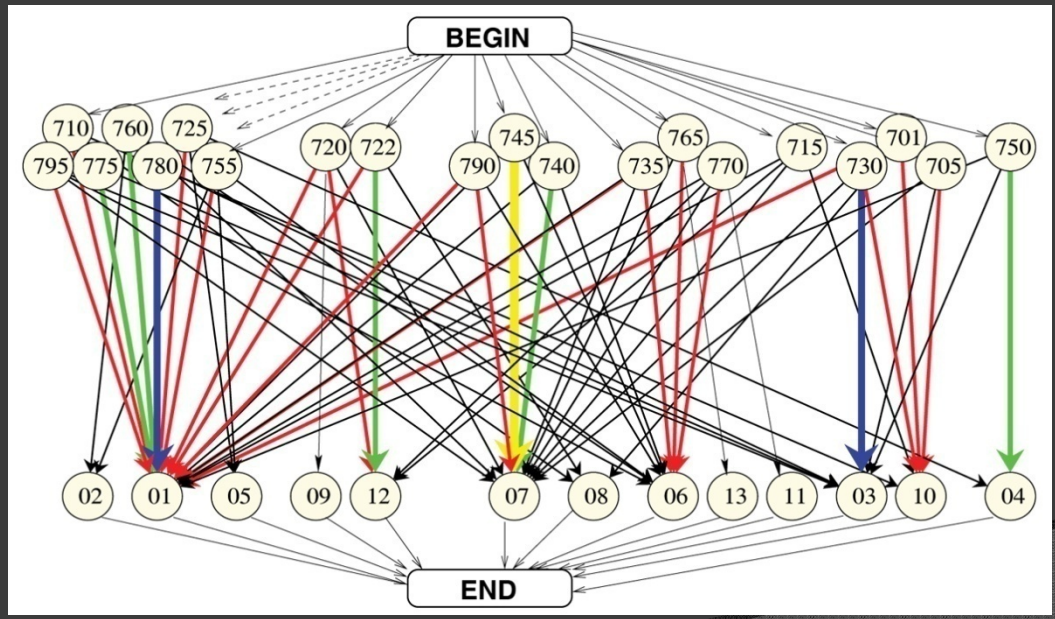


Quick Road Map

- Hidden Markov Models
- Rainbow Tables
- Crypto Algorithms
- World Domination

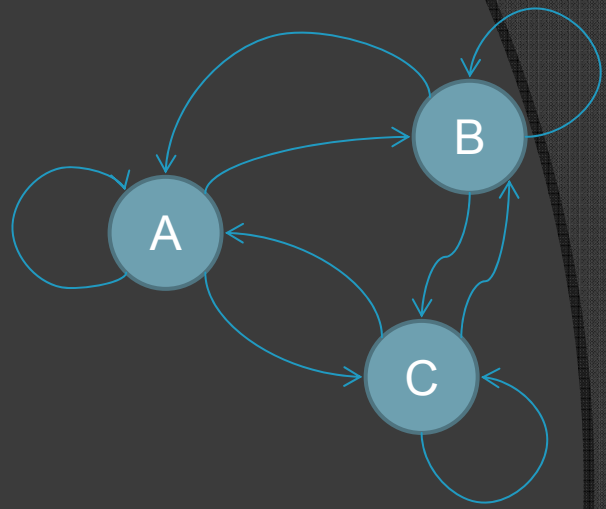
Some New Advancements

- Hidden Markov Model Password Generation
  - http://openwall.info/wiki/john/markov (9/08)



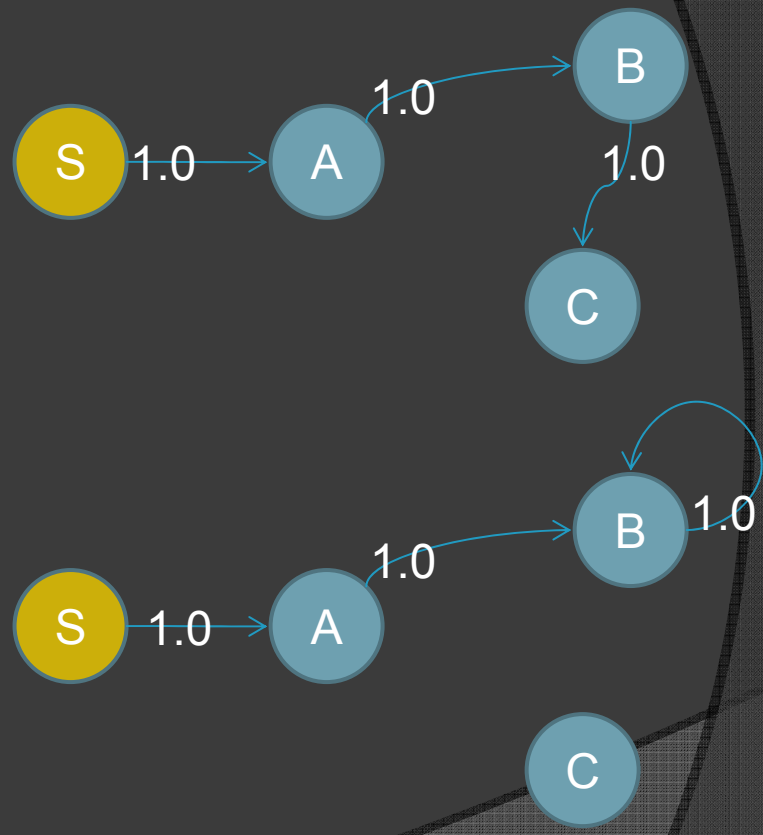
Hidden Markov Models

- Instead of using a wordlist you create a probability based state model for the character transitions
  - Used for:
    - Speech Recognition
    - Handwriting Recognition
    - Bioinformatics
    - Etc



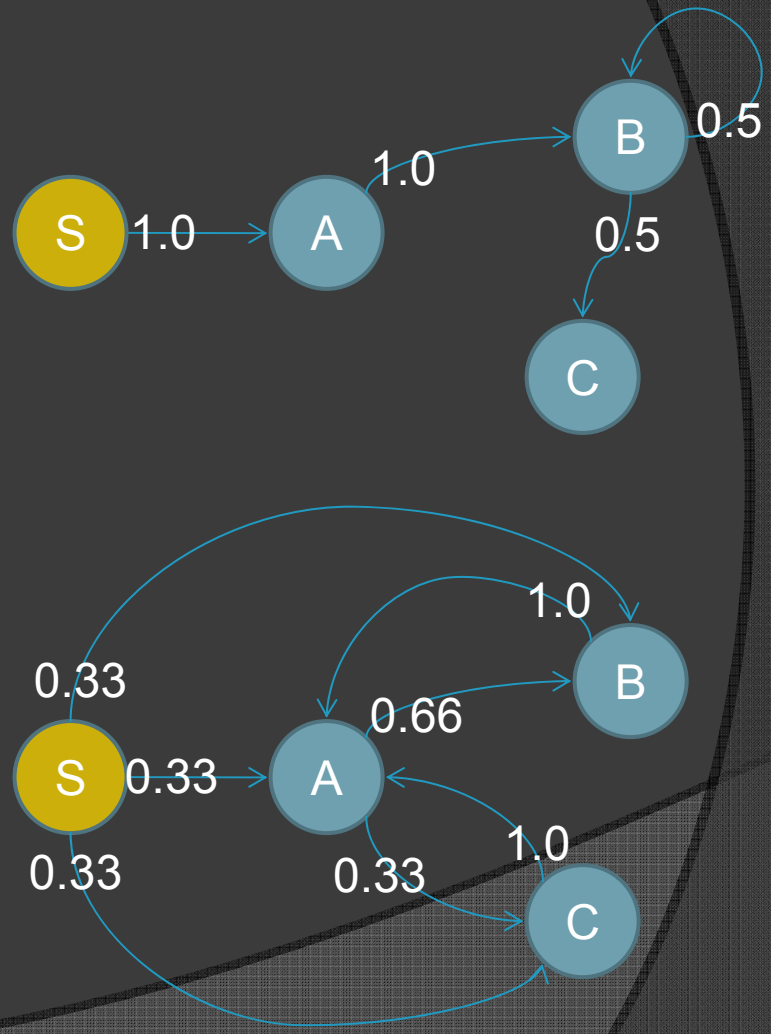
Hidden Markov Models

- For example:
  - ABC A->B B->C
  - ABB A->B B->B



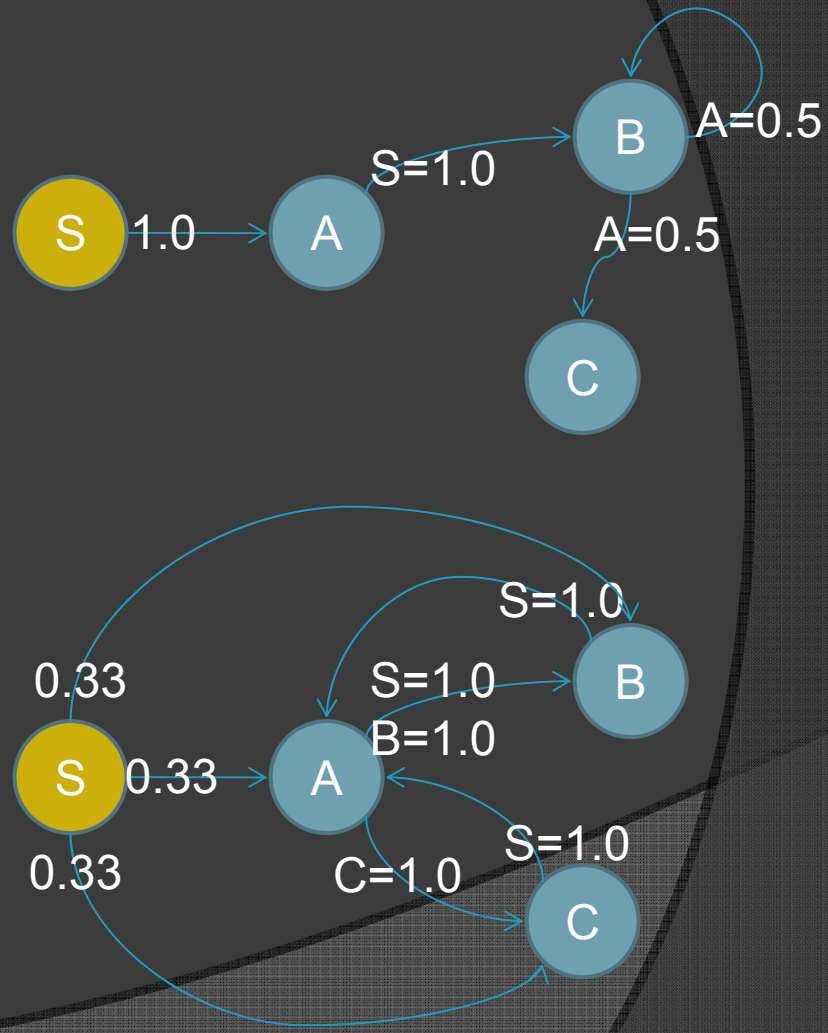
Hidden Markov Models

- Combined:
  - ABC A->B B->C
  - ABB A->B B->B
  - ABA A->B B->A
  - CAB C->A A->B
  - BAC B->A A->C



Hidden Markov Models

- 1st-order:
  - ABC SA->B AB->C
  - ABB SA->B AB->B
  - ABA SA->B AB->A
  - CAB SC->A CA->B
  - BAC SB->A BA->C



White Hat

- It models the frequency of different state transitions
- Some would argue that this is how we remember words
- Many password generators use HMMs to generate non-word passwords that users will remember

Black Hat

- Allows us to compress our wordlists or “train” the HMM
- Saves bus bandwidth when talking to FPGAs
- Our HMM will then generate passwords that are more likely to be real passwords
- Will allow us to generate many more possible passwords

Example

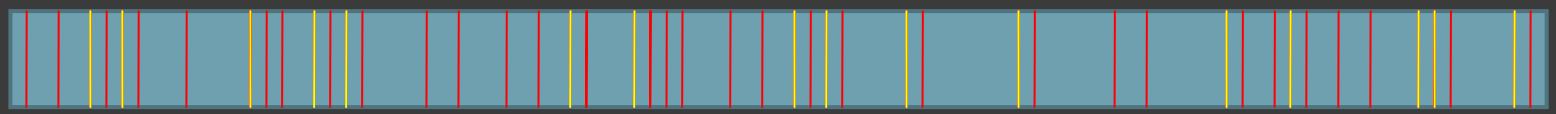
- HMM (0-order)
  - bonnarur
  - moshiffu
  - hirapuca
  - furviddr
  - tolveeru
  - tigerneb
  - bahoyete
  - ereusior
  - aareeiga
  - smindrme
- rand()
  - hdojushf
  - poaedfzv
  - jfuutywo
  - wisyqgma
  - exyohytr
  - spliutrg
  - cvawzmkl
  - erptprlg
  - isptpllg
  - zhwbswvs

HMM Wordlist Generation

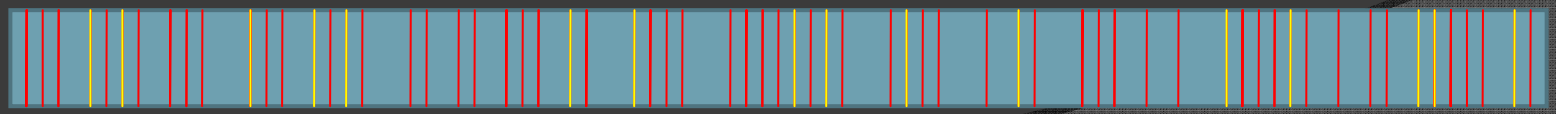
Entire Keyspace

Wordlist Words

HMM Space Small



HMM Space Large



Rainbow Tables

Rainbow Tables

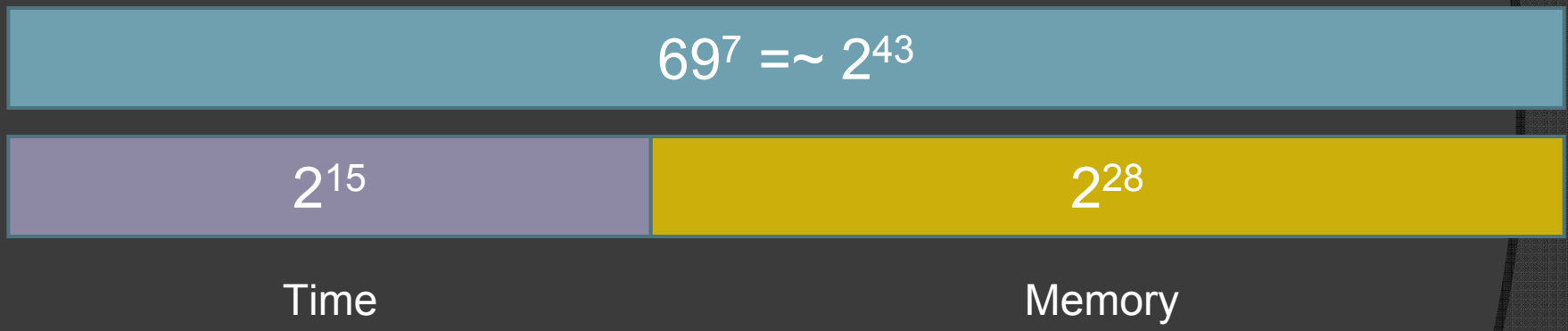
- Time/Memory Trade Off (TMTO)
- Lets you
  - Precompute a lookup table
  - But not have to store all of it
  - Trade time for memory

Rainbow Tables

- For example, breaking Lanman:
  - 69^7 possible passwords
  - Generate 69^7 hashes
  - To break a hash, look in table to see what password corresponds to your hash
  - 7.4 trillion * 16 bytes = 119TB storage

Rainbow Tables

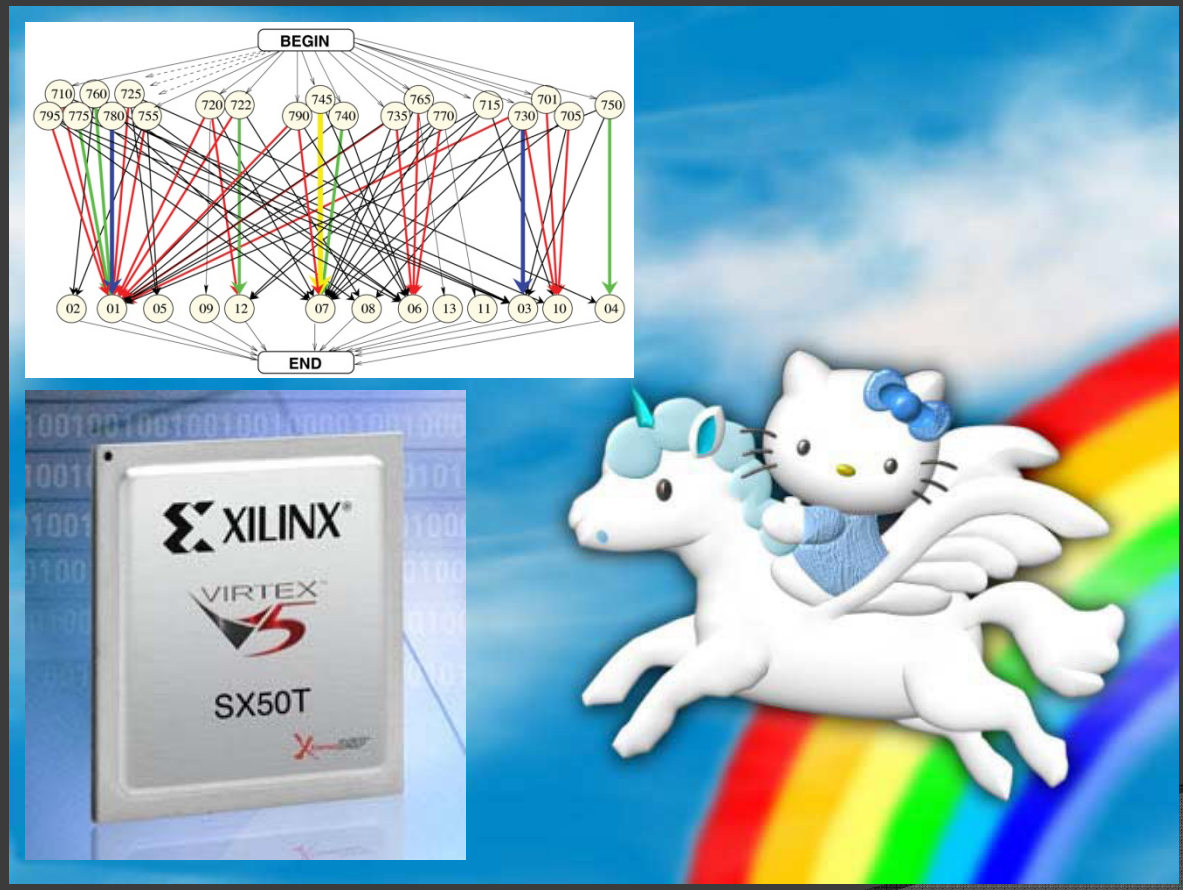
- Lets you split up the keyspace so you save part of it and compute part of it



- 2^{28} * 16 bytes = 4.2GB
- Takes 32,000 times longer to look up your hash

FPGA + Rainbow Tables

- Side effect
  - FPGAs can be used to attack keyspaces up to 12 bits larger than CPUs with certain algorithms


BEGIN
710 760 725 720 722 745 765 715 701 750
795 775 780 755 790 740 735 770 730 705
02 01 05 09 12 07 08 06 13 11 03 10 04
END
XILINX
VIRTEX
SX50T

Some New Advancements

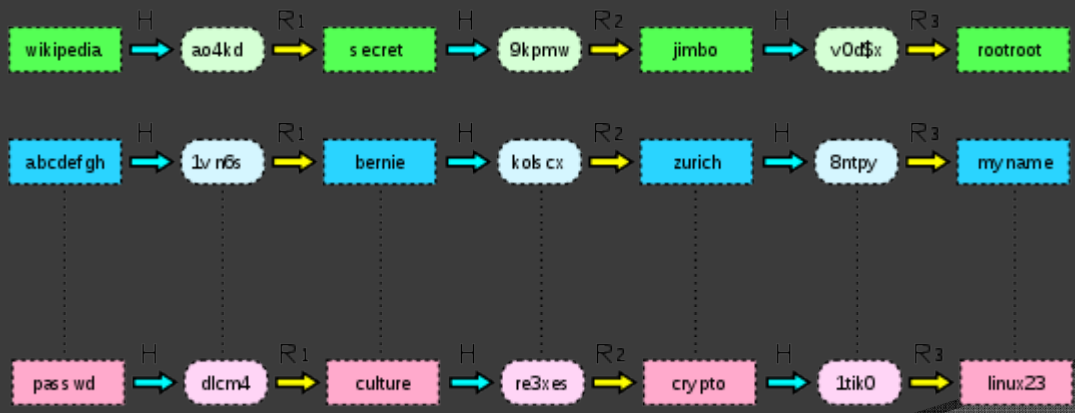
- Rainbow Tables \w Hidden Markov Models
  - Arvind Narayanan and Vitaly Shmatikov ’05

Rainbow Tables \w Hidden Markov Models

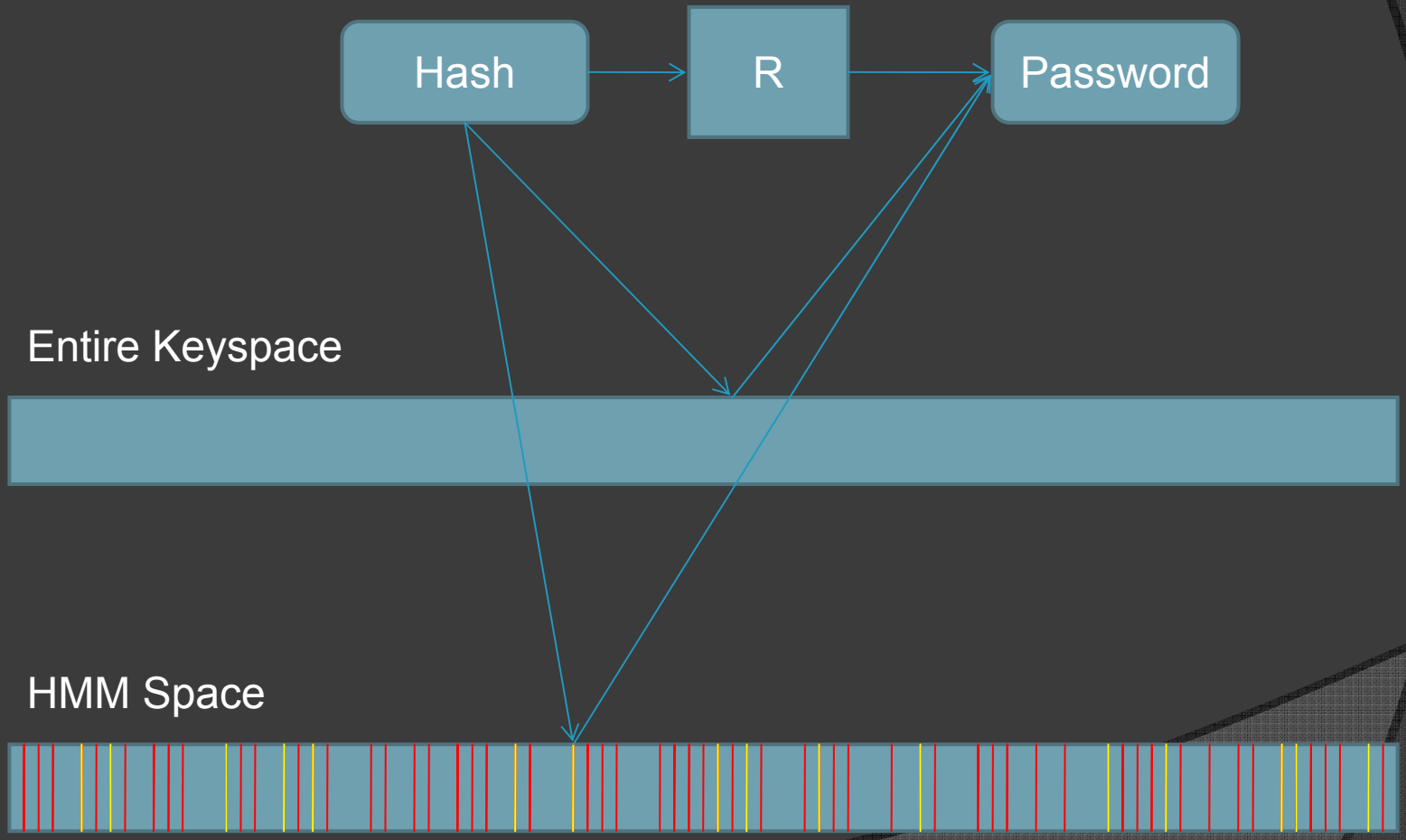
- 69^7 has many passwords that people would never use
- Why not build a rainbow table with only the most probable passwords?
- This could allow you to attack even more complicated passwords with less space

Rainbow Tables \w Hidden Markov Models

- Replace the reduction function with an HMM filter
- Easy to implement on FPGAs



Rainbow Tables \w Hidden Markov Models



What won’t Rainbow Tables work for?

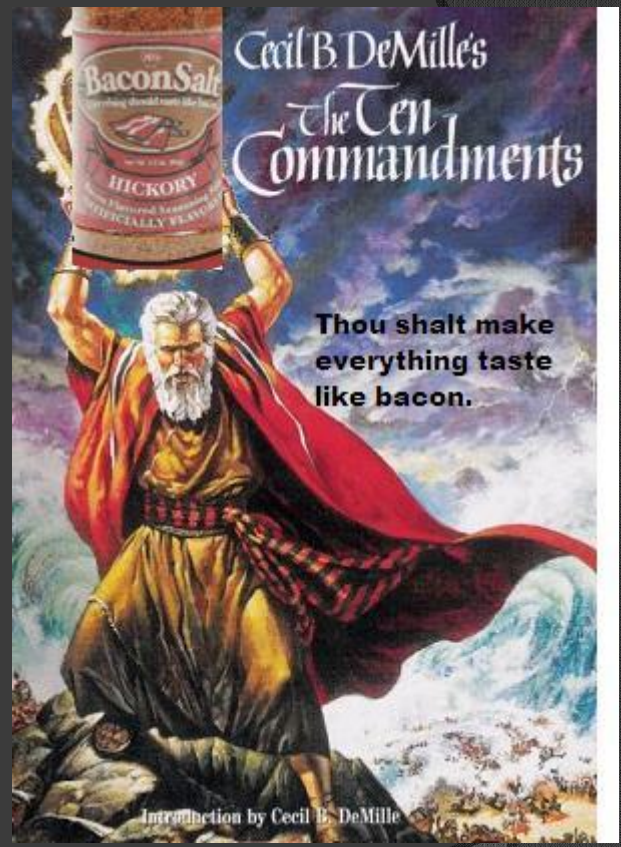
- Traditionally
  - Keyspaces that are too large
  - Salted algorithms

Larger keyspaces

- Larger keyspaces can now be attacked
  - Passwords of longer lengths with low entropy are now vulnerable to rainbow table attacks
  - Add FPGAs and now we're getting somewhere

What about Salt?

- Salt
  - salt | hash = H(salt | password)
  - Have to create a rainbow table for every possible salt as well as password
- Most new schemes use salt



Please pass the salt

The benefit provided by using a salted password is that a simple dictionary attack against the *encrypted* values becomes impractical if the salt is large enough. That is, an attacker would not be able to create a rainbow table, a dictionary of encrypted values (password + salt), because it would either take too much time, or too much space. This would force the attacker to use the provided authentication mechanism (which "knows" the correct salt value)."

http://en.wikipedia.org/wiki/Salt_(cryptography)

Please pass the salt

The benefit provided by using a salted password is that a simple dictionary attack against the *encrypted* values becomes impractical if the salt is large enough. That is, an attacker would not be able to create a rainbow table, a dictionary of encrypted values (password + salt), because it would either take too much time, or too much space. This would force the attacker to use the provided authentication mechanism (which "knows" the correct salt value)."

http://en.wikipedia.org/wiki/Salt_(cryptography)

Unix DES crypt()

- Used in most legacy and embedded *nix systems
- Supported by libc/perl/php/Apache
- /etc/passwd, /etc/shadow
  - 25 rounds of DES
  - Maximum of 8 character passwords
  - 12-bit salt
  - Do you see a problem here?

Unix DES crypt()

- Assumptions
  - We have a month of pre-computation time
  - We have 128 FPGAs to throw at this
- Performance
  - (1 billion / 25) * 128 * 60 * 60 * 24 * 30 =
  - 13,271,040,000,000,000 / 2^{12} =
  - 3,240,000,000,000 passwords per salt =
  - ~ 1/67th of the [a-zA-Z0-9] 8-character keyspace

Lesson

- Use a large salt value

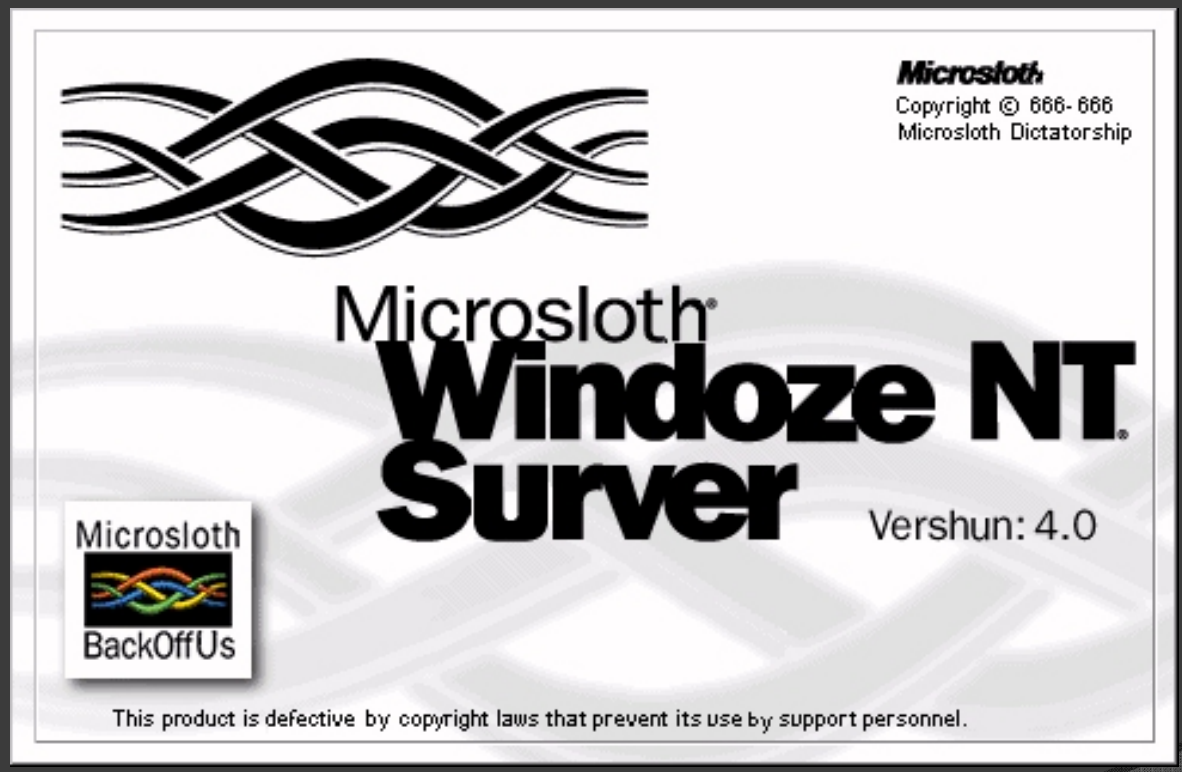
But, everyone uses salt these days! ..... rright?

- You would be surprised

Salt? Who needs Salt?

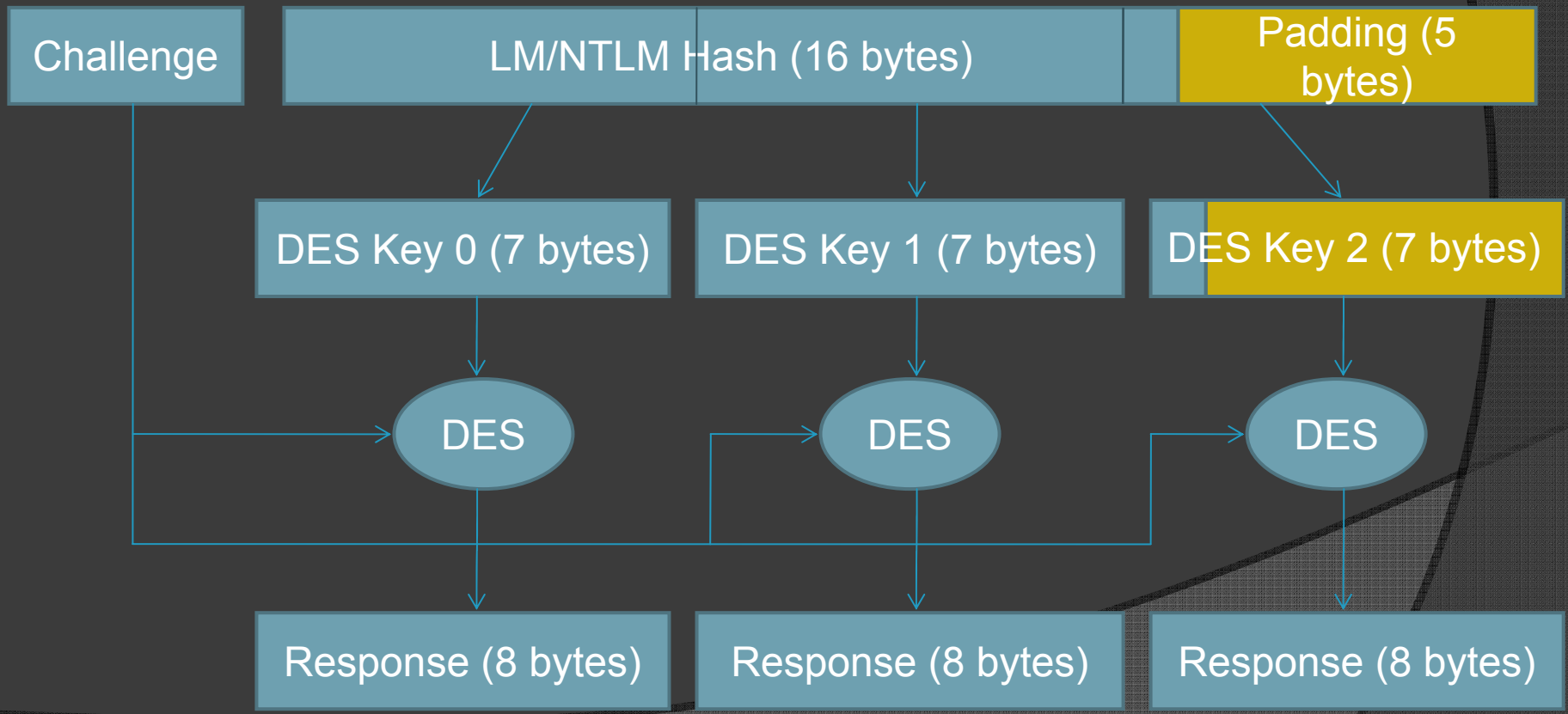
- MSG Grade Crypto:
  - Lanman
    - We already know this is broken
  - NTLM
    - No salt is just the tip of the iceberg
  - DES
    - Many DES implementations
  - MySQL
    - Even in their security fix for 4.1
  - Apache SHA1 (mod_dbd, auth, etc)
    - Also support standard unix crypt()
  - Office 97 effectively doesn't
    - 40-bit rainbowtables ala Objectif Securite
  - PDF 1.3 effectively doesn't
    - 40-bit PDF rainbowtables ala ElcomSoft

Lanman & NTLM


Microsloth
Copyright © 666-666
Microsloth Dictatorship
Microsloth
Windoze NT
Surver
Vershun: 4.0
Microsloth
BackOffUs
This product is defective by copyright laws that prevent its use by support personnel.

Lanman & NTLM

- Network authentication uses DES



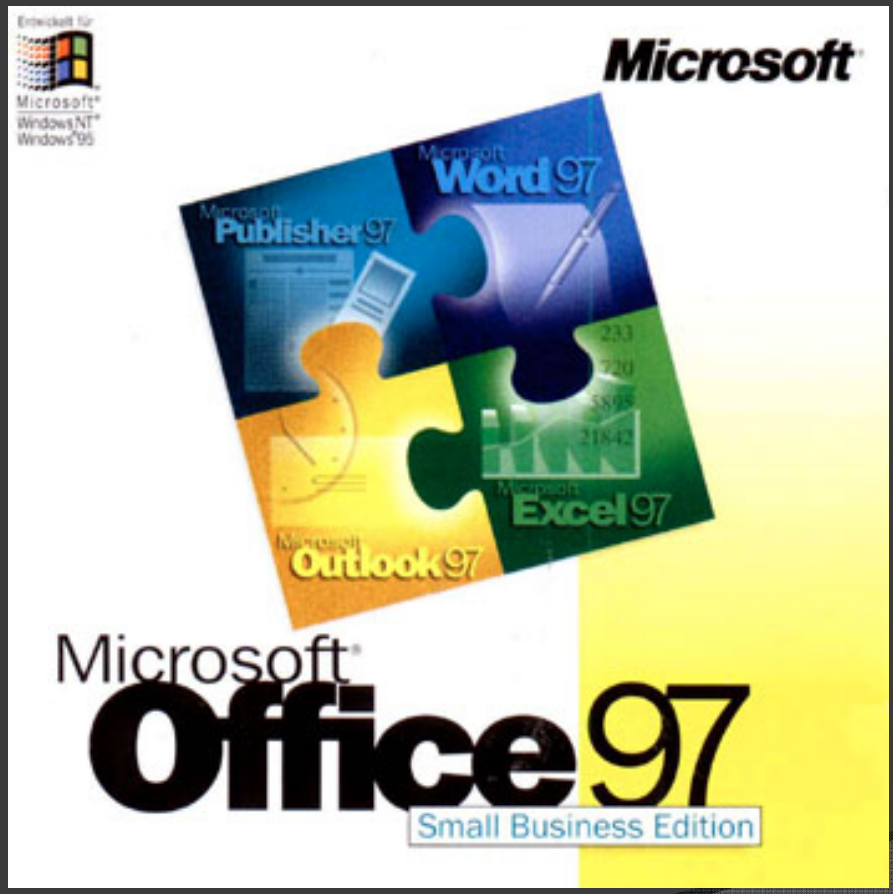
Lanman & NTLM

- DES
  - Static plaintext
  - 56-bit key
  - Perfect for a 56-bit rainbow table
  - FPGA acceleration makes it feasible
  - Applies to both Lanman & NTLM

Lanman & NTLM

- Assumptions
  - We have 128 FPGAs to throw at this
  - We will achieve about 50% collisions
- Performance
  - $(2^{56} * 2) / (1 \text{ billion} * 128 * 60 * 60 * 24) =$
  - 13 days
  - ~8TB of storage and 128 FPGAs for real-time decryption
  - Around 2 minutes with 1 FPGA

Office 97


Microsoft
Windows NT
Windows 95
Microsoft
Word 97
Publisher 97
Excel 97
Outlook 97
Microsoft
Office 97
Small Business Edition

Office 97

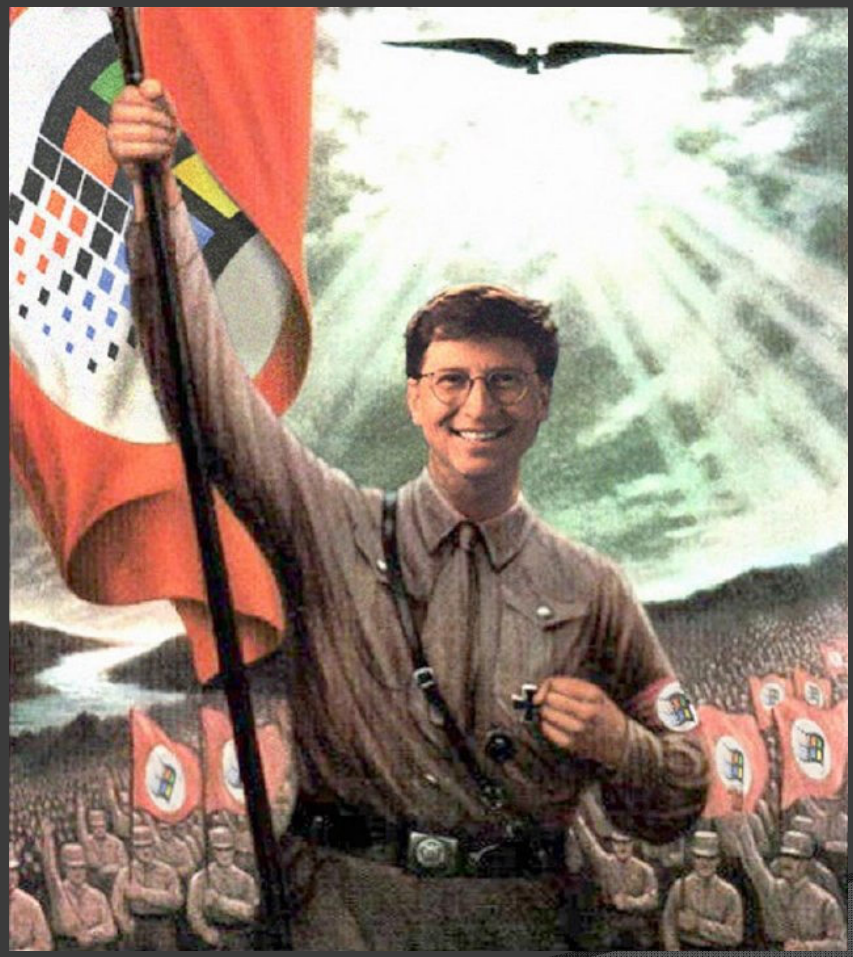
- 40-bit Rainbow Tables Exist
- 128-bit is possibly vulnerable to the HMM + FPGA attack

PDF 1.3

PDF 1.3

- 40-bit Rainbow Tables Exist
- 128-bit should be vulnerable to the HMM + FPGA attack

World Domination

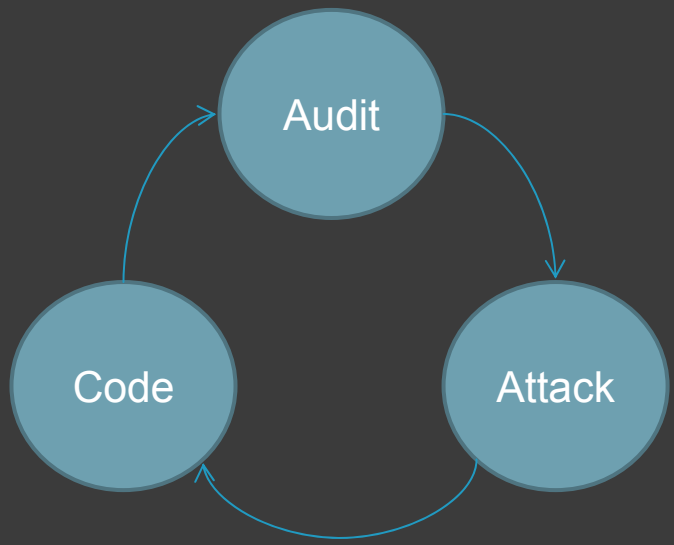


Where is security going?

- What should I take from this?
- How does the whole security ecosystem fit together?

Finding new classes of attacks

- Obvious



Using existing classes of attacks

- There are many attacks out there that people don't fully realize the scope of
- Many things are being exploited today with attacks that were developed decades ago

Example

- Buffer Overflows

Combining classes of attacks

- Sometimes things that are secure against one type of attack, aren't secure against 2 types combined

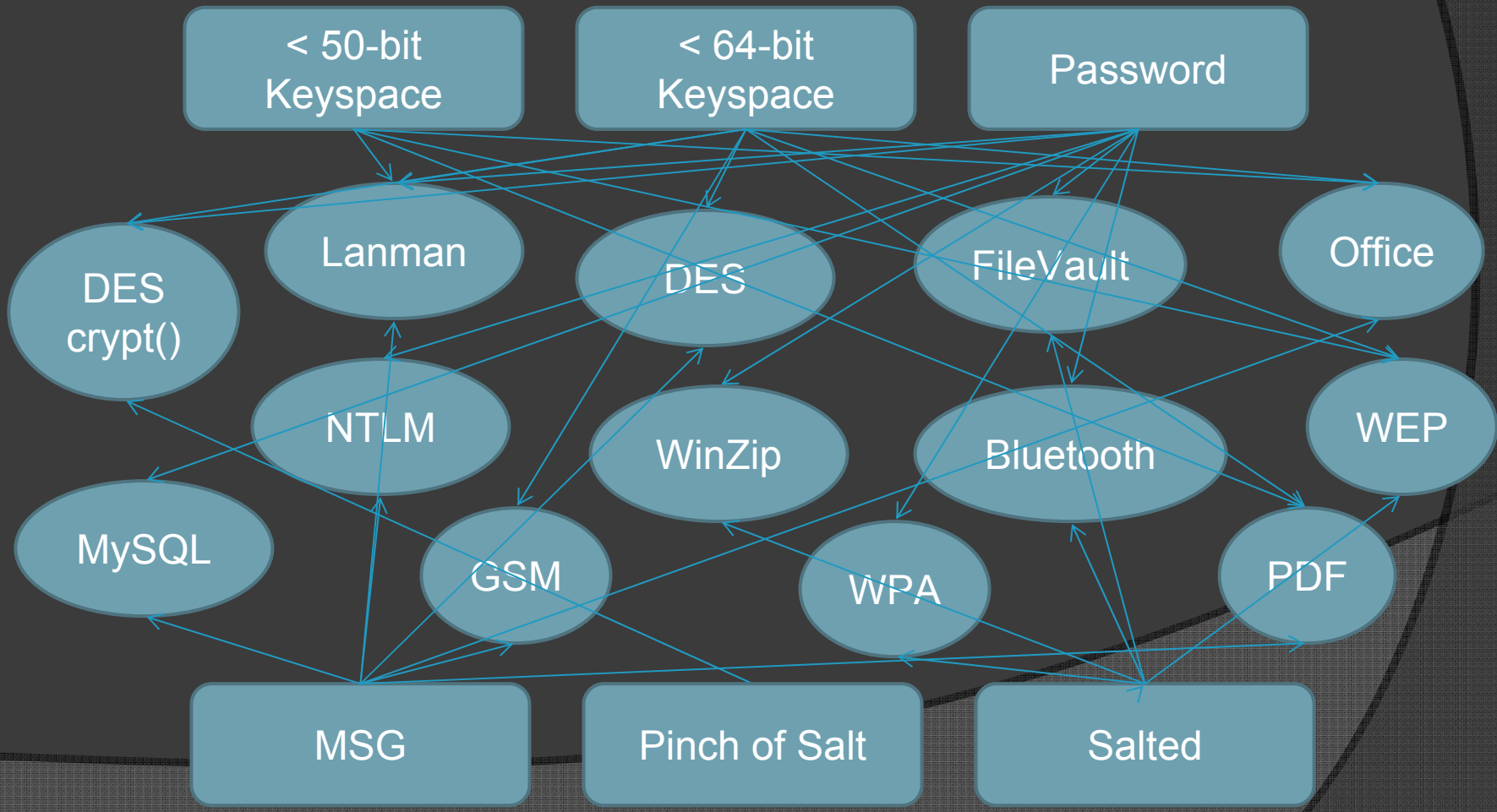
Example

- Cold Boot Attack
  - Gives you access to a PC's memory
  - Only useful when you're able to extract useful information from memory

- Dan Kaminsky DNS Attack
  - Opens up many new vulnerabilities
  - Auto-update attacks now possible
  - New phishing attacks
  - Vulnerabilities in SSL/VPNs/etc are now practical

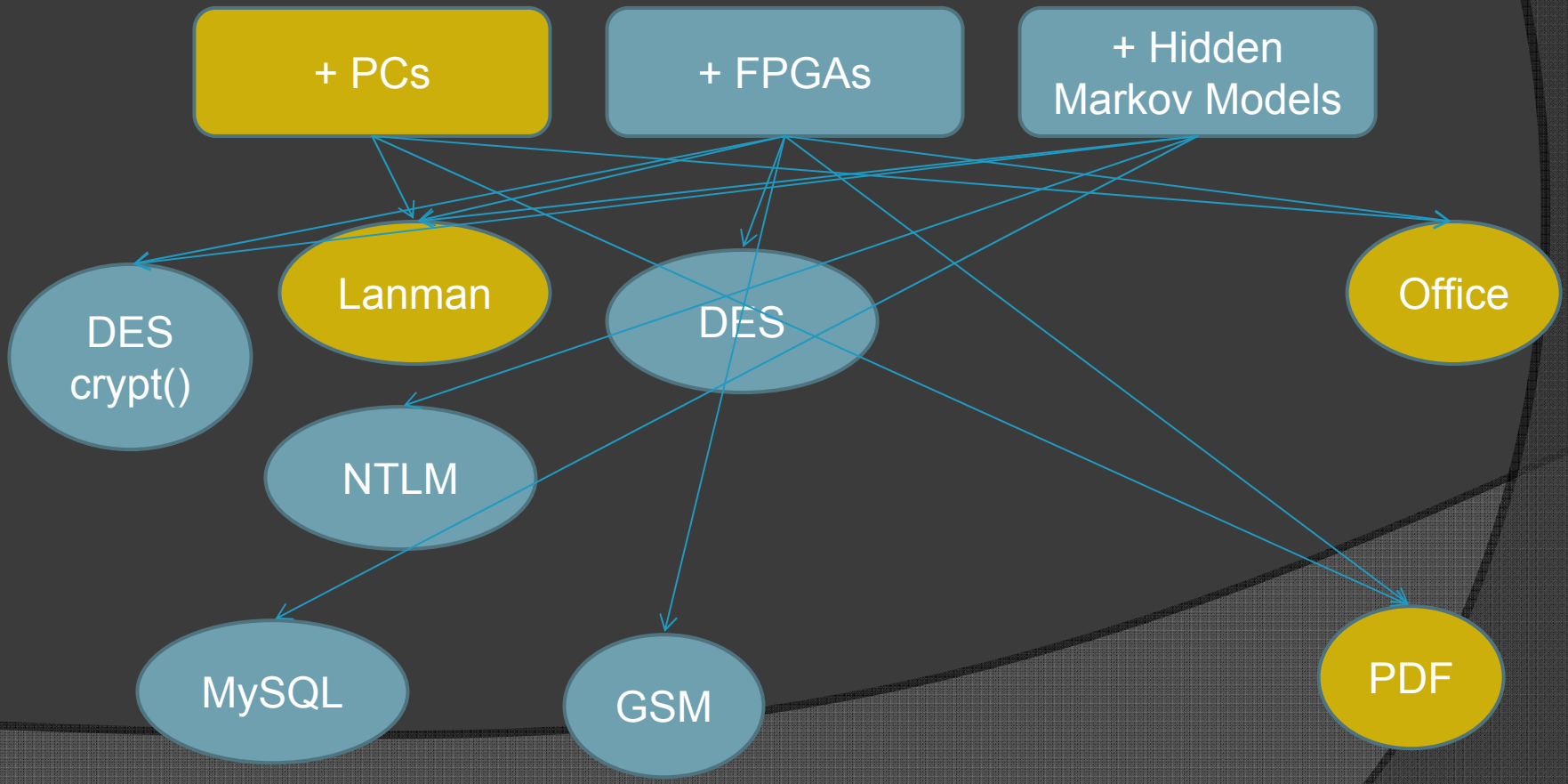
Example

- Going back to earlier



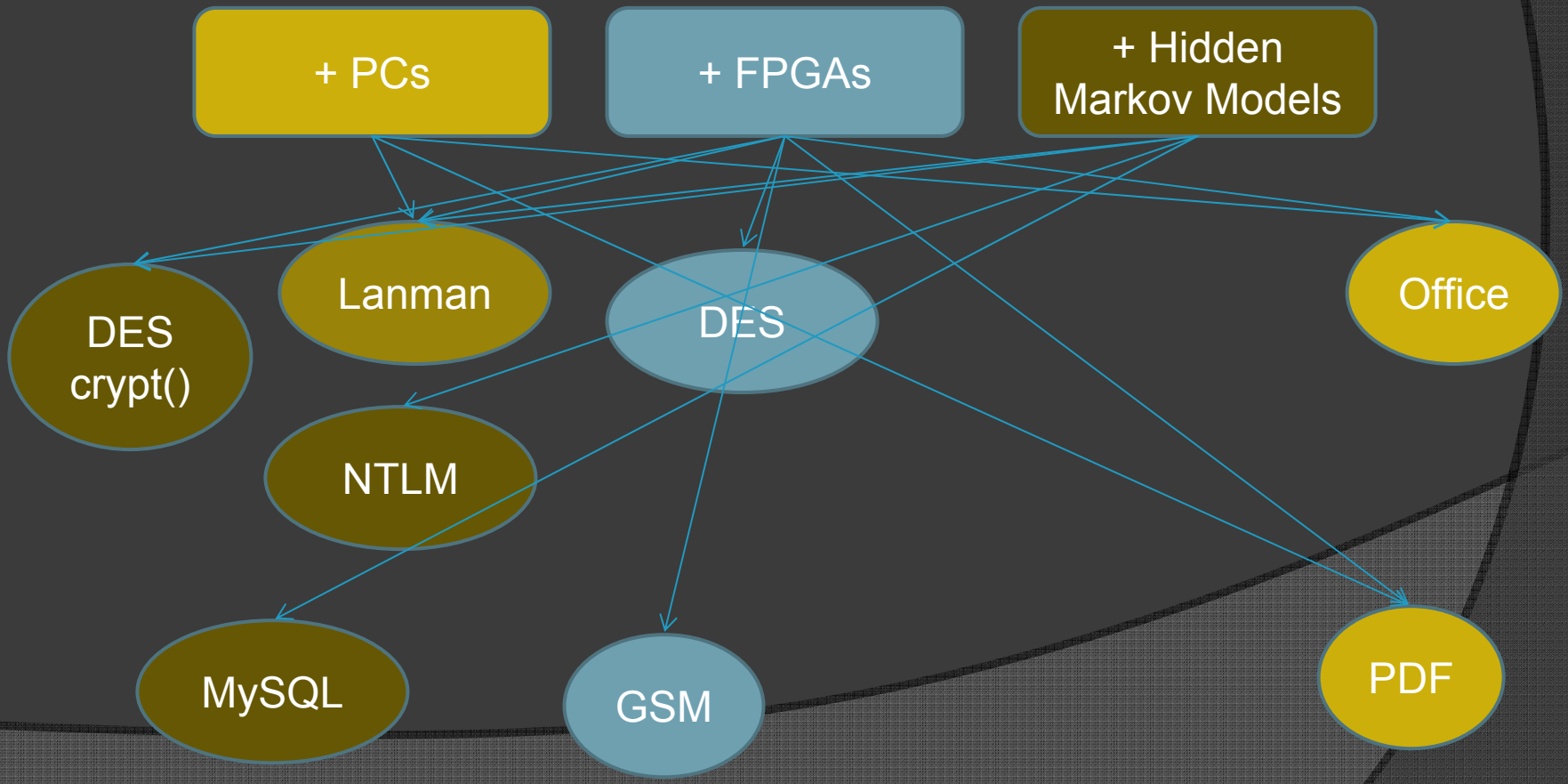
Rainbow Tables

- What could we do before with just PCs?



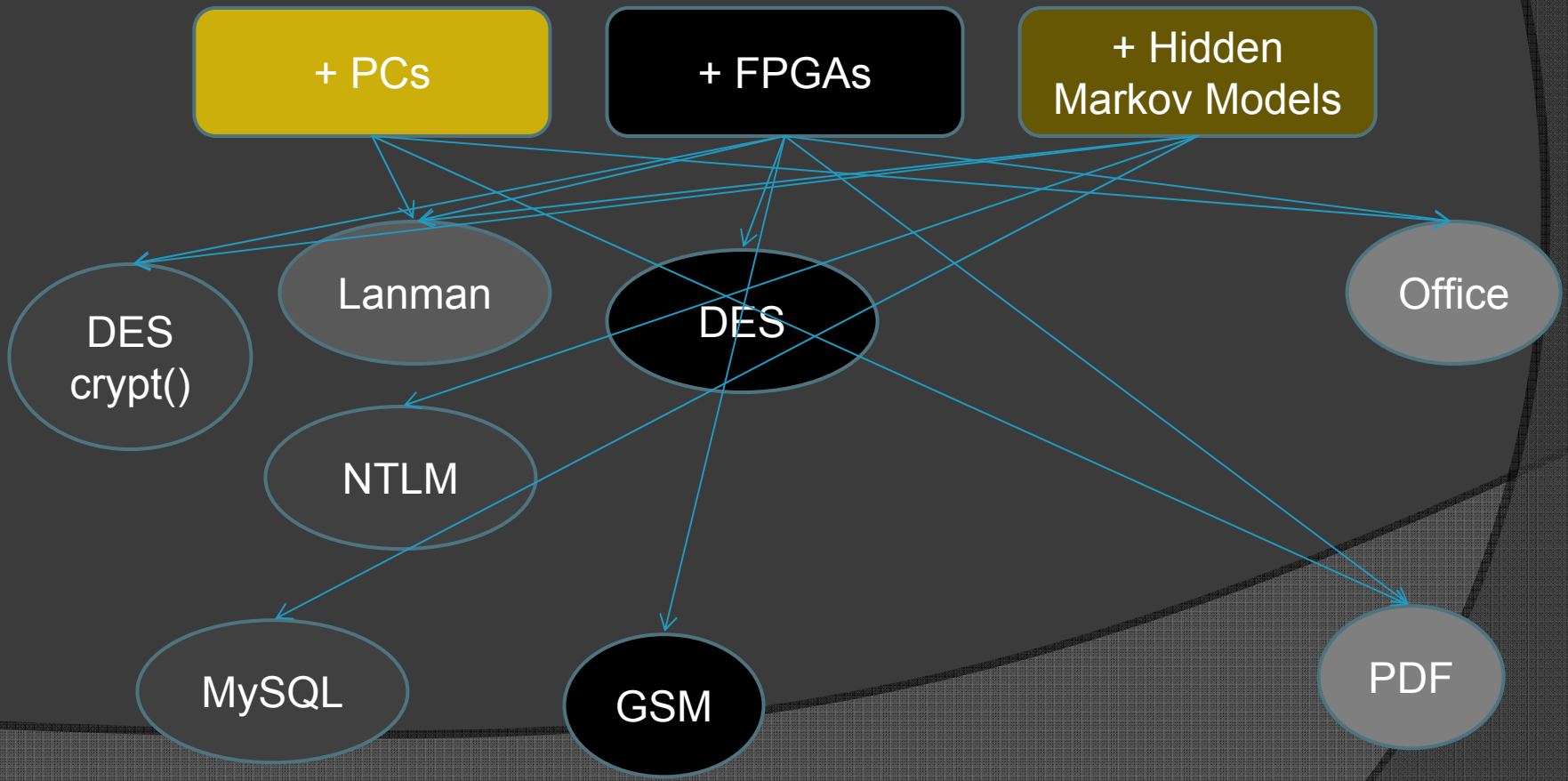
Rainbow Tables

- What can we do now with HMMs and PCs?



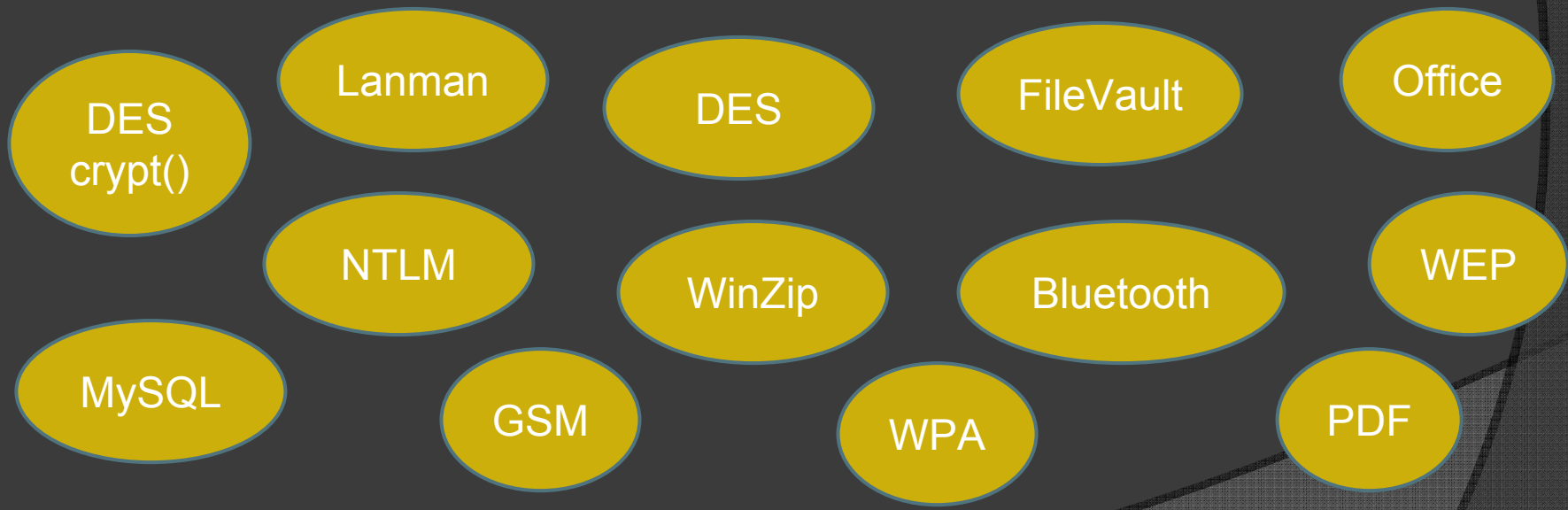
Rainbow Tables

- And with FPGAs?

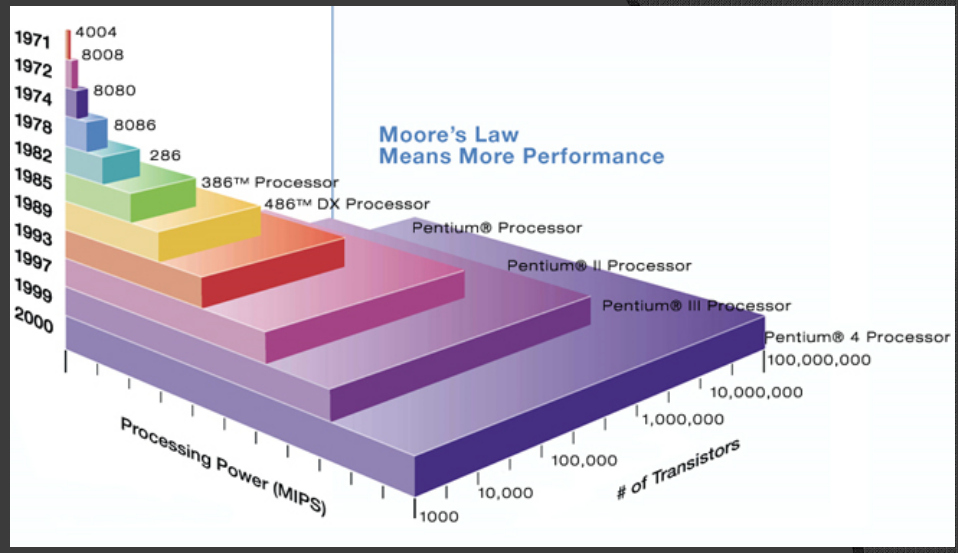


Brute Force

- FPGAs

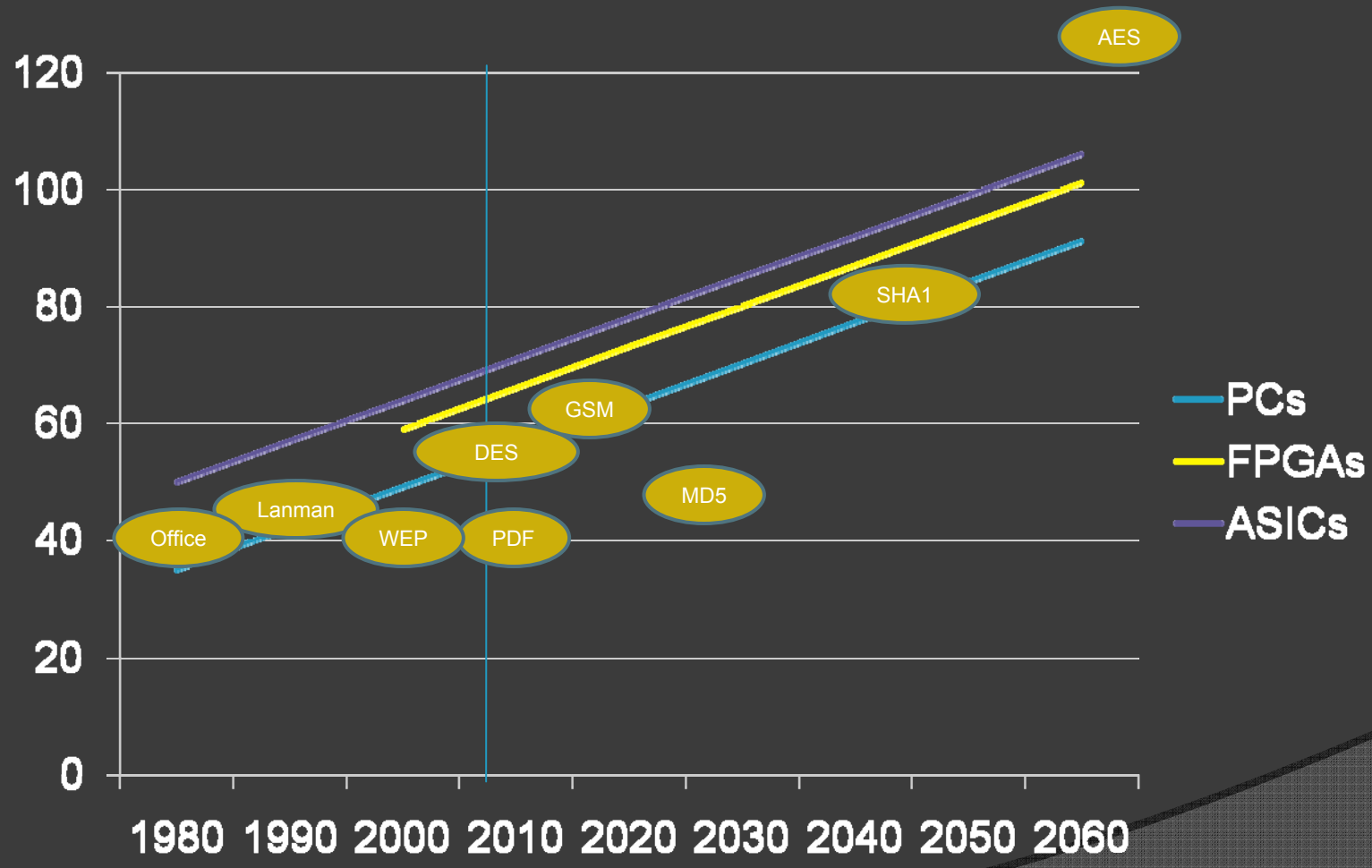


Moore’s Law



- The density of transistors on an ASIC doubles every 2 years
- Performance inherently increases with smaller transistors
- Performance doubles every 18 months
  - 10x every 5 years
  - 100x every 10 years
  - 10,000x every 20 years

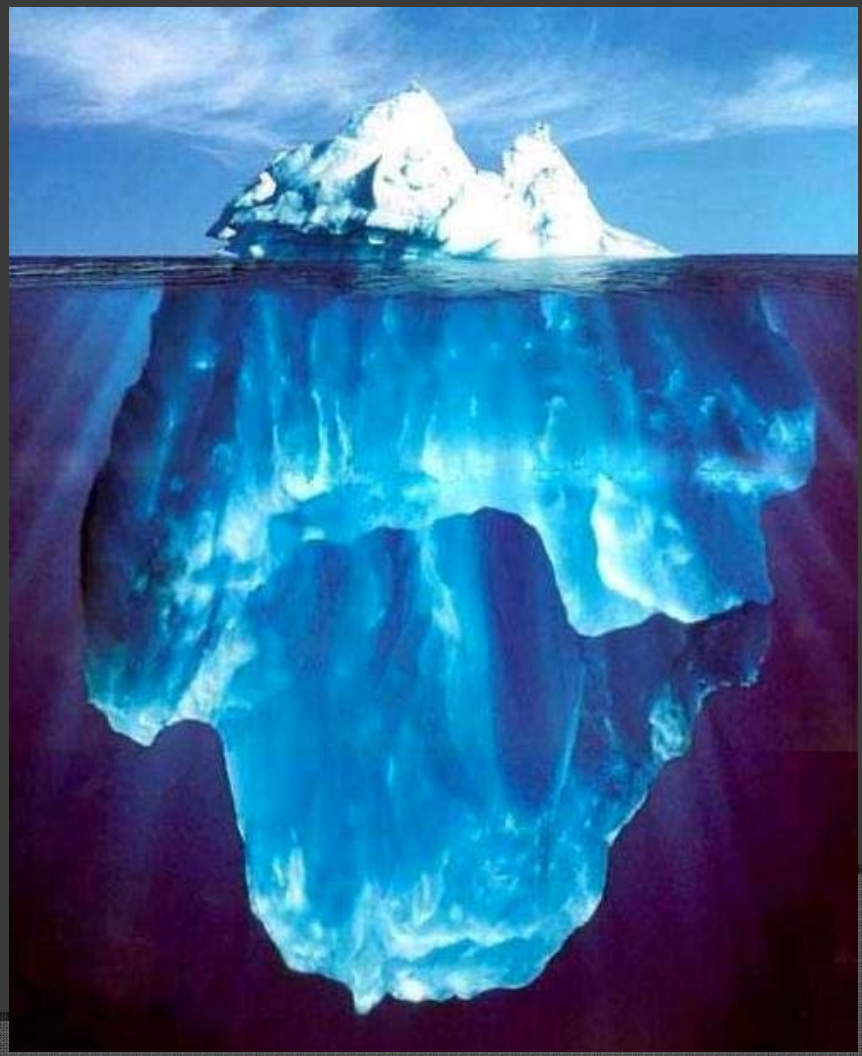
Moore's Law


120
100
80
60
40
20
0
1980
1990
2000
2010
2020
2030
2040
2050
2060
AES
SHA1
GSM
DES
MD5
Lanman
Office
WEP
PDF
PCs
FPGAs
ASICs

What should we take away from this?

- Security is an iterative process
- A point & click solution doesn't always work
- You must understand the intricacies of how the whole system fits together
- Just because something is deemed secure today, doesn't mean it will be tomorrow

There is much that's left to be done



Have fun!

Beware!


HACKERS
AHEAD
PROTECT YOUR STUFF: GET COMPUTER UPDATES.
SECURITY UPDATES DO MORE THAN JUST PROTECT YOUR COMPUTER AGAINST
THE LATEST WORMS, VIRUSES, HACKS, AND OTHER INTERNET VILLAINS.
THEY PROTECT YOUR PRIVACY AND PRODUCTIVITY.
For more information on protecting your PC, visit www.microsoft.com/protect
Microsoft
DELL
McAfee
symantec.

Questions?

- David Hulton
- 0x31337@gmail.com

- http://openciphers.org
- http://picocomputing.com
- http://toorcon.org