RSAConference2016

San Francisco | February 29 – March 4 | Moscone Center

Connect to Protect

SESSION ID: PDAC-W05

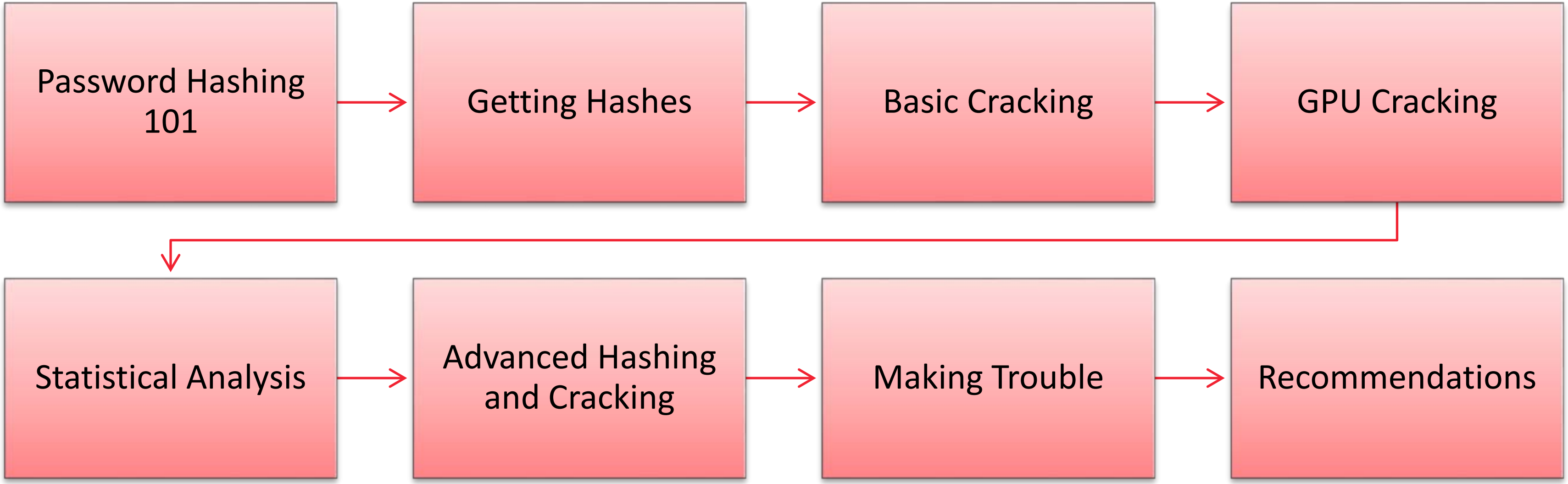
The State of Modern Password Cracking

Christopher Camejo

Director of Threat and Vulnerability Analysis
NTT Com Security
@0x434a

#RSAC

Presentation Overview



Password Hashing 101

Getting Hashes

Basic Cracking

GPU Cracking

Statistical Analysis

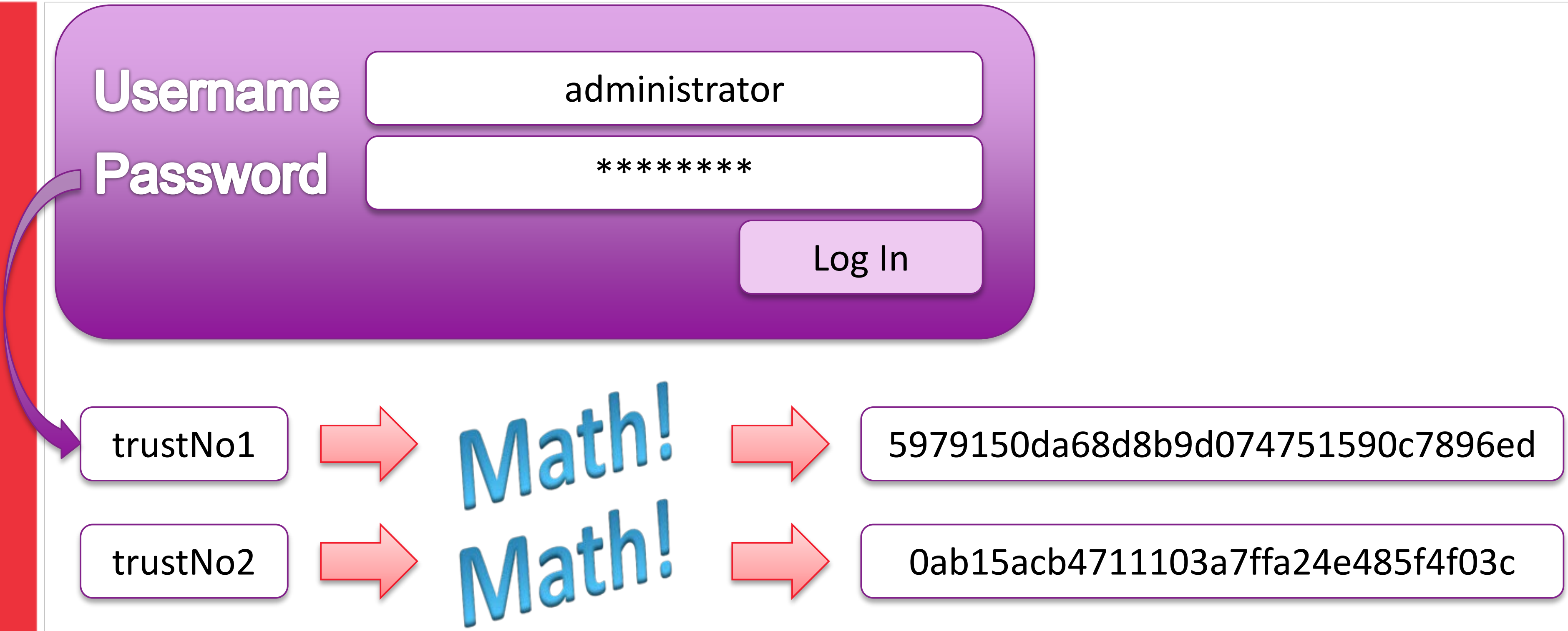
Advanced Hashing and Cracking

Making Trouble

Recommendations

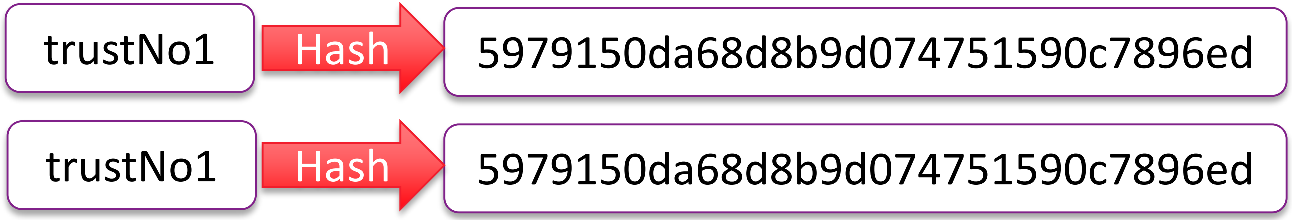
Password Hashing 101

Password Hashing 101



Adding Some Salt

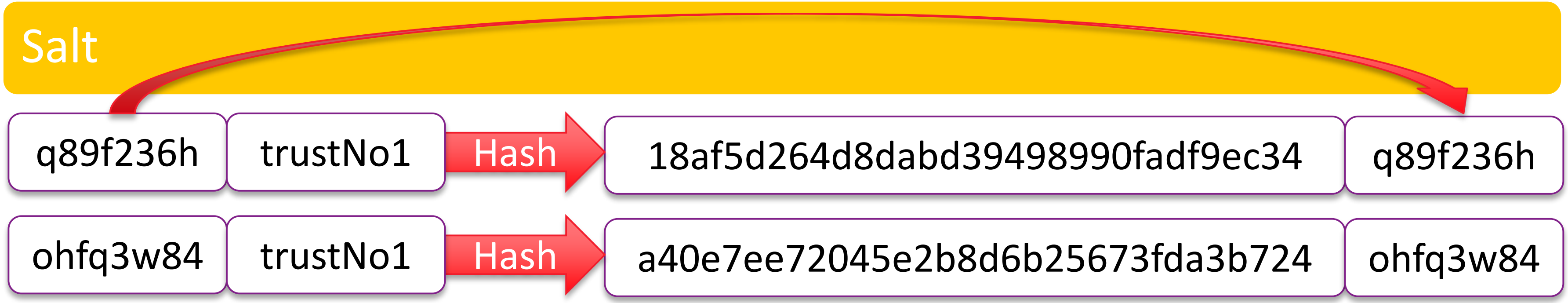
No Salt



trustNo1 → Hash → 5979150da68d8b9d074751590c7896ed

trustNo1 → Hash → 5979150da68d8b9d074751590c7896ed

Salt



q89f236h trustNo1 → Hash → 18af5d264d8dabd39498990fadf9ec34 q89f236h

ohfq3w84 trustNo1 → Hash → a40e7ee72045e2b8d6b25673fda3b724 ohfq3w84

Getting Hashes

Stealing Hashes

Compromise a Host

- Local Caches
- Network Sniffing

Application Vulnerabilities

- SQL Injection
- File Inclusion

Leaked Code

- Hardcoded Client Passwords
- Backdoor Hashes

Indecent Exposure

Search GitHub for *‘abc.com’ password*

```
Warehouse.define do
  warehouse :User do
    login "admin@abc.com"
    name ""
    access  "super"
    email "admin@abc.com"
    password "$2a$08$Y.JcIVvVQMk4UiToFFlLSObWeHYIT2zHdJrhYsgjdZdW7ZzByioh6"
    reset_token nil
 end
end
```

Basic Password Cracking

Entropy (lack thereof)

Standard keyboard:	• 95 characters
“Reasonable” password length:	• 10 characters
Possible combinations:	• 60,510,648,114,517,000,000
Time to crack @ 200 million KPS:	• 9,587 years

Powers of 2

Time to crack @ 200 million keys per second

Length	Lowercase Letters	Lowercase Alphanumeric	Mixed Case Alphanumeric	All characters
6 character	**1.7 seconds**	11.2 seconds	4.9 minutes	1.1 hours
7 characters	41.8 seconds	[illegible]	[illegible] hours	4.1 days
8 characters	18.1 minutes	4.1 hours	[illegible].9 weeks	1.1 years
9 characters	7.9 hours	0.9 weeks	[illegible]	1.1 centuries
10 characters	1.3 weeks	31.1 weeks	1.4 centuries	**9.6 millennia**

secret

Secret123!

Entropy (lack thereof)

OED Entries:

- 291,500 entries
- @200 million/sec = 0.0015 seconds

Our “English” file

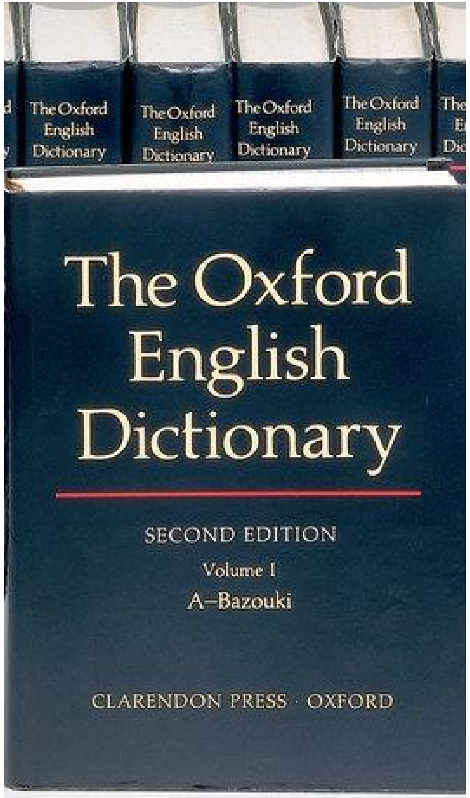
- 394,748 entries
- @200 million/sec = 0.0020 seconds

Our “Crack” file

- 148,903,320 entries
- @200 million/sec = 0.75 seconds

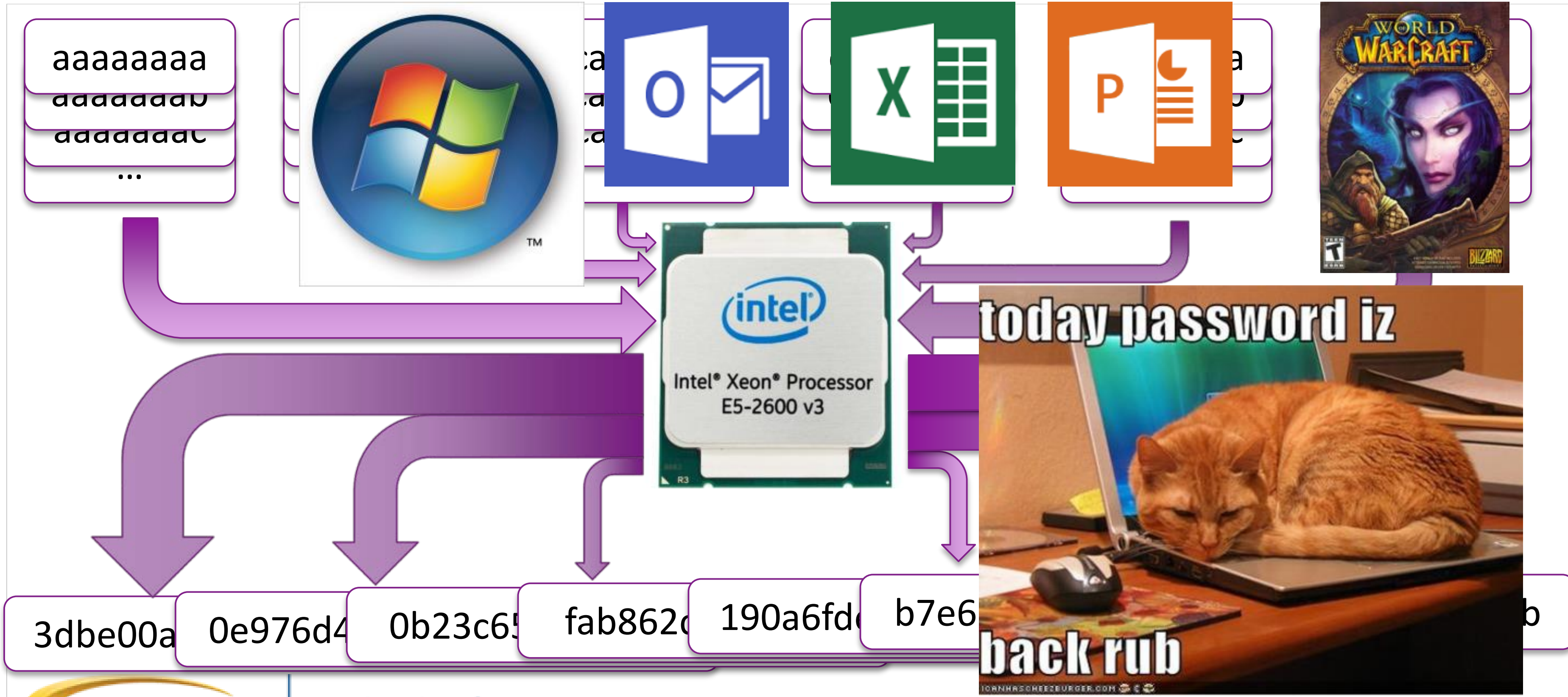
CEWL

- Spiders web sites and adds unique terms it finds to the dictionary file

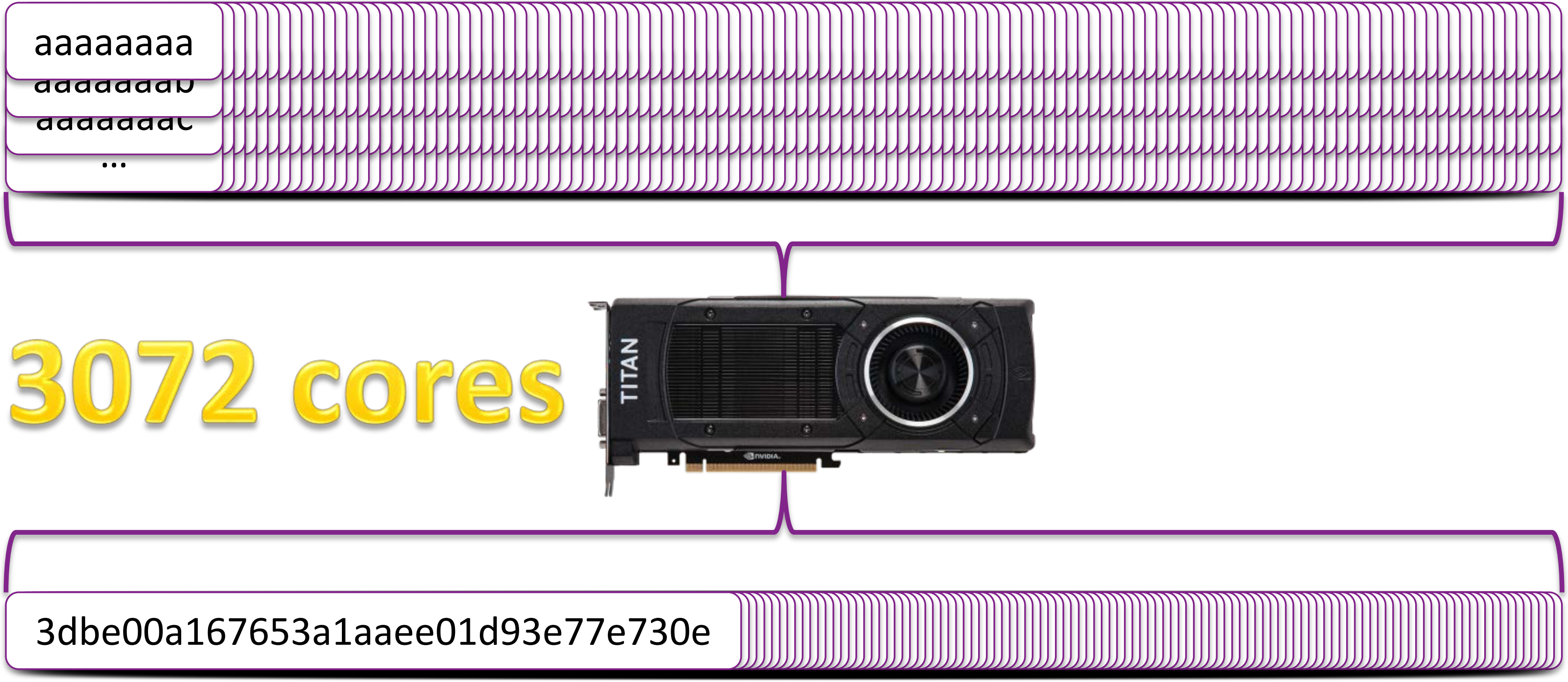


GPU Cracking

The Speed Problem



The Bottleneck Solution



aaaaaaaa

aaaaaaab

aaaaaaac

…

3072 cores

3dbe00a167653a1aaee01d93e77e730e

Hashcat

CPU and GPU cracking

Free/Open Source

Rules

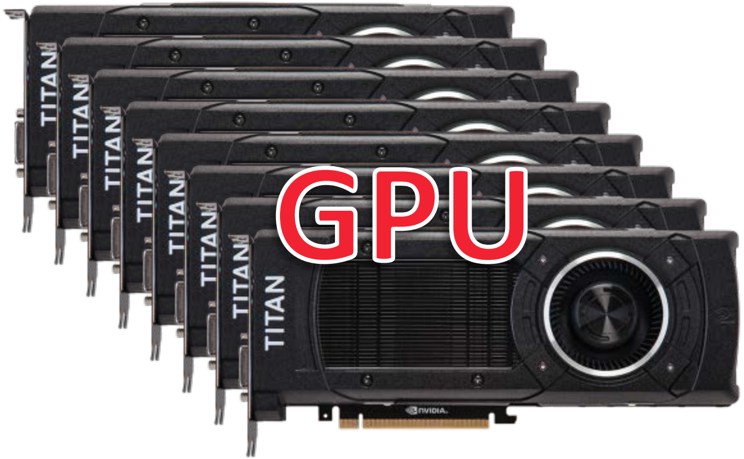
Modify dictionary words

Masks

Selective brute force

More Power

VS



	CPU		GPU
Model	Intel Xeon E5-2620 v3; $400 ea.	Model	Nvidia GeForce GTX Titan X; $1,000 ea.
Cores	6@2.4GHz x 2 CPUs = 12 cores	Cores	3,072@1GHz x 8 GPUs= 24,576 cores
MD5	205 million/sec	MD5	132 billion/sec
Crack 10 characters	9,353 years	Crack 10 characters	15 years

Statistical Analysis

PCI Compliance

PCI™ Security Standards Council

Req 8.2.3:

- 7 characters
- Alphanumeric

Req 8.2.4:

- Change <90 days

Time to 7 characters alphanumeric

MD5
5 mins

SHA512
6 mins

Time to 10 characters alphanumeric

MD5
3 days

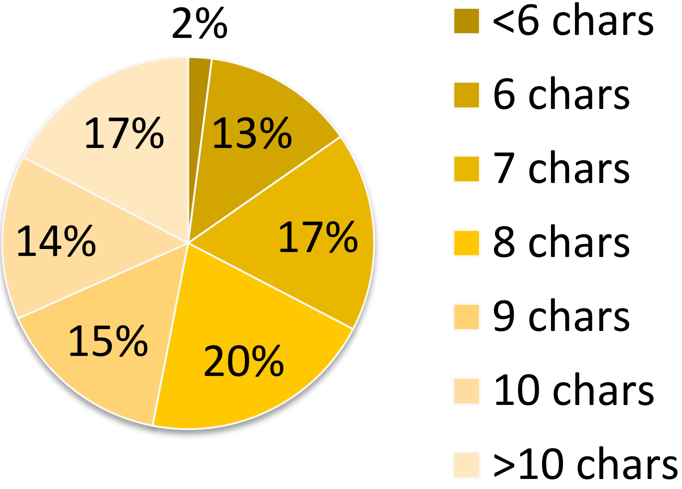
SHA512
9 days

Analyzing leaked passwords

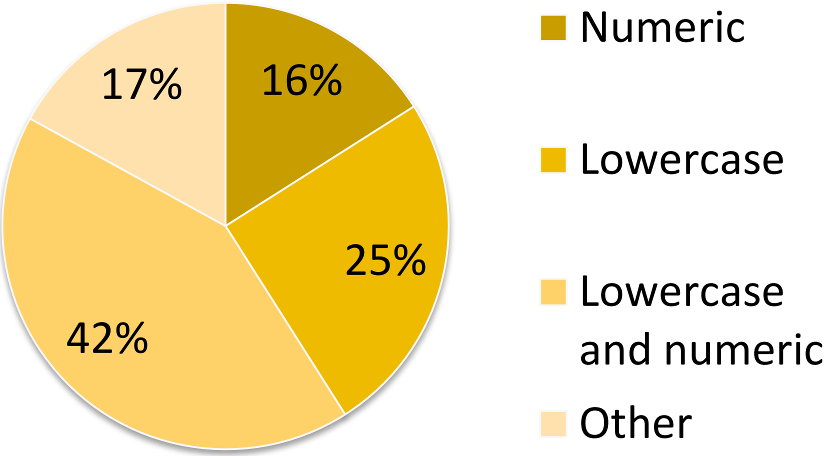
rockyou

Breached in 2009: 14.3 million plaintext passwords leaked

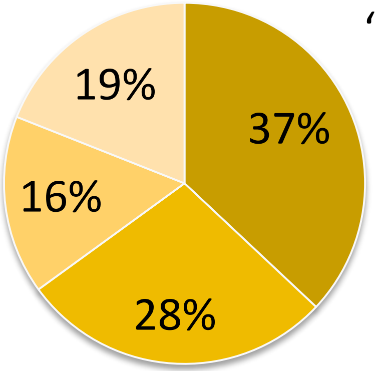
Password Length



Password Complexity



Smarter Analysis



- ■ All Numbers
- ■ All Letters
- ■ Letters then Numbers
- ■ Other

"LD" Pattern:

- All lowercase or all numbers
- Lowercase with last 1-4 characters numeric

1-10 characters alphanumeric

- Recovered: 71%
- MD5: 3 days
- SHA512: 9 days

1-10 characters using pattern

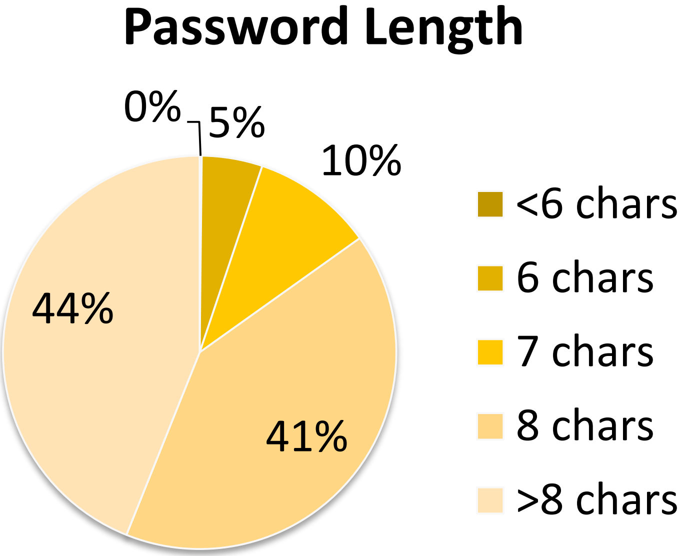
- Recovered: 61%
- MD5: 6h23m
- SHA512: 17h41m

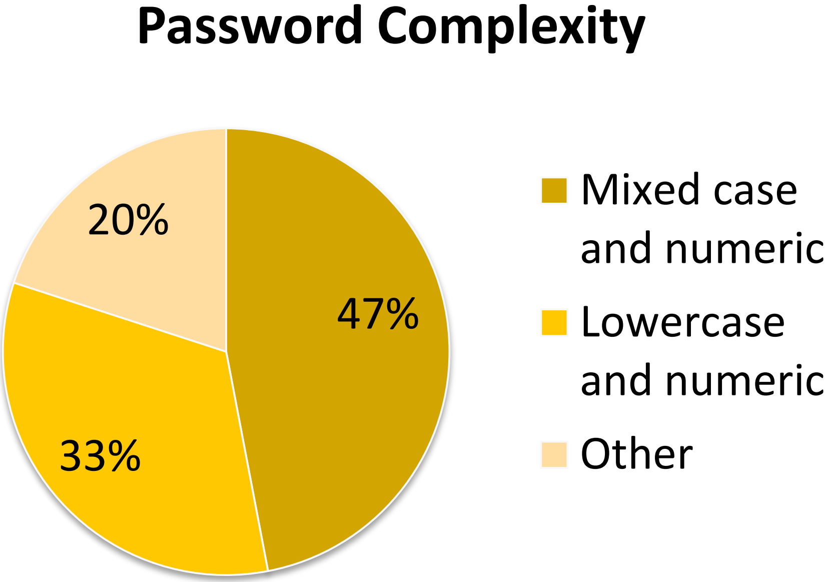
Breaking NTLM for fun and profit

- Old Windows domain authentication system
- Uses very weak hashes
- Hashes are everywhere
- Keys to the Kingdom
- Other security fails besides cracking

Crack and Analyze

8,930/15,902 “stolen” NTLM hashes (< 9 chars)





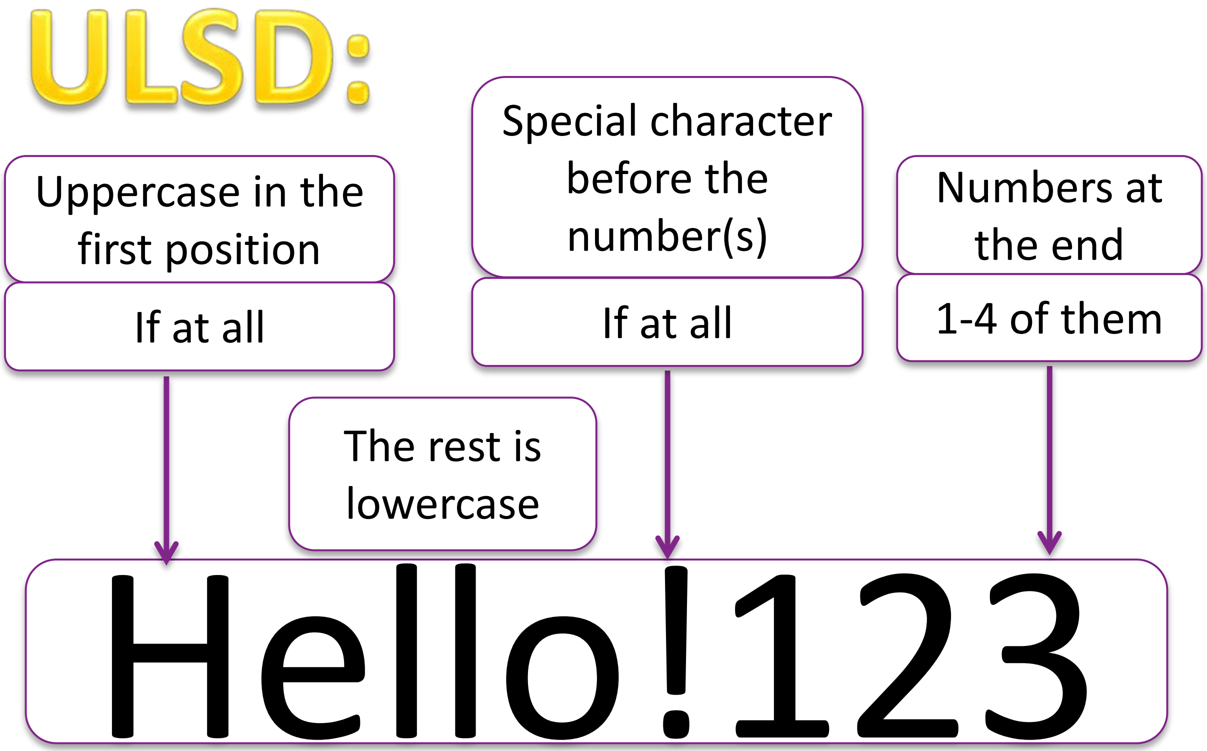
Recurring Themes

Variations on Company name

Variations on "P@5$w0rd"

Likely IT defaults that never got changed

A pattern emerges…



ULSD:

Uppercase in the first position

If at all

Special character before the number(s)

If at all

Numbers at the end

1-4 of them

The rest is lowercase

Hello!123

Live Fire - Patterns

15,902 NTLM hashes “stolen” in penetration tests

Fast

Method	Recovered	Time
All to 7 chars	15%	10 mins
ULSD 8 chars	12%	1 min
ULSD 9 chars	5%	12 mins
LD 10 chars	2%	44 mins
Total	**35%**	**~1 hour**

Thorough

Method	Recovered	Time
All to 8 chars	56%	17 hours
ULSD 9 chars	5%	12 mins
ULSD 10 chars	4%	6 hours
LD to 11 chars	1%	19 hours
Total	**67%**	**~41 hours**

Live Fire – Adding Dictionaries

15,902 NTLM hashes “stolen” in penetration tests

Method	Recovered	Time
149 million dictionary words with Best64 rule	24.6%	53s
149 million dictionary words with d3ad0ne rule	**44.1%**	**5m**

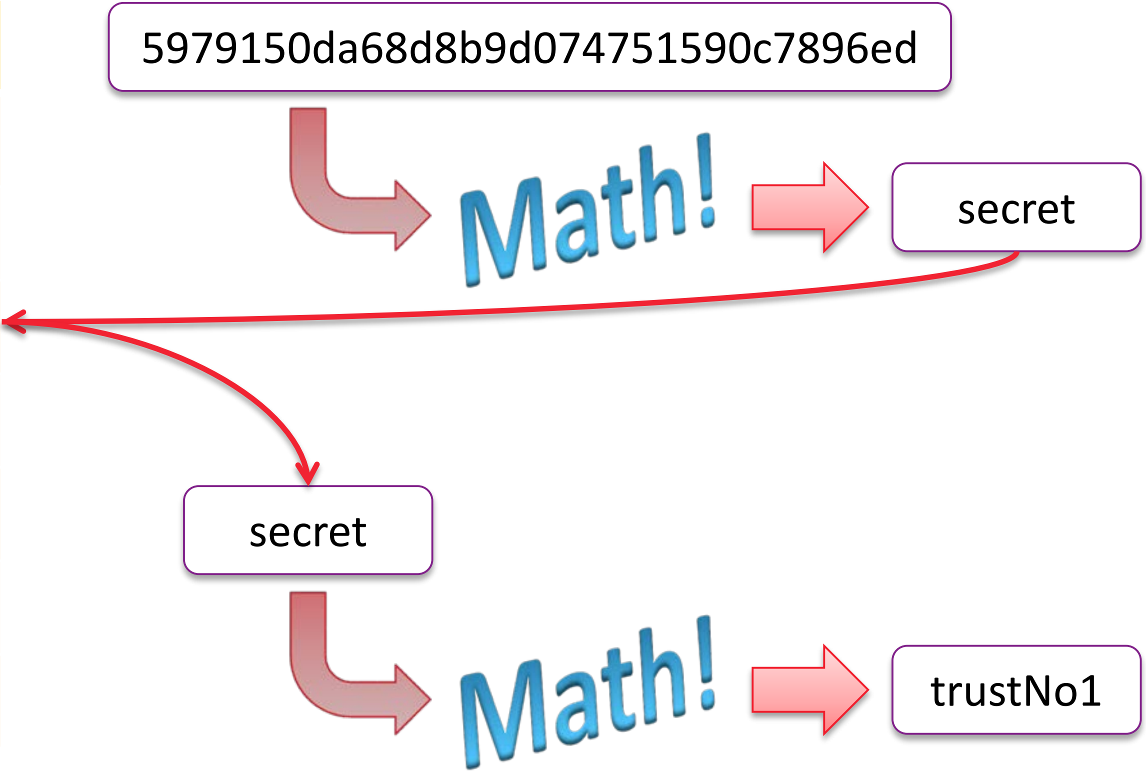
Fast (<1 hour)	Recovered
No dictionary	35.0%
With dictionary	**47.7%**

Thorough (<2 days)	Recovered
No dictionary	67%
With dictionary	**73.9%**

Advanced Hashing and Cracking

Rainbow Tables (Horribly Oversimplified)

Start	End
aaaaaa	abcabc
bbbbbb	kitten
cccccc	secret
dddddd	sesame
eeeeee	random
ffffff	archer
…	…



5979150da68d8b9d074751590c7896ed

Math!

secret

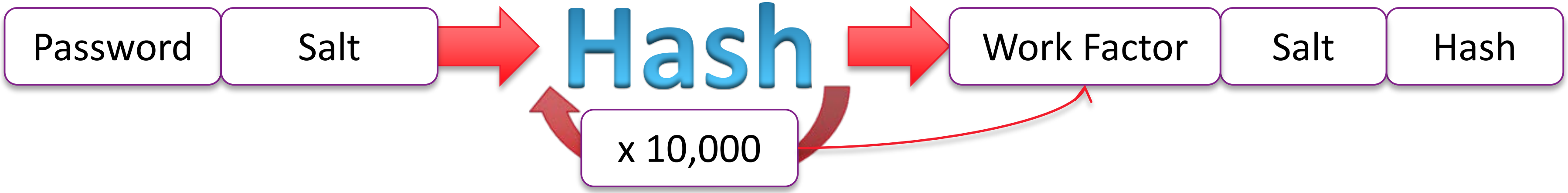
secret

Math!

trustNo1

Better Hashing

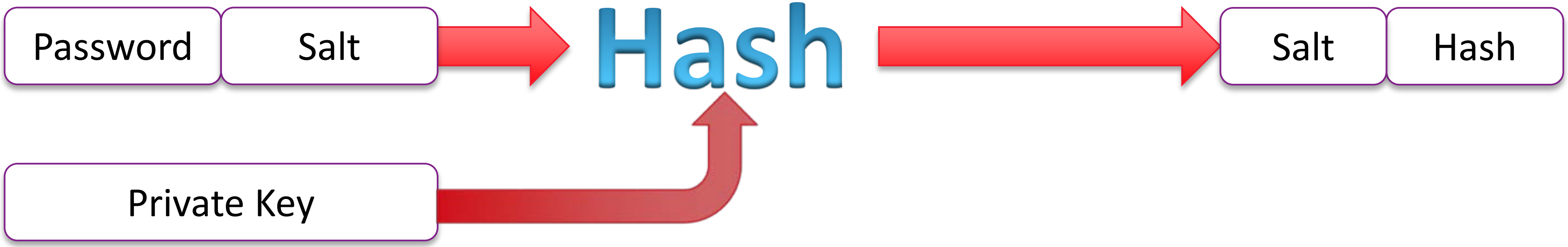
Key Derivation Functions (KDFs)



Password | Salt → Hash → Work Factor | Salt | Hash

x 10,000

Hash-based Message Authentication Codes (HMACs)



Password | Salt → Hash → Salt | Hash

Private Key

Making Trouble

Making Trouble

Access or escalate privileges on a target network

Impersonate a user for fraud

Publicly post them to embarrass a target

Add them to cracking dictionary

Recommendations

Keep Hashes Safe

Strong SDLC for custom apps

Lock down Windows security configuration

Use admin credentials only when necessary

Penetration test to find weaknesses

Strong Password Policy

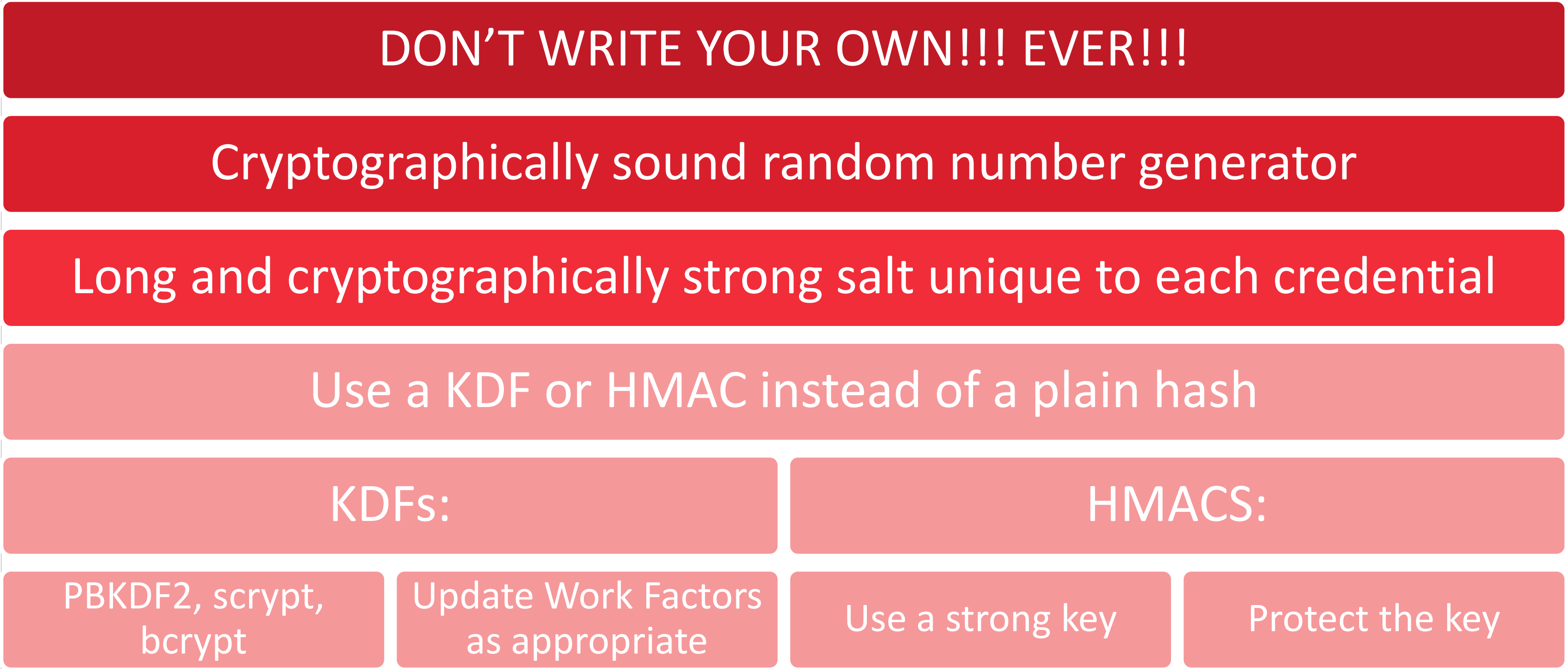
Enforce password requirements

Change <90 days	12+ characters	All character types	Prohibit re-use	Pattern checks?

Support

Crack your own passwords	Awareness of phishing and re-use

Use Appropriate Crypto



DON’T WRITE YOUR OWN!!! EVER!!!

Cryptographically sound random number generator

Long and cryptographically strong salt unique to each credential

Use a KDF or HMAC instead of a plain hash

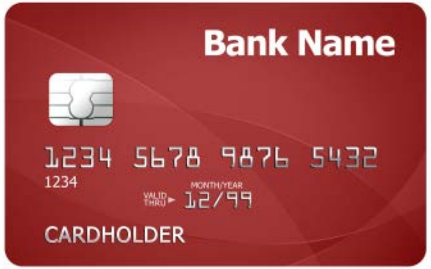
KDFs:

- PBKDF2, scrypt, bcrypt
- Update Work Factors as appropriate

HMACS:

- Use a strong key
- Protect the key

When it really needs to be secure



Something You Know

PIN

Password

Something You Have

Token Card

Certificate File

Don’t Muck It Up

- Broken authentication and session management
- Password reset procedures
- Leaking plaintext passwords
- Users with the same password on every site
- Users who fall for phishing
- Malware and keyloggers

Wrapping Up

Apply this knowledge

Next week you should:

- Change YOUR password to something long, complex, and unique to each service
- Do some Google searches for your own company's code and passwords (e.g. GitHub)

In the first three months following this presentation you should:

- Implement a better password policy and enforce it
- Look for incorrect salt usage, use of plain hashes, and weak crypto, and unnecessary backwards-compatibility settings

Within six months you should:

- Disable as much backwards compatibility and outdated crypto as possible
- Use salted KDF or HMACs for all password authentication
- Implement 2-factor or other password alternatives where appropriate

Contact Details

NTT Communications | NTT Com Security

chris.camejo@nttcomsecurity.com - @0x434a