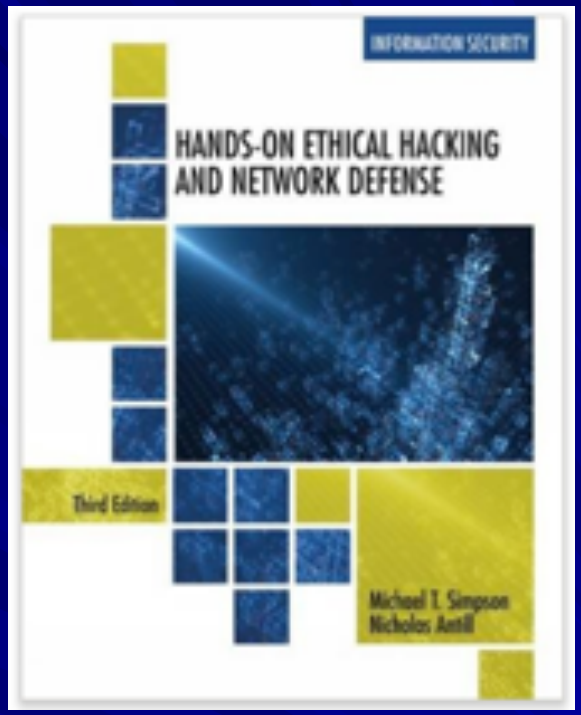
THOMSON
COURSE TECHNOLOGY

# Hands-On Ethical Hacking and Network Defense 3rd Edition




## *Chapter 12 Cryptography*

Last modified 11-22-17 12:15 pm

# Overview

- Mathematical Basis
  - Symmetric encryption
  - Asymmetric encryption
  - Hashing
- Implementation
  - Public-Key Infrastructure
  - Cryptographic Attacks

# Symmetric Encryption

# Understanding Cryptography Basics

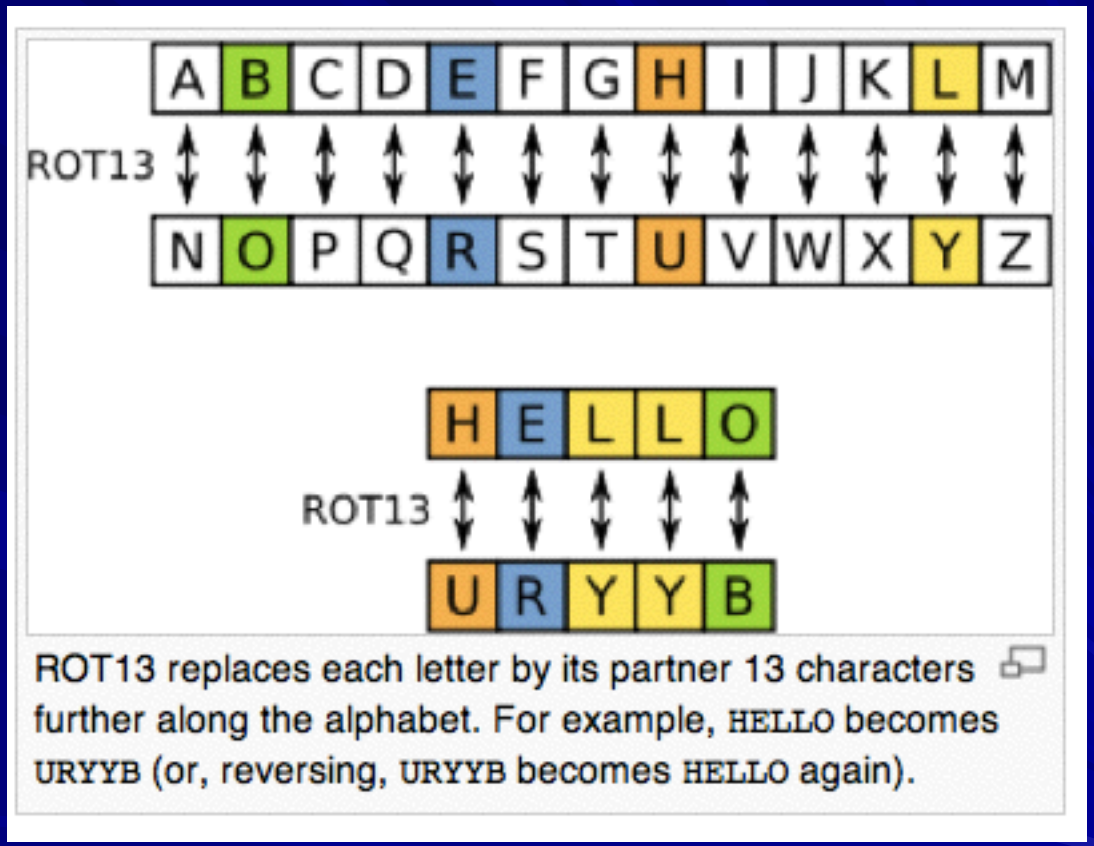
- Cryptography is the process of converting plaintext into ciphertext
  - Plaintext: readable text (also called cleartext)
  - Ciphertext: unreadable or encrypted text
- Cryptography is used to hide information from unauthorized users
- Decryption is the process of converting ciphertext back to plaintext

# History of Cryptography

- Substitution cipher
  - Replaces one letter with another letter based on a key
  - Example: Julius Caesar's Cipher
    - Used a key value of 3
    - **ABCDEFGHIJKLMNOPQRSTUVWXYZ**
    - **DEFGHIJKLMNOPQRSTUVWXYZABC**

# ROT-13

- A Caesar cipher
- Performing ROT-13 twice undoes it
- Obfuscation, not Encryption
- From Wikipedia




ROT13 replaces each letter by its partner 13 characters further along the alphabet. For example, HELLO becomes URYYB (or, reversing, URYYB becomes HELLO again).

# Demonstration

```
Sams-MacBook-Pro-3:~ sambowne$ python caesar.py
MESSAGE TO ENCRYPT: HELLO
Shift number: 3

Result:  KHOOR
Sams-MacBook-Pro-3:~ sambowne$ python caesar.py
MESSAGE TO ENCRYPT: HELLO
Shift number: 13

Result:  URYYB
Sams-MacBook-Pro-3:~ sambowne$ python caesar.py
MESSAGE TO ENCRYPT: URYYB
Shift number: 13

Result:  HELLO
Sams-MacBook-Pro-3:~ sambowne$
```

# Demonstration

```
  GNU nano 2.0.6                    File: caesar.py

alp = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

plaintext = raw_input("MESSAGE TO ENCRYPT: ")
shift = int(raw_input("Shift number: "))

n = len(plaintext)
nalp = len(alp)

ciphertext = ""
for i in range(n):
  loc = alp.find(plaintext[i])
  newloc = (loc + shift) % nalp
  ciphertext += alp[newloc]

print
print "Result: ", ciphertext
```

# History of Cryptography (continued)

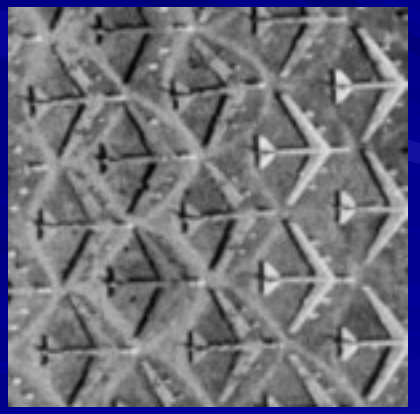
- Cryptanalysis studies the process of breaking encryption algorithms
- When a new encryption algorithm is developed, cryptanalysts study it and try to break it
  - Or prove that it is impractical to break it (taking much time and many resources)

# Enigma

- Used by the Germans during World War II
  - Replaced letters as they were typed
  - Substitutions were computed using a key and a set of switches or rotors
  - Image from Wikipedia (link Ch 12a)

# Steganography

- The process of hiding data in plain view in pictures, graphics, or text
  - Example: changing colors slightly to encode individual bits in an image
- The image on the left contains the image on the right hidden in it (link Ch 12c)



# Algorithms

- An algorithm is a mathematical function or program that works with a key
- Security comes from
  - A strong algorithm—one that cannot be reversed without the key
  - A key that cannot be found or guessed

# Keys
## (not in textbook)

- A sequence of random bits
  - The range of allowable values is called a *keyspace*
- The larger the *keyspace*, the more secure the key
  - 8-bit key has $2^8 = 256$ values in *keyspace*
  - 24-bit key has $2^{24}$ = 16 million values
  - 56-bit key has $2^{56} = 7 \times 10^{16}$ values
  - 128-bit key has $2^{128} = 3 \times 10^{38}$ values

# Brute Force
(not in textbook)

- In 1997 a 56-bit key was broken by brute force
  - Testing all possible 56-bit keys
  - Used 14,000 machines organized via the Internet
  - It took 3 months
  - See link Ch 12d

# How Many Bits Do You Need?

(not in textbook)

- How many keys could all the computers on Earth test in a year?
  - Pentium 4 processor: $10^9$ cycles per second
  - One year = $3 \times 10^7$ seconds
  - There are less than $10^{10}$ computers on Earth
    - One per person
  - $10^9 \times 3 \times 10^7 \times 10^{10} = 3 \times 10^{26}$ calculations
  - 128 bits should be enough ($3 \times 10^{38}$ values)
    - Unless computers get *much* faster, or someone breaks the algorithm

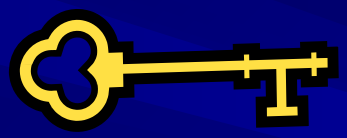
# But if Moore's Law Continues

(not in textbook)

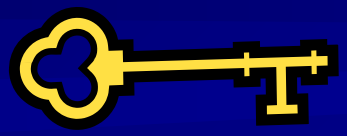
- Suppose computers double in speed every 2 years (link Ch 12zi)
  - 1000x faster every 20 years
- 2010:$10^{27}$ calcs/year 90 bits
- 2030:$10^{30}$ calcs/year 100 bits
- 2050:$10^{33}$ calcs/year 110 bits
- 2070:$10^{36}$ calcs/year 120 bits
- 2090:$10^{39}$ calcs/year 130 bits
  - 128 bits may not be enough ($3 \times 10^{38}$ values)

# Symmetric Cryptography

- One key encrypts and decrypts data
- Cleartext with Key makes Ciphertext

Winning Lotto #s:  aWDHOP#@-w9

- Ciphertext with Key makes Cleartext

aWDHOP#@-w9  Winning Lotto #s:

# Symmetric Cryptography Algorithms

- Symmetric algorithms have one key that encrypts and decrypts data
- Advantages
  - Symmetric algorithms are fast
  - They are difficult to break if a large key size is used
  - Only one key needed

# Symmetric Cryptography Algorithms

- Disadvantages
  - Symmetric keys must remain secret
  - Difficult to deliver keys (key distribution)
  - Symmetric algorithms don't provide *authenticity* or *nonrepudiation*
    - You can't know for sure who sent the message, since two people have the same key

# Symmetric Cryptography Algorithms

- Types of symmetric algorithms
  - Stream ciphers
    - Operate on plaintext one bit at a time
  - Block ciphers
    - Operate on blocks of plaintext

# DeCSS

- Commercial DVDs are encoded with a 40-bit key
  - It's simple to crack it by brute force
  - Three hackers did that in 1999
    - See links Ch 12e, 12f
  - Legislation such as the DMCA made it illegal to publish the algorithm
    - See Illegal Prime Number (Link Ch 12g)

# Data Encryption Standard (DES)

- National Institute of Standards and Technology (NIST)
  - Wanted a means of protecting sensitive but unclassified data
  - Invited vendors in early 1970 to submit data encryption algorithms
- IBM proposed Lucifer
  - A 128-bit encryption algorithm

# Data Encryption Standard (DES)

- The National Security Agency (NSA) reduced the key size from 128 bits to 64 bits and created DES
  - Only 56 bits of the key are actually used

# Data Encryption Standard (DES) (continued)

- In 1988, NSA thought the standard was at risk to be broken
- In 1997, a DES key was broken in 3 months
- In 1998, the EFF built a a computer system that cracked a DES key in 3 days
  - Link Ch 12h

# Demonstration

```
Sams-MacBook-Pro-3:~ sambowne$ python
Python 2.7.14 (default, Sep 25 2017, 09:53:17)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from Crypto.Cipher import DES
>>> key = "Secret!!"
>>> cipher = DES.new(key)
>>> c = cipher.encrypt("RUN FAST")
>>> print c.encode("hex")
2813f621a40c4699
>>> cipher.decrypt(c)
'RUN FAST'
```

# Triple DES (3DES)

- Triple Data Encryption System (3DES)
- 3DES served as a quick fix to the vulnerabilities of DES
- 3DES performs three DES encryptions
- $2^{56}$ times stronger than DES
  - More secure but slower to compute
    - See link Ch 12i

# Advanced Encryption Standard (AES)

- Became effective in 2002 as a standard
  - The process took 5 years
- Block cipher that operates on 128-bit blocks of plaintext
- Keys can be 128, 192, or 256 bits
- Uses Rindjael algorithm
  - Link Ch 12j

# Demonstration

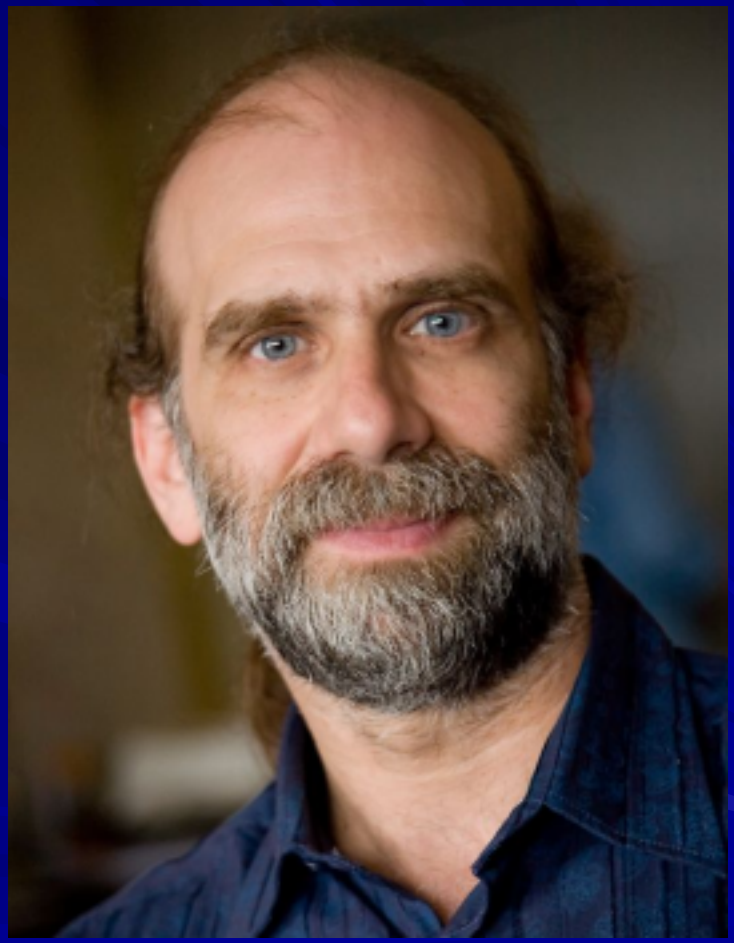
```
Sams-MacBook-Pro-3:~ sambowne$ python
Python 2.7.14 (default, Sep 25 2017, 09:53:17)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from Crypto.Cipher import AES
>>> key = "16 bytes long..."
>>> cipher = AES.new(key)
>>> c = cipher.encrypt("Pigeons plotting")
>>> print c.encode("hex")
92139a8d58347d3797bba14ec9203ec4
>>> cipher.decrypt(c)
'Pigeons plotting'
```

# International Data Encryption Algorithm (IDEA)

- Block cipher that operates on 64-bit blocks of plaintext
- It uses a 128-bit key
- Developed by Xuejia Lai and James Massey
  - Designed to work more efficiently in computers used at home and in businesses
- IDEA is free for noncommercial use
  - It is included in PGP encryption software

# Blowfish

- Block cipher that operates on 64-bit blocks of plaintext
- The key length can be as large as 448 bits
- Developed by Bruce Schneier



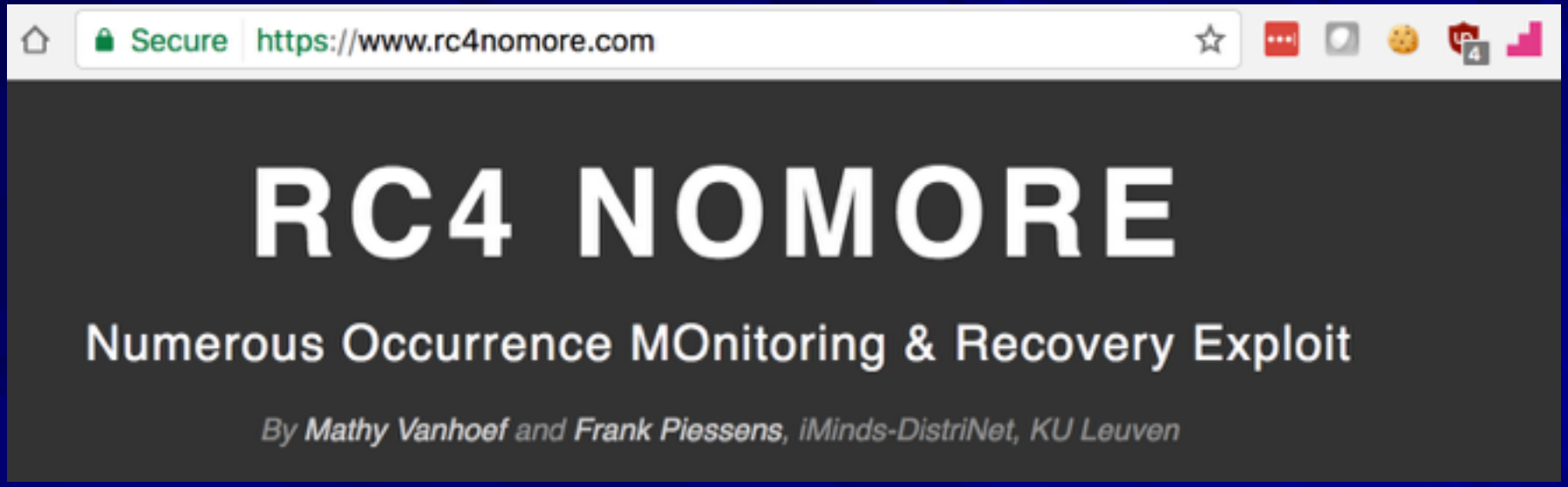
# RC4

**RC4 crypto: Get RID of it already, say boffins**

This one simple attack busts WPA-TKIP in less than an hour

16 Jul 2015 at 04:58, Richard Chirgwin

- The most widely used stream cipher
- Used in WEP and WPA (part of TKIP)
- Also used in some versions of TLS
  - Link Ch 12zz1




- Weakness: RC4 is biased--some byte sequences are more common than others
- Can decrypt an HTTPS cookie in 75 hours
  - If RC4 is used
- Can break WPA-TKIP within an hour

# RC5

- Block cipher that can operate on different block sizes: 32, 64, and 128
- The key size can reach 2048 bits
- Created by Ronald L. Rivest in 1994 for RSA Data Security

# Cracking RC5

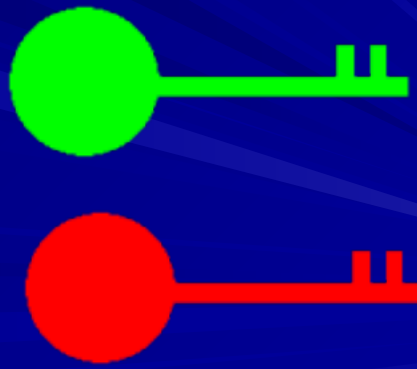
- 56-bit and 64-bit key RC5s have already been cracked
- The RC5-72 project is underway, trying to crack a 72-bit key
  - At the current rate, it will take 1000 years
    - Links Ch 12l, 12m


Kahoot!

# Asymmetric Encryption

# Asymmetric Cryptography Algorithms

- Use two keys that are mathematically related
  - Data encrypted with one key can be decrypted only with the other key
- Another name for asymmetric key cryptography is public key cryptography
  - Public key: known by the public
  - Private key: known only by owner



# Public-Key Encryption

- Mailboxes work this way
- Anyone can put mail in
- Only the postal worker with the **private key** can take mail out
  - Image is free clip art from clipartpanda.com

# Asymmetric Cryptography

- Cleartext with Public Key makes Ciphertext

Winning Lotto #s:

aWDHOP#@-w9

- Ciphertext with Private Key makes Cleartext

aWDHOP#@-w9

Winning Lotto #s:

# Asymmetric Cryptography

- Provides message authenticity and nonrepudiation
  - Authenticity validates the sender of a message
  - Nonrepudiation means a user cannot deny sending a message

# Asymmetric Cryptography

- Asymmetric algorithms are more scalable but slower than symmetric algorithms
  - Scalable: can adapt to larger networks
  - Each person needs only one key pair
    - Everyone can use the same public key to send you data
    - Each person signs messages with their own private key

# RSA

- Developed in 1977 by Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman
- The algorithm is based on the difficulty of factoring large numbers
- The Secure Socket Layer (SSL) protocol uses the RSA algorithm

Ron Rivest

# RSA Encryption in Python

```
>>> from Crypto.PublicKey import RSA
>>> key = RSA.generate(2048)
>>> publickey = key.publickey()
>>> plain = 'encrypt this message'
>>> ciphertext = publickey.encrypt(plain, 0)[0]
>>> print ciphertext.encode("hex")
536eda071ab9e526442f2b56e71fa5abfc603c88c2eac03d91f22bab6d0ea14bab2e8c8247df477c
5f15ce3ccc551227799d1f4f8943fa8bd278639bd90292c5799d11f9f6601c94d88f10fc314317fb
1d75f55e20d1c5dd4e7448ff39018dab44091b6664610657516bfaf95a3f0e63e9194f1e08343421
f7cf8c35550ed951b240e4c42f94b8bfc73ec3ccd519f7c489c28aaf799c78d6a695707423f72c05
4edfd8f4c2ac0f5c25a996647b8958f160983db8bdf2214fe131b0f3d558aeb7560e67f0621f0224
fd21f18034eebb9c8773e6310f80975539765d7235235a446f037179e94e504b21f9ffac6679570a
95848f238cdd3243723ed4722e549498
```

# RSA Decryption in Python

```
>>> decrypted = key.decrypt(ciphertext)
>>> print decrypted
encrypt this message
```

# Diffie-Hellman

- Developed by Whitfield Diffie and Martin Hellman
- Does not provide encryption but is used for key exchange
  - Two parties agree on a key without ever sending it directly over the network
  - The numbers transmitted can be used to compute the key, but only by the parties holding secret private numbers
- Prevents sniffing attacks

Whitfield Diffie

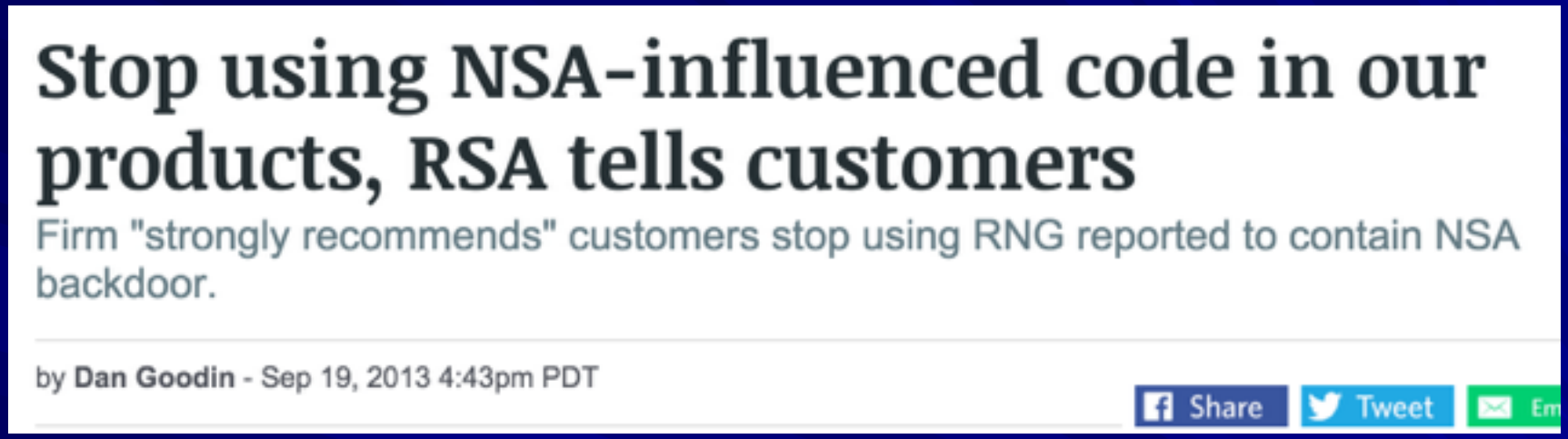
# Elliptic Curve Cryptosystems (ECC)

- It is an efficient algorithm requiring few resources
  - Memory
  - Disk space
  - Bandwidth
- ECC is used for encryption as well as digital signatures and key distribution

# Elgamal

- Public key algorithm used to
  - Encrypt data
  - Create digital signature
  - Exchange secret keys
- Written by Taher Elgamal in 1985
- The algorithm uses discrete logarithm problems
  - Solving a discrete logarithm problem can take many years and require CPU-intensive operations

# NSA & ECC



## Stop using NSA-influenced code in our products, RSA tells customers

Firm "strongly recommends" customers stop using RNG reported to contain NSA backdoor.

by **Dan Goodin** - Sep 19, 2013 4:43pm PDT

Share Tweet Em

- NSA apparently backdoor the random number generator
- Goal: NOBUS (Nobody But Us)
- Link Ch 12zw, 12zx, 12zy

# Silent Circle ditches NIST cryptographic standards to thwart NSA spying


- Link Ch 12zz

# NSA & Quantum Computing


## CNSA Suite and Quantum Computing FAQ

**Q: Doesn't CNSSP-15 require all commercial NSS acquisitions to incorporate Suite B elliptic curve algorithms by October 2015?**

A: Prior to the release of CNSS Advisory Memorandum 02-15 in August 2015 it did. That was an important consideration in the timing of the memorandum. CNSS Advisory Memorandum 02-15 removes that requirement. CNSSP-15 is being updated and will take some time to publish. In the interim, CNSS Advisory Memorandum 02-15 describes the most up-to-date algorithm guidance. See the advisories tab at www.cnss.gov.

# NSA & Quantum Computing

In more precise terms this means that NSS should no longer use

- ECDH and ECDSA with NIST P-256
- SHA-256
- AES-128
- RSA with 2048-bit keys
- Diffie-Hellman with 2048-bit keys

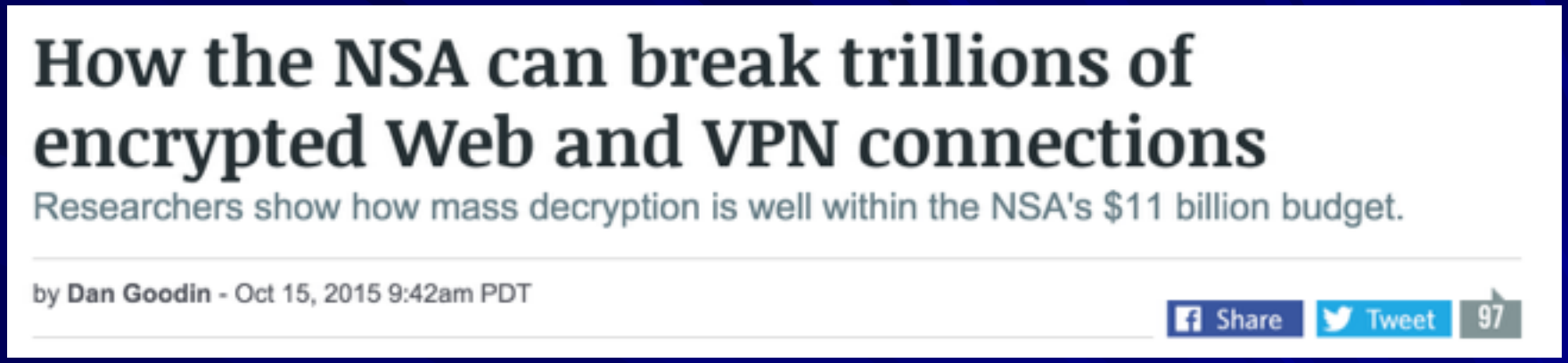
- NSS = National Security System
- Link Ch 12zy

# NSA & Quantum Computing

- The public-key algorithms (RSA, Diffie-Hellman, ECDH, and ECDSA) are all vulnerable to attack by a sufficiently large quantum computer.
- Quantum computing techniques are much less effective against symmetric algorithms than against widely used public key algorithms.
- AES-256 and SHA-384 are believed to be safe from attack by a large quantum computer.
  - Link Ch 12zy

# NSA & Quantum Computing

- NSS should now use these key sizes:
  - Elliptic Curves: 384 bits
  - RSA & Diffie-Hellman: 3072 bits
    - Link Ch 12zy



# How the NSA can break trillions of encrypted Web and VPN connections

Researchers show how mass decryption is well within the NSA's $11 billion budget.

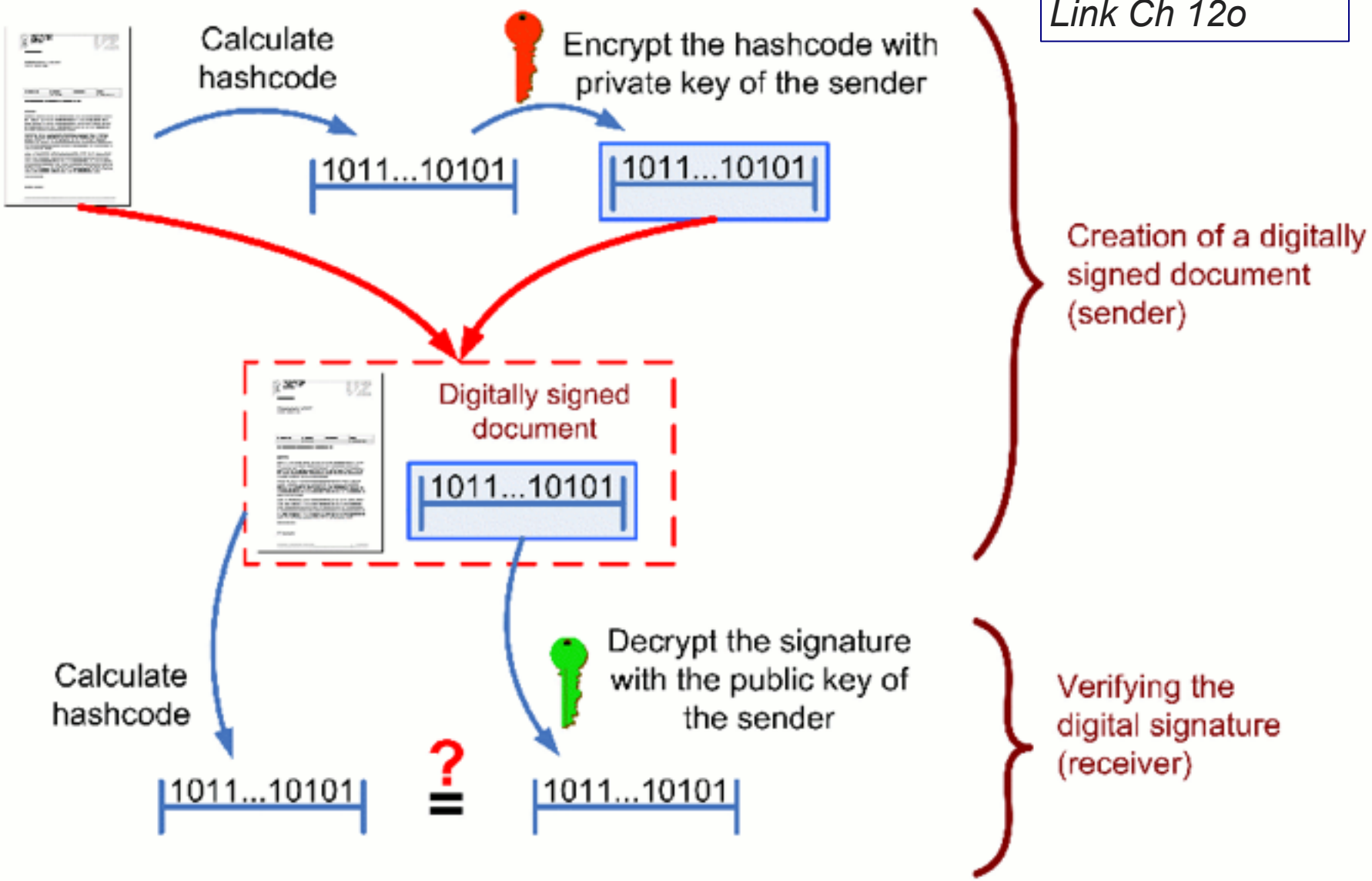
by **Dan Goodin** - Oct 15, 2015 9:42am PDT

Share Tweet 97

- Since a handful of primes are so widely reused, …
- Breaking a single, common 1024-bit prime would allow NSA to passively decrypt connections to two-thirds of VPNs and a quarter of all SSH servers globally.
- Breaking a second 1024-bit prime would allow passive eavesdropping on connections to nearly 20% of the top million HTTPS websites.
  - Link Ch 12zx (Oct, 2015)

# Creating and verifying a digital signature

*From Wikipedia*
*Link Ch 12o*




If the calculated hashcode does not match the result of the decrypted signature, either the document was changed after signing, or the signature was not generated with the private key of the alleged sender.

# Digital Signature Standard (DSS)

- Established by the NIST in 1991
  - Ensures that digital signatures rather than written signatures can be verified
- Federal government requirements
  - RSA and Digital Signature Algorithm (DSA) must be used for all digital signatures
  - Hashing algorithm must be used to ensure the integrity of the message
    - NIST required that the Secure Hash Algorithm (SHA) be used

# Pretty Good Privacy (PGP)

- Developed by Phil Zimmerman as a free e-mail encryption program
  - Zimmerman was almost arrested for his innovation
  - Back in the mid-1990s, any kind of “unbreakable” encryption was seen as a weapon and compared to selling arms to the enemy

# Pretty Good Privacy (PGP)

- PGP is a free public key encryption program
- It uses certificates similar to those in public key infrastructure (PKI)
  - PGP does not use a centralized CA
  - Verification of a certificate is not as efficient as PKI

# Pretty Good Privacy (PGP) (continued)

- Algorithms supported by PGP
  - IDEA
  - RSA
  - DSA
  - Message Digest 5 (MD5)
  - SHA-1

# Secure Multipurpose Internet Mail Extension (S/MIME)

- Is another public key encryption standard used to encrypt and digitally sign e-mail
- Can encrypt e-mail messages containing attachments
- Can use PKI certificates for authentication
- S/MIME version 2 defined in RFC 2311
- S/MIME version 3 defined in RFC 2633

# Privacy-Enhanced Mail (PEM)

- Internet standard that is compatible with both symmetric and asymmetric methods of encryption
- Can use the X.509 certificate standards and encrypt messages with DES
- Not used as much today
  - MIME Object Security Services (MOSS) is a newer implementation of PEM


Kahoot!

# Hashing

# Hashing Algorithms

- Take a variable-length message and produce a fixed-length value called a message digest
- A hash value is equivalent to a fingerprint of the message
  - If the message is changed later, the hash value changes

# Collisions

- If two different messages produce the same hash value, it results in a collision
  - A good hashing algorithm must be collision-free
- MD5 has known collisions
  - It was never approved by NIST for any purpose

# SHA-1

- SHA-1 is one of the most popular hashing algorithms
- Approved by NIST
- Replaced MD5 for decades

```
>>> import hashlib
>>> p = "P@ssw0rd"
>>> hashlib.new('md5', p).hexdigest()
'161ebd7d45089b3446ee4e0d86dbcf92'
>>>
>>> hashlib.new('sha1', p).hexdigest()
'21bd12dc183f740ee76f27b78eb39c8ad972a757'
```

# SHA-1 Collision Found

- Collision found on Feb. 23, 2017
  - Links Ch 12-2017-1, 2, and 3

```
Sams-MacBook-Pro-3:proj14 sambowne$ ls -l sha*
-rw-r--r--@ 1 sambowne  staff  422435 Feb 23  2017 shattered-1.pdf
-rw-r--r--@ 1 sambowne  staff  422435 Feb 23  2017 shattered-2.pdf
Sams-MacBook-Pro-3:proj14 sambowne$ shasum shattered-1.pdf
38762cf7f55934b34d179ae6a4c80cadccbb7f0a  shattered-1.pdf
Sams-MacBook-Pro-3:proj14 sambowne$ shasum shattered-2.pdf
38762cf7f55934b34d179ae6a4c80cadccbb7f0a  shattered-2.pdf
Sams-MacBook-Pro-3:proj14 sambowne$ md5 shattered-1.pdf
MD5 (shattered-1.pdf) = ee4aa52b139d925f8d8884402b0a750c
Sams-MacBook-Pro-3:proj14 sambowne$ md5 shattered-2.pdf
MD5 (shattered-2.pdf) = 5bd9d8cabc46041579a311230539b8d1
```

Google Security Blog

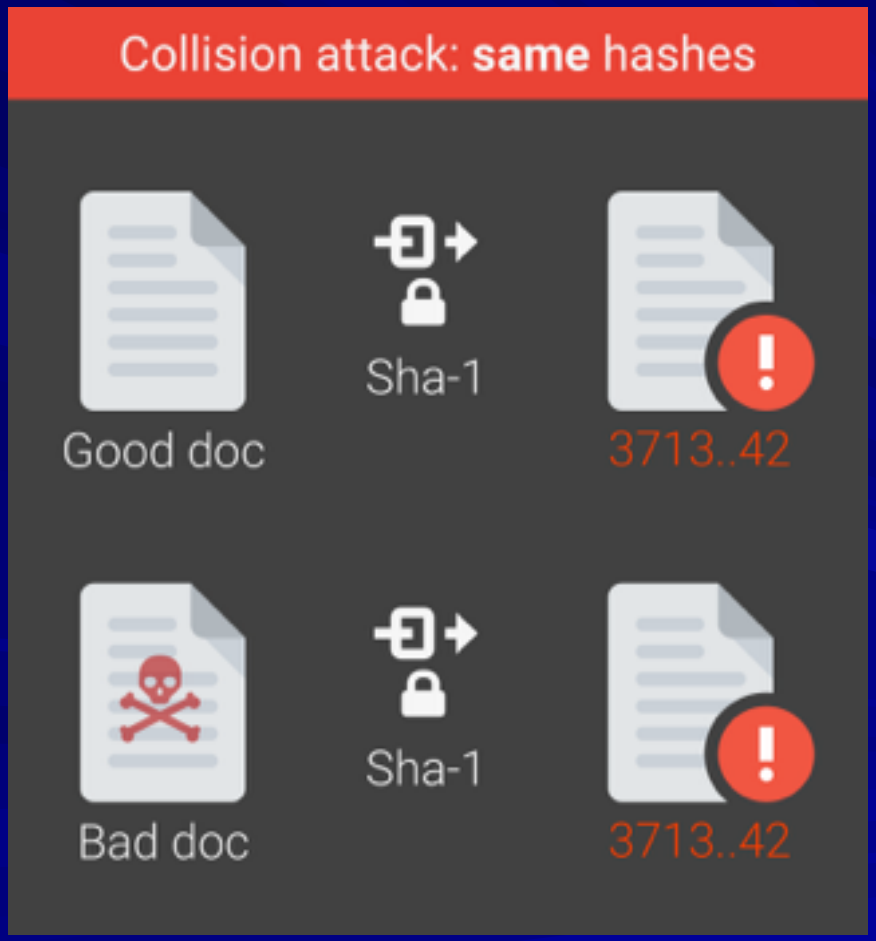
The latest news and insights from Google on security and safety on the Internet

## Announcing the first SHA1 collision

February 23, 2017

**'First ever' SHA-1 hash collision calculated. All it took were five clever brains... and 6,610 years of processor time**

# SHATTERED

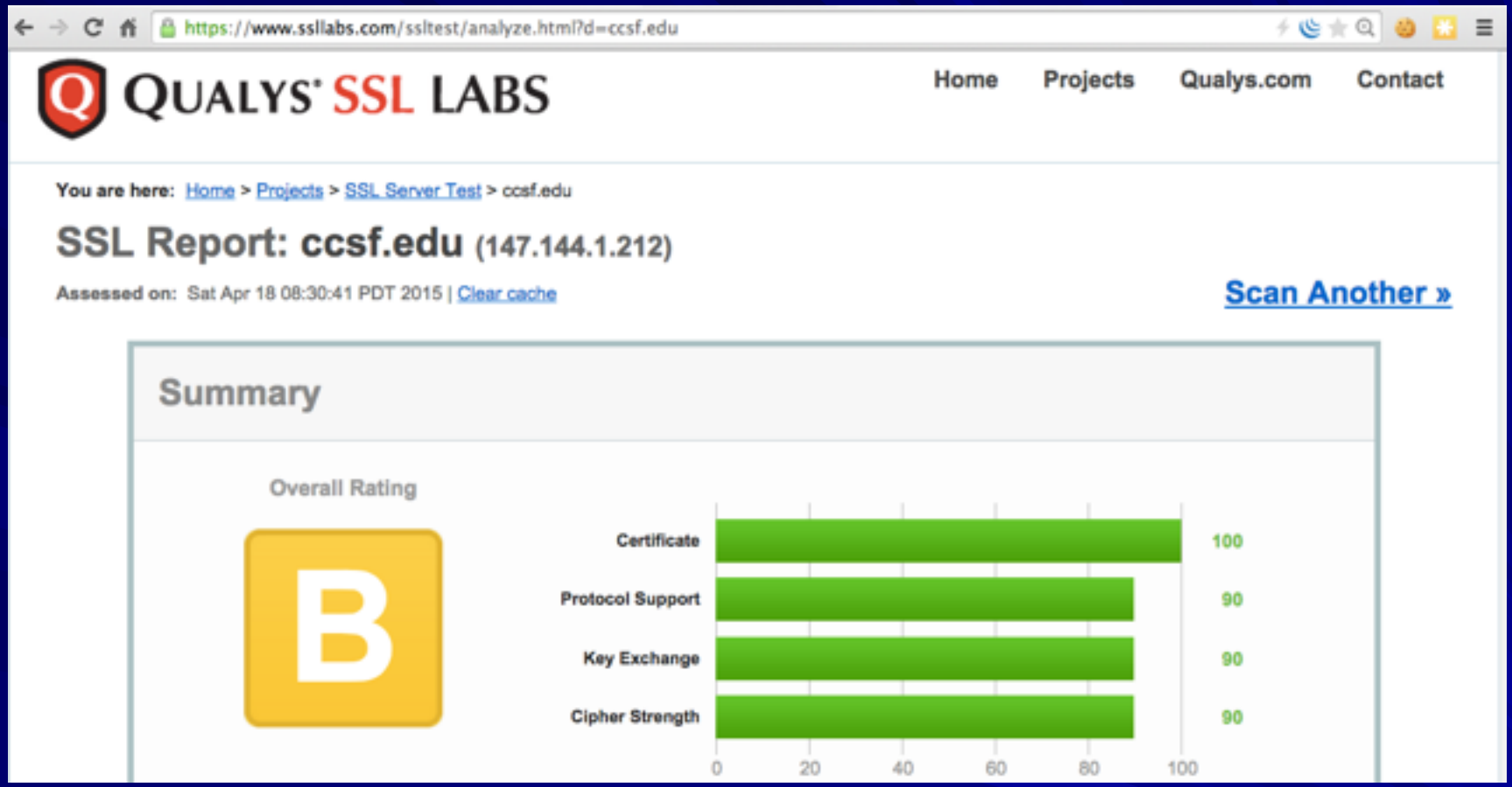

Collision attack: same hashes
Good doc
Sha-1
3713..42
Bad doc
Sha-1
3713..42

# Browsers Deprecated SHA-1

*Microsoft, Google, and Mozilla will begin phasing out trust for SHA-1 certificates in 2016. With these dates approaching, it's time to move to SHA-2.*

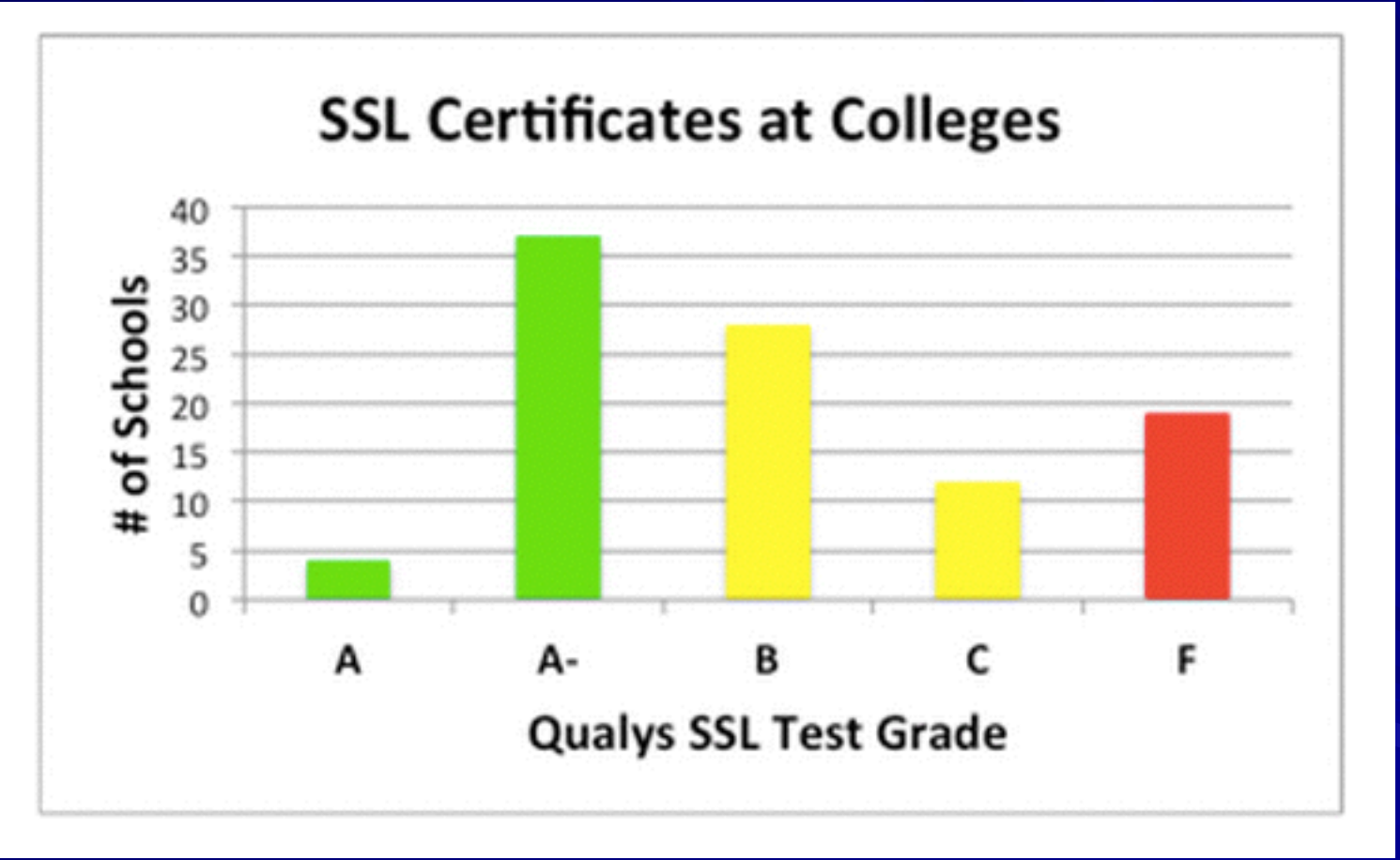
| | |
|---|---|
| ***November 2014 –*** | *SHA-1 SSL Certificates expiring any time in 2017 will show a warning in Chrome.* |
| ***December 2014 –*** | *SHA-1 SSL Certificates expiring after June 1, 2016, will show a warning in Chrome.* |
| ***January 2015 –*** | *SHA-1 SSL Certificates expiring any time in 2016 will show a warning in Chrome.* |
| ***December 2015 –*** | *SHA-1 SSL Certificates issued after January 1, 2016, will show the "untrusted connection" error in Firefox.* |
| ***January 2016 –*** | *SHA-1 SSL Certificates issued after January 1, 2016, will show a certificate error in Chrome.*<br>*Certificate criteria: signed with a SHA-1-base signature, issued after January 1, 2016, and chained to a public CA.* |
| ***January 1, 2017 –*** | *Microsoft, Google, and Mozilla will end trust for all SHA-1 SSL Certificates.*<br>*Mozilla and Google say it is feasible to move this date up to July 1, 2016, in light of recent attacks on SHA-1.*<br>*Microsoft says it is feasible to move this date up to as early as June 2016, in light of recent attacks on SHA-1.* |

- Link Ch 12zr


https://www.ssllabs.com/ssltest/analyze.html?d=ccsf.edu
QUALYS SSL LABS
Home
Projects
Qualys.com
Contact
You are here: Home > Projects > SSL Server Test > ccsf.edu
SSL Report: ccsf.edu (147.144.1.212)
Assessed on: Sat Apr 18 08:30:41 PDT 2015 | Clear cache
Scan Another »
Summary
Overall Rating
B
Certificate
100
Protocol Support
90
Key Exchange
90
Cipher Strength
90
0
20
40
60
80
100

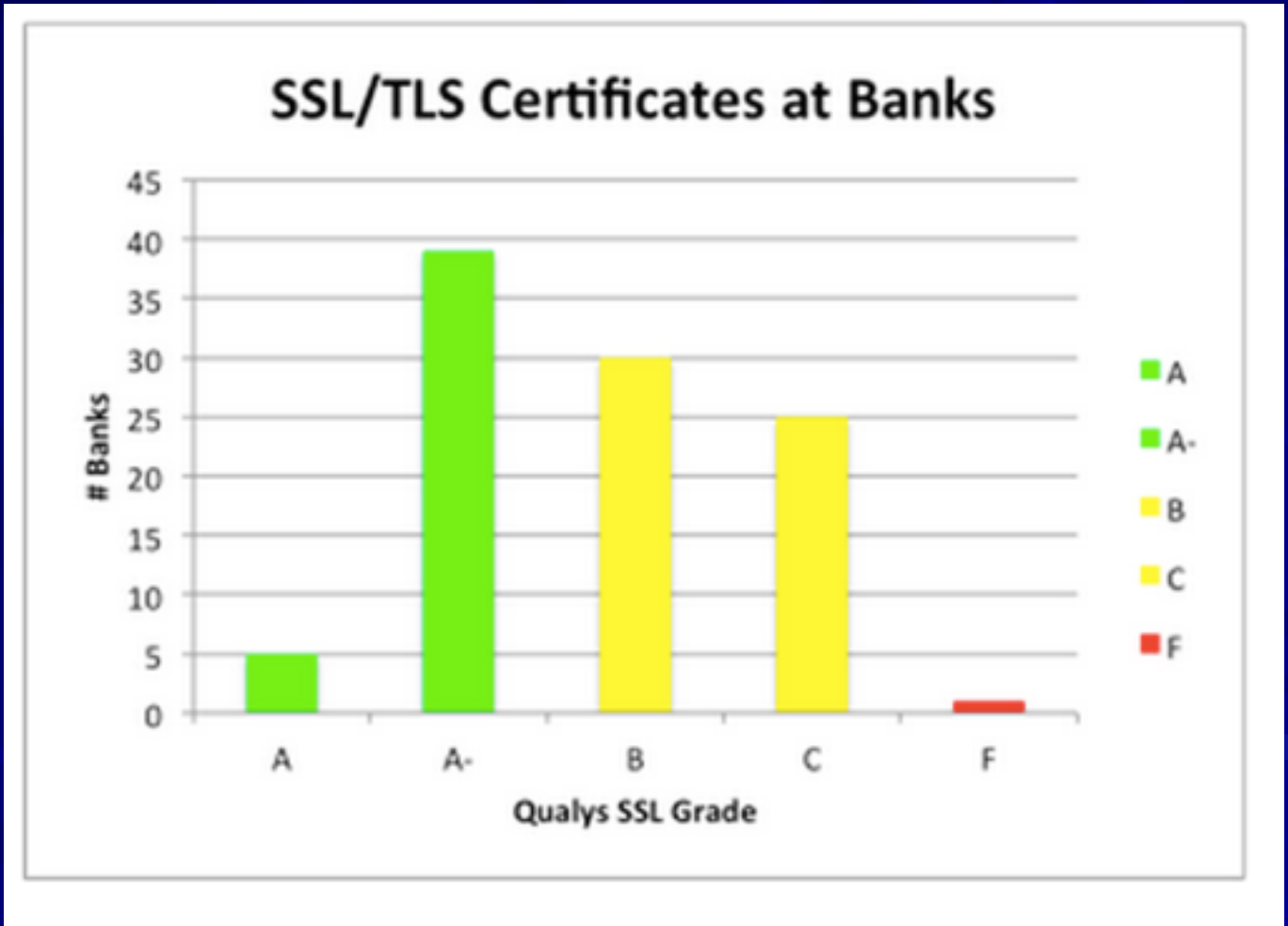
- Link Ch 12zq

# Colleges Tested in 2014



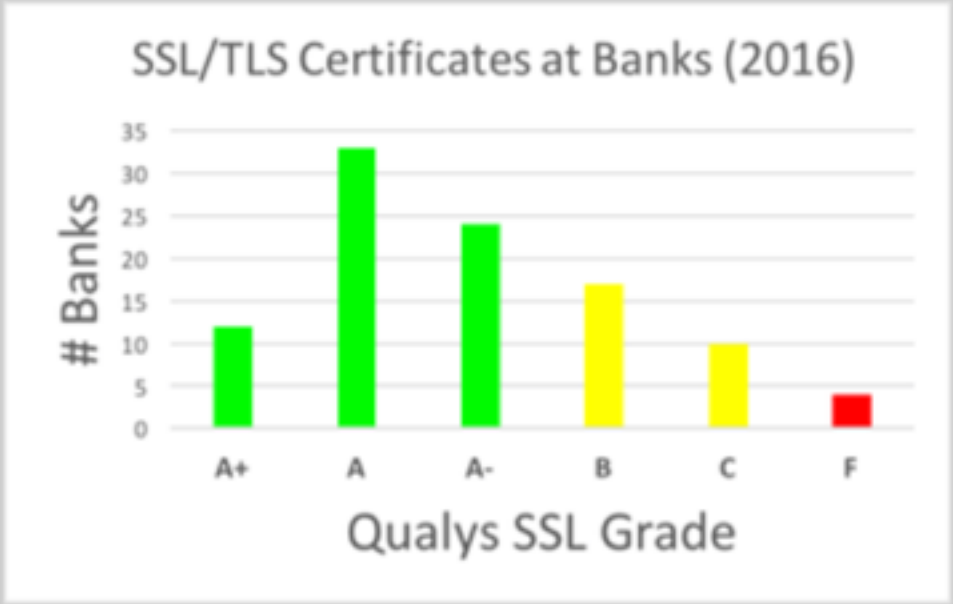

- Link Ch 12zs

# Banks Tested in 2014



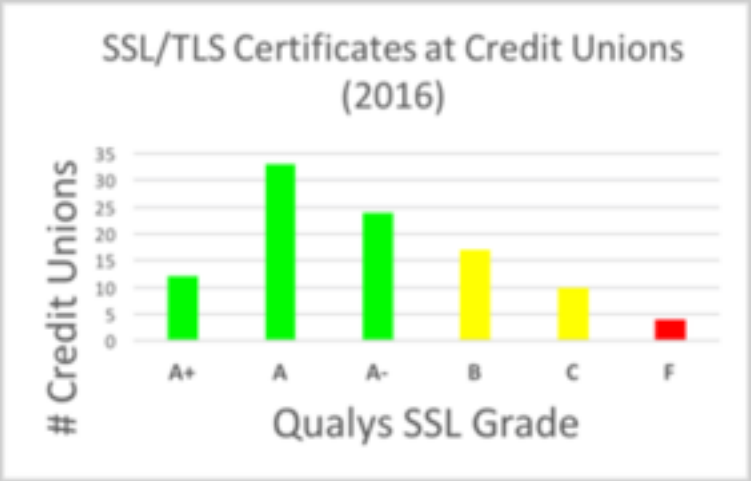

- Link Ch 12zt

# Results from 2016




This time there were four F's:

| | |
|---|---|
| **Middlesex** | https://www.middlesexbank.com/ |
| **Trustco Bank** | https://www.trustcobank.com/ |
| **HSA Bank** | https://secure.hsabank.com/ibanking3/login.aspx |
| **INTRUST Bank** | https://www.intrustbank.com/ |




Here are the five F's:

| | |
|---|---|
| **NCUA.gov** | https://www.ncua.gov/Pages/default.aspx |
| **OneCU** | https://www.onecu.org/ |
| **Guardians CU** | https://www.pbccuvirtual.org/ISuite5/Features/Auth/SelfEnrollment/SelfEnrollmentDisclosure.aspx |
| **Vibrant CU** | https://vibrantcreditunion.org/ |
| **Deere Employees CU** | https://content.dccu.com/ |

# Digital Signatures

- A hash value ensures that the message was not altered in transit (*integrity*)
- Asymmetric encryption assures *authenticity* and *nonrepudiation*

# Symmetric Algorithms (Private-key)

| Name | Key size | Notes |
|---|---|---|
| DES | 56 bits | Insecure |
| 3DES | 168 bits | Being replaced by AES |
| AES | 128,192, or 256 | US Govt classified info |
| IDEA | 128 bits | Used in PGP, very secure |
| Blowfish | 32 to 448 | Public domain |
| RC5 | Up to 2040 | Secure for 72-bits or more |

# Asymmetric Algorithms (Public-key)

| Name | Notes |
| --- | --- |
| Diffie-Hellman | Key exchg, not encryption |
| RSA | Secure, used by SSL |
| ECC | Efficient newer technique |
| Elgamal | Used in GPG and PGP |

# Hashing Algorithms

| Name | Notes |
| --- | --- |
| MD2 | Written for 8-bit machines, no longer secure |
| MD4 | No longer secure |
| MD5 | Security is questionable now |
| SHA-1 | The successor to MD5, Used in:<br>TLS, SSL, PGP, SSH, S/MIME, IPsec<br>No longer completely secure |
| SHA-2 | Not yet broken |
| SHA-3 | Approved by NIST in 2015 |


Kahoot!

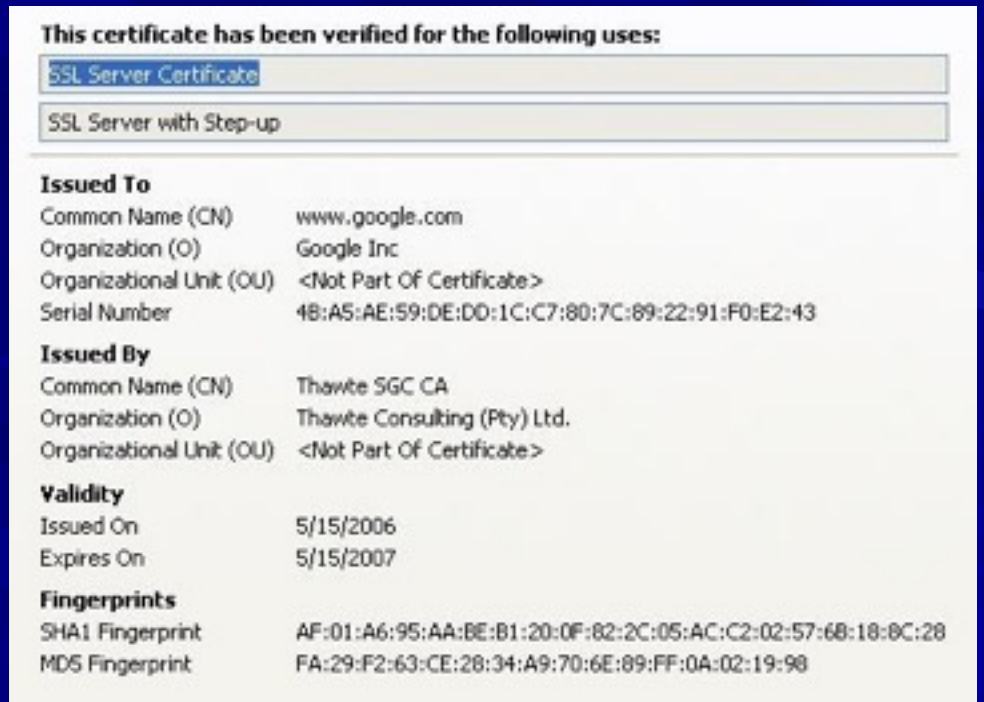
# Public-Key Infrastructure

# Public Key Infrastructure (PKI)

- Not an algorithm
- A structure that consists of programs, protocols, and security protocols
- Uses public key cryptography
- Enables secure data transmission over the Internet

# PKI Components

- Certificate: a digital document that verifies the identity of an entity
  - Contains a unique serial number and must follow the X.509 standard



**This certificate has been verified for the following uses:**

SSL Server Certificate

SSL Server with Step-up

**Issued To**

| | |
|---|---|
| Common Name (CN) | www.google.com |
| Organization (O) | Google Inc |
| Organizational Unit (OU) | <Not Part Of Certificate> |
| Serial Number | 4B:A5:AE:59:DE:DD:1C:C7:80:7C:89:22:91:F0:E2:43 |

**Issued By**

| | |
|---|---|
| Common Name (CN) | Thawte SGC CA |
| Organization (O) | Thawte Consulting (Pty) Ltd. |
| Organizational Unit (OU) | <Not Part Of Certificate> |

**Validity**

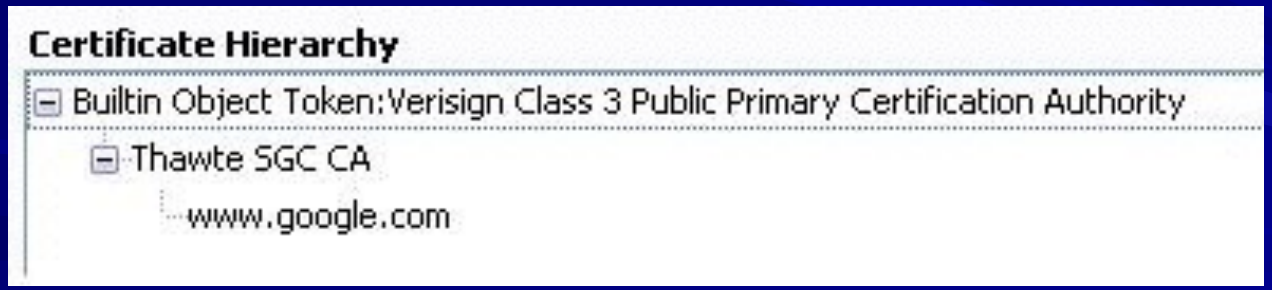
| | |
|---|---|
| Issued On | 5/15/2006 |
| Expires On | 5/15/2007 |

**Fingerprints**

| | |
|---|---|
| SHA1 Fingerprint | AF:01:A6:95:AA:BE:B1:20:0F:82:2C:05:AC:C2:02:57:68:18:8C:28 |
| MD5 Fingerprint | FA:29:F2:63:CE:28:34:A9:70:6E:89:FF:0A:02:19:98 |

# PKI Components

- Public keys are issued by a certification authority (CA)
- A certificate that the CA issues to a company binds a public key to the recipient's private key



# Certificate Expiration and Renewal

- A period of validity is assigned to each certificate
  - After that date, the certificate expires
- A certificate can be renewed with a new expiration date assigned
  - If the keys are still valid and remain uncompromised

# Certificate Revocation and Suspension

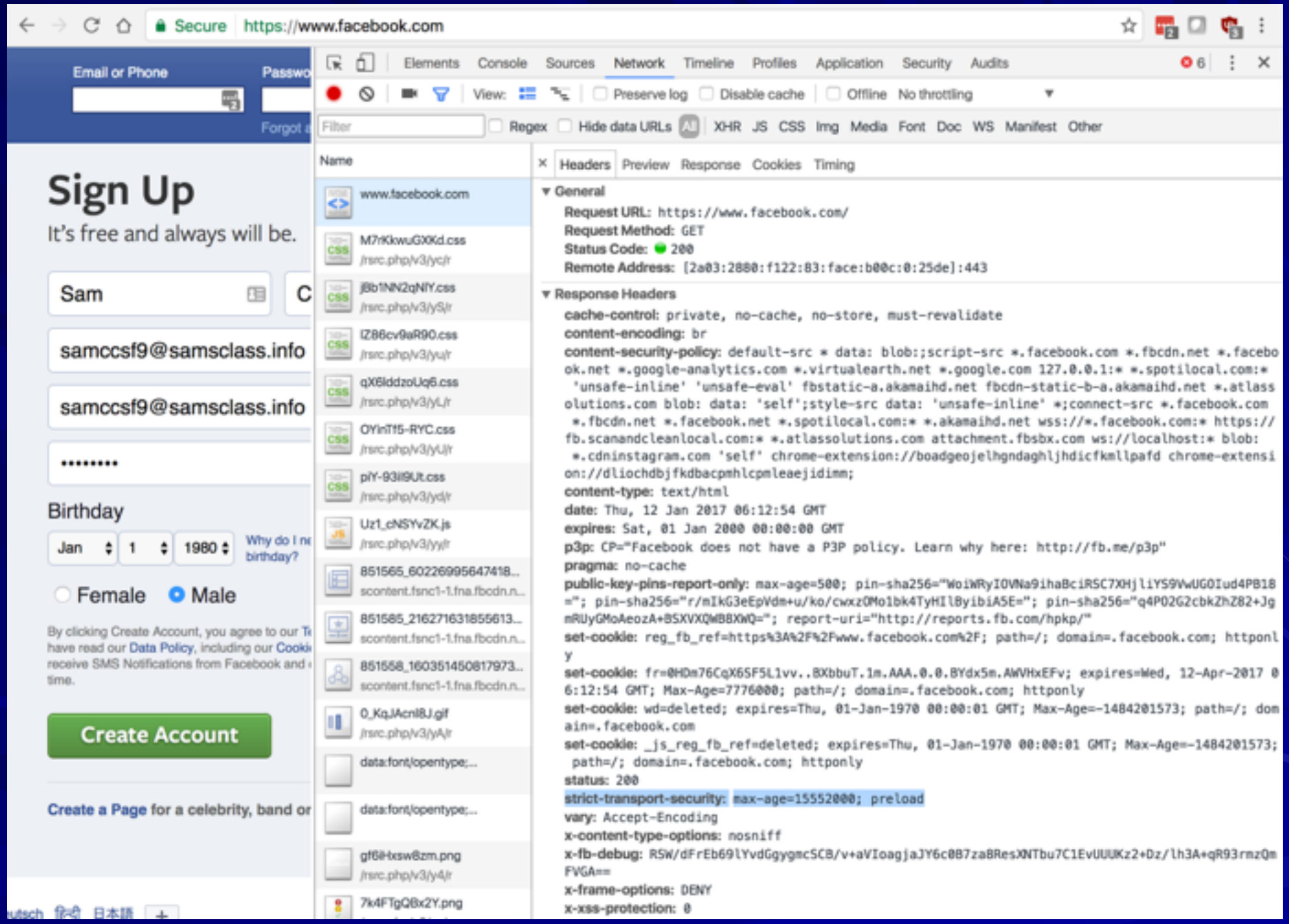
- Reasons to suspend or revoke a certificate
  - A user leaves the company
  - A hardware crash causes a key to be lost
  - A private key is compromised
- Revocation is permanent
- Suspension can be lifted

# Certificate Revocation and Suspension

- Certificate Revocation List (CRL)
  - Contains all revoked and suspended certificates
  - Issued by CA's

# HTTP Strict Transport Security

- A header field in an HTTP response
  - Tells a browser to load this page securely (via HTTPS)
  - If the certificate is not valid, the browser will block access to that website


Secure https://www.facebook.com
Email or Phone
Sign Up
It's free and always will be.
Sam
samccsf9@samsclass.info
samccsf9@samsclass.info
Birthday
Jan 1 1980
Female
Male
Create Account
Create a Page for a celebrity, band or
Elements Console Sources Network Timeline Profiles Application Security Audits
Preserve log Disable cache Offline No throttling
Filter Regex Hide data URLs XHR JS CSS Img Media Font Doc WS Manifest Other
Name
Headers Preview Response Cookies Timing
www.facebook.com
M7rKkwuGX0Kd.css
/rsrc.php/v3/yc/r
jBb1NN2qNIY.css
/rsrc.php/v3/yS/r
IZ86cv9aR90.css
/rsrc.php/v3/yu/r
qX6lddzoUq6.css
/rsrc.php/v3/yL/r
OYinIT15-RYC.css
/rsrc.php/v3/yU/r
piY-93il9Ut.css
/rsrc.php/v3/yd/r
Uz1_cNSYvZK.js
/rsrc.php/v3/yy/r
851565_60226995647418...
scontent.fsnc1-1.fna.fbcdn.n...
851565_216271631855613...
scontent.fsnc1-1.fna.fbcdn.n...
851558_160351450817973...
scontent.fsnc1-1.fna.fbcdn.n...
0_KqJAcnl8J.gif
/rsrc.php/v3/yA/r
data:font/opentype;...
data:font/opentype;...
gf6#txsw8zm.png
/rsrc.php/v3/y4/r
7k4FTgQBx2Y.png
General
Request URL: https://www.facebook.com/
Request Method: GET
Status Code: 200
Remote Address: [2a03:2880:f122:83:face:b00c:0:25de]:443
Response Headers
cache-control: private, no-cache, no-store, must-revalidate
content-encoding: br
content-security-policy: default-src * data: blob:;script-src *.facebook.com *.fbcdn.net *.facebook.net *.google-analytics.com *.virtualearth.net *.google.com 127.0.0.1:* *.spotilocal.com:* 'unsafe-inline' 'unsafe-eval' fbstatic-a.akamaihd.net fbcdn-static-b-a.akamaihd.net *.atlassolutions.com blob: data: 'self';style-src data: 'unsafe-inline' *;connect-src *.facebook.com *.fbcdn.net *.facebook.net *.spotilocal.com:* *.akamaihd.net wss://*.facebook.com:* https://fb.scanandcleanlocal.com:* *.atlassolutions.com attachment.fbsbx.com ws://localhost:* blob: *.cdninstagram.com 'self' chrome-extension://boadgeojelhgndaghljhdicfkmllpafd chrome-extension://dliochdbjfkdbacpmhlcpmleaejidimm;
content-type: text/html
date: Thu, 12 Jan 2017 06:12:54 GMT
expires: Sat, 01 Jan 2000 00:00:00 GMT
p3p: CP="Facebook does not have a P3P policy. Learn why here: http://fb.me/p3p"
pragma: no-cache
public-key-pins-report-only: max-age=500; pin-sha256="WoiWRyIOVNa9ihaBciRSC7XHjliYS9VwUGOIud4PB18="; pin-sha256="r/mIkG3eEpVdm+u/ko/cwxzOMo1bk4TyHIlByibiA5E="; pin-sha256="q4PO2G2cbkZhZ82+JgmRUyGMoAeozA+B5XVXQWB8XWQ="; report-uri="http://reports.fb.com/hpkp/"
set-cookie: reg_fb_ref=https%3A%2F%2Fwww.facebook.com%2F; path=/; domain=.facebook.com; httponly
set-cookie: fr=0HDm76CqX6SF5L1vv..BXbbuT.1m.AAA.0.0.BYdx5m.AWVHxEFv; expires=Wed, 12-Apr-2017 06:12:54 GMT; Max-Age=7776000; path=/; domain=.facebook.com; httponly
set-cookie: wd=deleted; expires=Thu, 01-Jan-1970 00:00:01 GMT; Max-Age=-1484201573; path=/; domain=.facebook.com
set-cookie: _js_reg_fb_ref=deleted; expires=Thu, 01-Jan-1970 00:00:01 GMT; Max-Age=-1484201573; path=/; domain=.facebook.com; httponly
status: 200
strict-transport-security: max-age=15552000; preload
vary: Accept-Encoding
x-content-type-options: nosniff
x-fb-debug: RSW/dFrEb69lYvdGgygmcSCB/v+aVIoagjaJY6c0B7zaBResXNTbu7C1EvUUUKz2+Dz/lh3A+qR93rmzQmFVGA==
x-frame-options: DENY
x-xss-protection: 0

# Backing Up Keys

- Backing up keys is critical
  - If keys are destroyed and not backed up properly, encrypted business-critical information might be irretrievable
- The CA is usually responsible for backing up keys
  - A key recovery policy is also part of the CA's responsibility

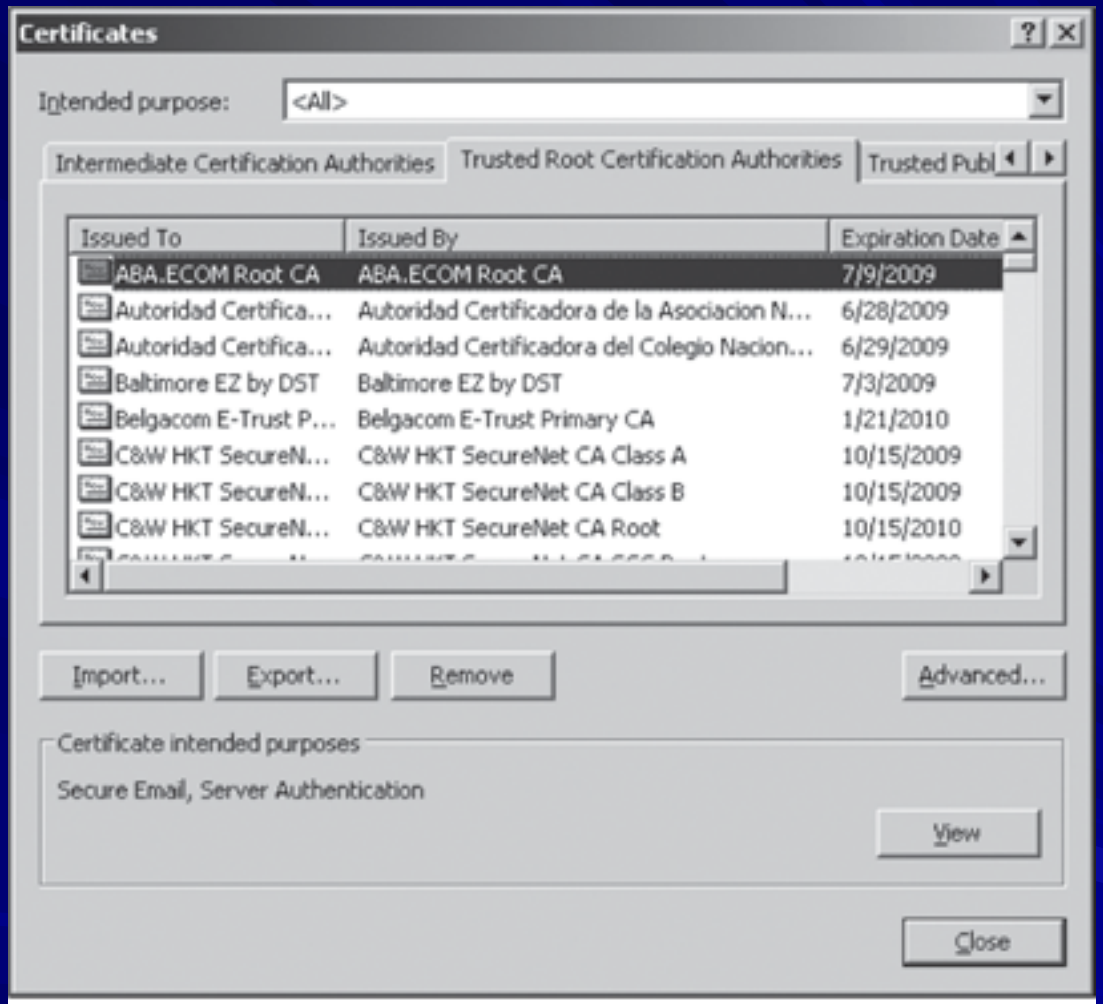
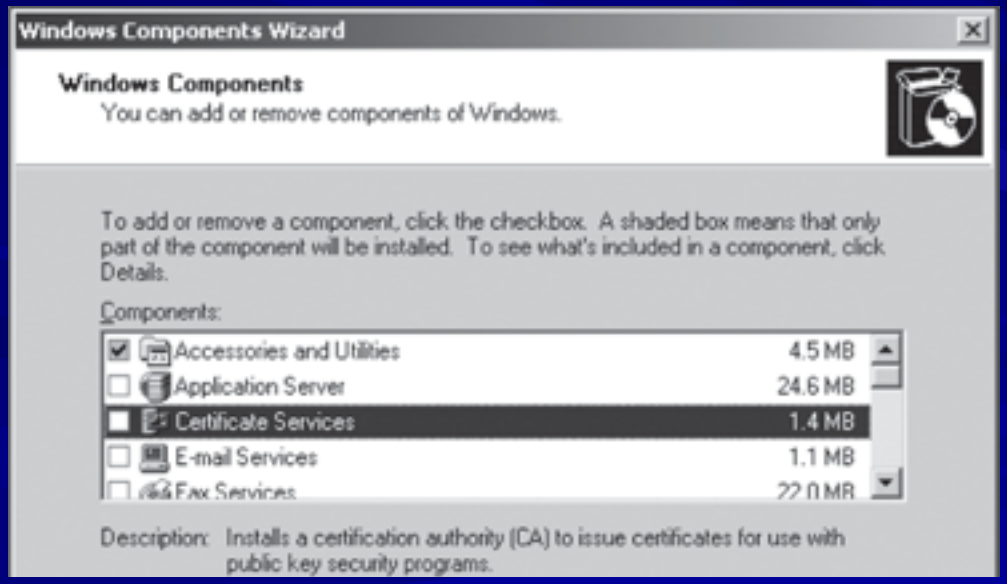



**Figure 12-7** Using Internet Explorer to view the Root CA

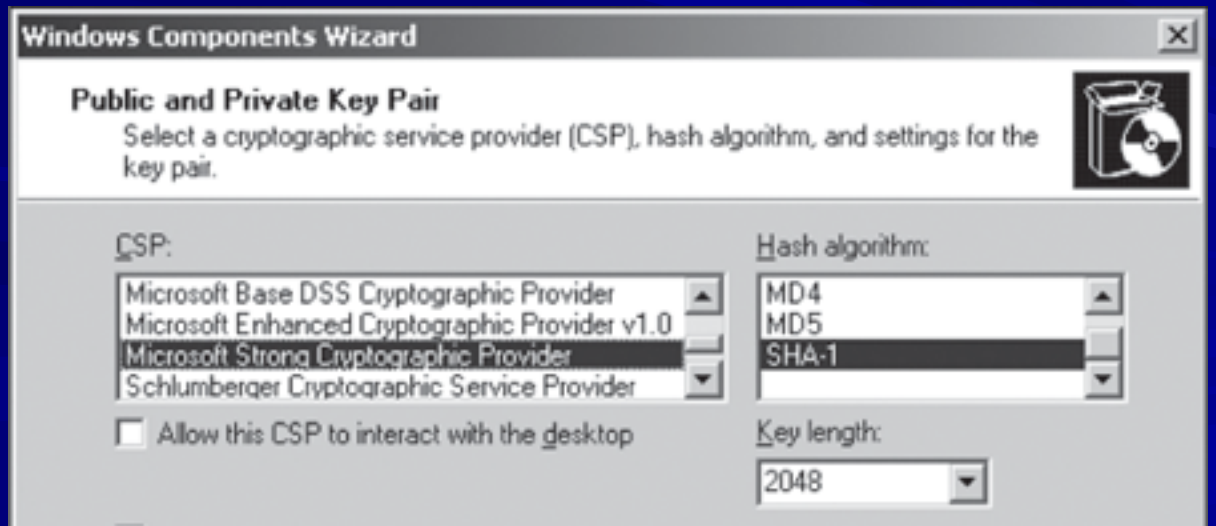
# Microsoft Root CA

- You can set up your own Certificate Authority Server
  - Windows Server
  - Install Certificate Services



# Microsoft Root CA

- Specify options to generate certificates, including
  - Cryptographic Service Provider
  - Hash algorithm
  - Key length




Kahoot!

# Cryptographic Attacks

# Understanding Cryptographic Attacks

- Sniffing and port scanning are passive attacks – just watching
- Active attacks attempt to determine the secret key being used to encrypt plaintext
- Cryptographic algorithms are usually public
  - Follows the open-source culture
  - Except the NSA and CIA and etc.

# Birthday Attack

- If 23 people are in the room, what is the chance that they all have different birthdays?

$$\frac{365}{365} \times \frac{364}{365} \times \frac{363}{365} \times \frac{363}{365} \times \frac{361}{365} \times \frac{360}{365} \times \cdots \frac{343}{365}$$

$$= 49\%$$

So there's a 51% chance that two of them have the same birthday

- See link Ch 12r

# Birthday Attack

- If there are N possible hash values,
  - You'll find collisions when you have calculated 1.2 x sqrt(N) values
- SHA-1 uses a 160-bit key
  - Theoretically, it would require $2^{80}$ computations to break
  - SHA-1 has no known collisions, but they are expected to be found soon

# Mathematical Attacks

- Properties of the algorithm are attacked by using mathematical computations
- Categories
  - Ciphertext-only attack
    - The attacker has the ciphertext of several messages but not the plaintext
    - Attacker tries to find out the key and algorithm used to encrypt the messages
    - Attacker can capture ciphertext using a sniffer program such as Ethereal or Tcpdump

# Mathematical Attacks

- Categories
  - Known plaintext attack
    - The attacker has messages in both encrypted form and decrypted forms
    - This attack is easier to perform than the ciphertext-only attack
    - Looks for patterns in both plaintext and ciphertext
  - Chosen-plaintext attack
    - The attacker has access to plaintext and ciphertext
    - Attacker has the ability to choose which message to encrypt

# Mathematical Attacks

- Categories (continued)
  - Chosen-ciphertext attack
    - The attacker has access to the ciphertext to be decrypted and to the resulting plaintext
    - Attacker needs access to the cryptosystem to perform this type of attack

# Brute Force Attack

- An attacker tries to guess passwords by attempting every possible combination of letters
  - Requires lots of time and patience
  - Password-cracking programs that can use brute force
    - John the Ripper
    - Cain and Abel
    - Ophcrack
      - Also uses memory to save time – “Rainbow tables”

# Man-in-the-Middle Attack


- Victim sends public key to Server
  - Attacker generates two “false” key pairs
  - Attacker intercepts the genuine keys and send false keys out
  - Both parties send encrypted traffic, but not with the same keys
- These false keys won’t be verified by a CA

# SSL/TLS Downgrade Attack

- Attacker in the middle can alter a request
  - So that it appears to come from an old system, such as Windows 95
  - Only capable of using old, broken ciphers, such as ones with 40-bit keys
  - Or "export-grade" encryption which was designed to be crackable by the US Gov't
- Fix: configure server to only use secure ciphers

# Dictionary Attack

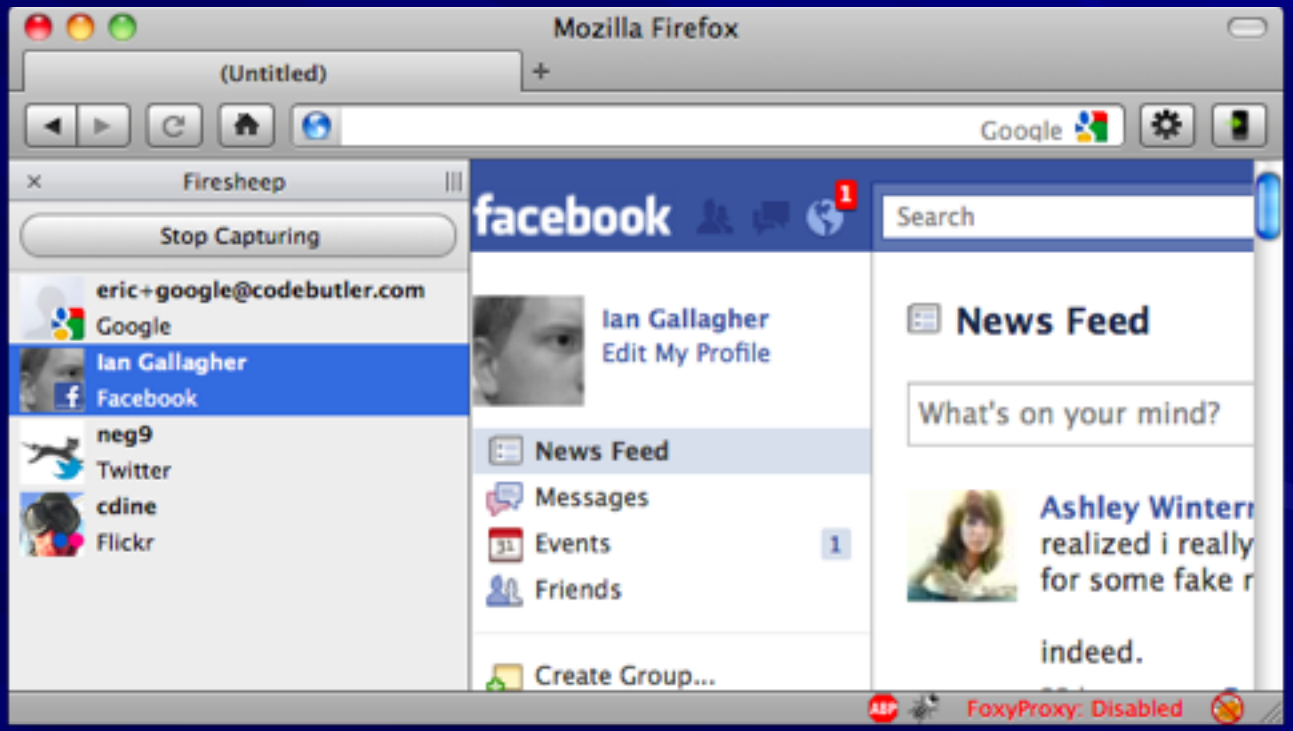
- Attacker uses a dictionary of known words to try to guess passwords
  - There are programs that can help attackers run a dictionary attack
- Programs that can do dictionary attacks
  - John the Ripper
  - Cain and Abel

# Replay Attack

- The attacker captures data and attempts to resubmit the captured data
  - The device thinks a legitimate connection is in effect
- If the captured data was logon information, the attacker could gain access to a system and be authenticated
- Most authentication systems are resistant to replay attacks

# Firesheep

- Replays cookies to access others' accounts on wireless networks



# Password Cracking

- Password cracking is illegal in the United States
  - It is legal to crack your own password if you forgot it
- You need the hashed password file
  - /etc/passwd or /etc/shadow for *NIX
  - The SAM database in Windows
- Then perform dictionary or brute-force attacks on the file

# Password cracking programs

- John the Ripper
- Hydra (THC)
- EXPECT
- L0phtcrack and Ophcrack
- Pwdump
- Ophcrack does it all for you – gathering the SAM database and cracking it

# Recent SSL Vulnerabilities

- Sslstrip MITM
  - Convert secure connection to insecure one
  - Works on mixed-mode authentication pages like Twitter (link Ch 12zj)
  - Written by Moxie Marlinspike

# Recent SSL Vulnerabilities

- Wildcard certificates
  - *%00.evil.com
  - Fools browser (link Ch 12zk)
- Renegotiation vulnerability
  - Can break any SSL/TLS session (Ch 12zl)
- Browsers often fail to check Certificate Revocation Lists
- Untrustworthy CA entries in browser


Kahoot!