CS 161: Computer Security

Lecture 7

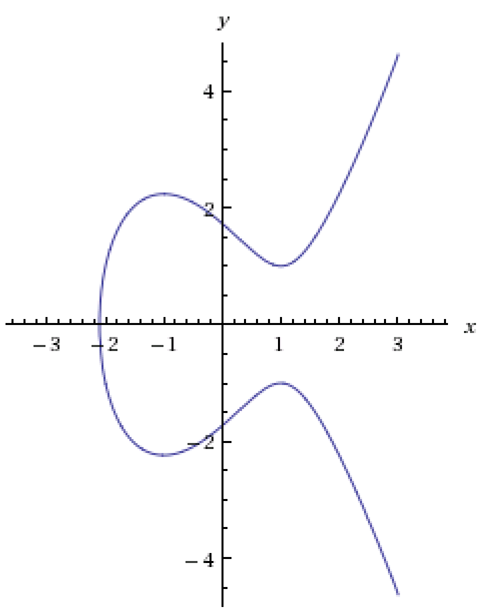
September 22, 2015

Where we are

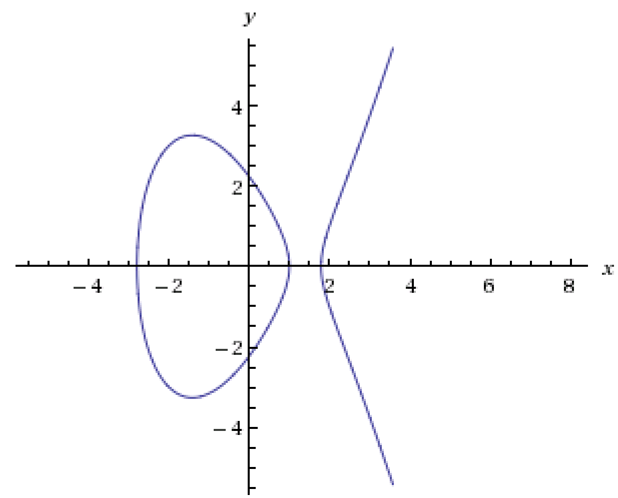
- How did NSA break SSL?
- Basic number theory
- RSA
- Digital certificates
- Shamir secret sharing
- Rabin signatures
- Secure hashing
- Elliptic curve cryptography
- Pseudo-random number generation
- SSL protocol

Review: Elliptic curves

- Weierstrass equations
- $y^2 = x^3 + Ax + B$

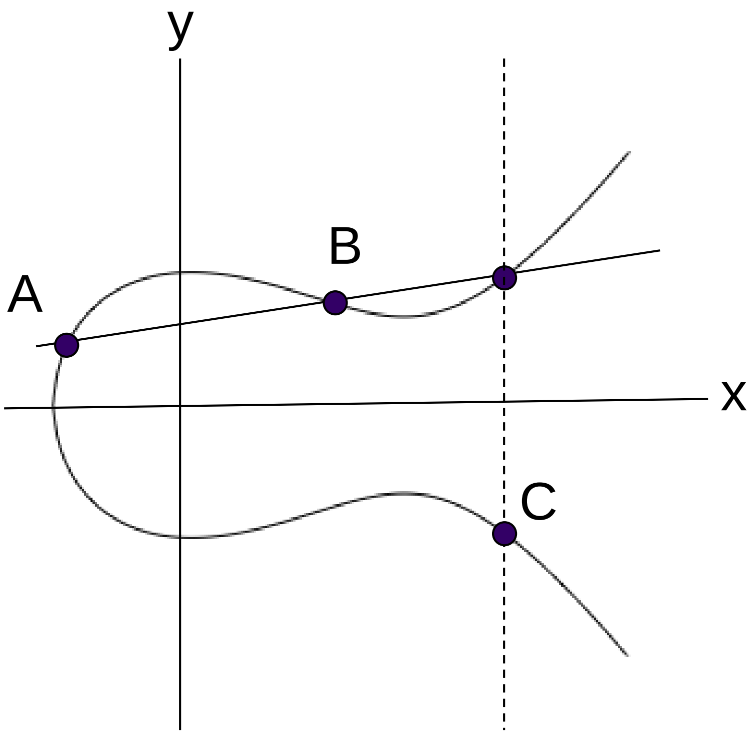


$$y^2 = x^3 - 3x + 3$$



$$y^2 = x^3 - 6x + 5$$

Review: Elliptic Curve operation: $\oplus$



$$C = A \oplus B$$

Review: Addition rules

- $P \oplus \mathcal{O} = P$
- $(x, y) \oplus (x, -y) = \mathcal{O}$
- $\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{if } P \neq Q \\ \dfrac{3x_1^2 + A}{2y_1} & \text{if } P = Q \end{cases}$
- $P \oplus Q = (x_3, y_3)$
- $x_3 = (\lambda^2 - x_1 - x_2)$ & $y_3 = \lambda(x_1 - x_3) - y_1$

Important: EC can include points: (0,0), (0,y), (x,0)

Homework 3.1

- E: $y^2 = x^3 + 4x + 3 \bmod 3$
- Points include $(0,0)$!
- Note $(0,0) \neq \mathcal{O}$

Review: Scalar multiplication

- $0P = \mathcal{O}$
- $1P = P$
- $2P = P \oplus P$
- $3P = P \oplus P \oplus P$
- $4P = P \oplus P \oplus P \oplus P$
- $\ldots$

Review: Discrete logarithm problem

- Fix a prime p and a generator $g \in \mathbb{Z}_p$
- Discrete logarithm problem:

Given $a \in \mathbb{Z}_p$, find k such that $g^k \equiv a \pmod{p}$

- Fix an elliptic curve E mod p and a point P
- Discrete logarithm problem:

Given $Q \in E$, find k such that $kP = Q$

Review: Best algorithms for discrete log

- Discrete log mod p

$$e^{((c+o(1))(\log p)^{1/3}(\log\log p)^{2/3})}$$

- Discrete log over elliptic curve mod p

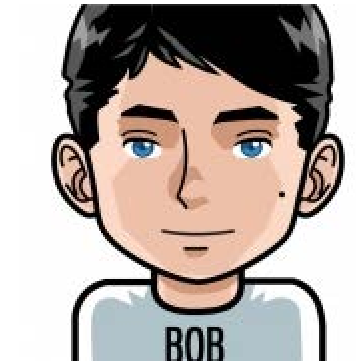
$$\sqrt{p}$$

- Elliptic curves make things <u>much</u> harder

Review: Diffie-Hellman key exchange

Alice

Bob



Elliptic Curve $E \bmod p, P \in E$

$\longrightarrow$

$n_A P$

$\longrightarrow$

$n_B P$

$\longleftarrow$

$n_A(n_B P)$

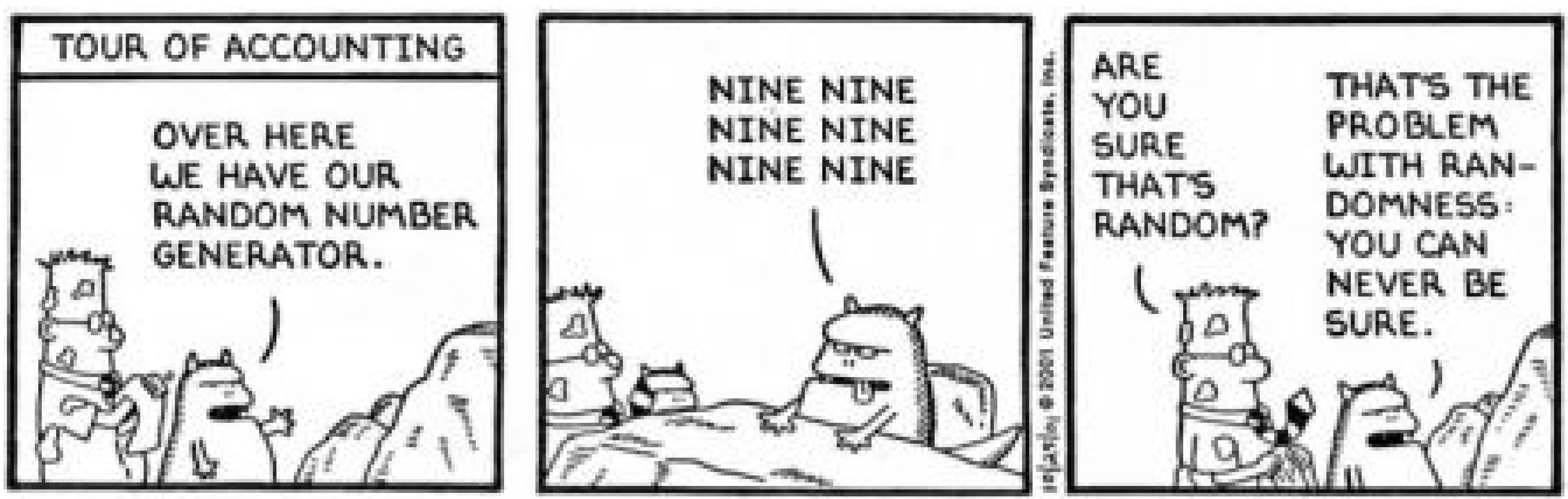
$n_B(n_A P)$

Review: Elgamal cryptosystem

- Referee
  - elliptic curve $E \bmod p, P \in E$
- Bob
  - Picks random x
  - $Q = xP$
  - public key (E, P, Q); secret key x
- Alice
  - message $M \in E$, random k
  - $A = kP$; $B = M \oplus kQ$
  - transmits $\langle A, B \rangle$
- Bob
  - $B \oplus (-x)A = M \oplus kQ \oplus (-x)kP = M \oplus xkP \oplus (-x)kP = M$

Pseudo-random number generation

- Random bits are quite valuable
- Famous book: *A Million Random Digits*
- Example: in Elgamal cryptosystem, security depends on Alice choosing a random k



Linear-congruential PRNG

- Recommended in Knuth

  p large prime

  $s_0 \leftarrow$random seed

  $s_{i+1} \leftarrow as_i + b \bmod p$

  $b_i \leftarrow (s_i \bmod 2)$

Output $b_1, b_2, \ldots$

Linear-congruential PRNG problems

- Linear-congruential PRNG passes most statistical tests of randomness
- Unfortunately, linear-congruential PRNG are not good enough for security purposes
- If we observe $b_1, b_2, \ldots$ we can infer constants PRNG equation

Another approach

- Other approaches use encryption functions

  $s_0 \leftarrow$ random seed

  $s_{i+1} \leftarrow \mathrm{Encrypt}(s_i)$

  $b_i \leftarrow (s_i \bmod 2)$

- This can be effective,
- but also can have several technical problems
  - computational cost
  - cycles

Cryptographically strong PRNG

- Given a sequence of pseudo-random bits, it is intractable to predict next bit with probability greater than $50\% + o\left(\frac{1}{n}\right)$
  - n is a parameter of cryptographic security, such as the length of a modulus

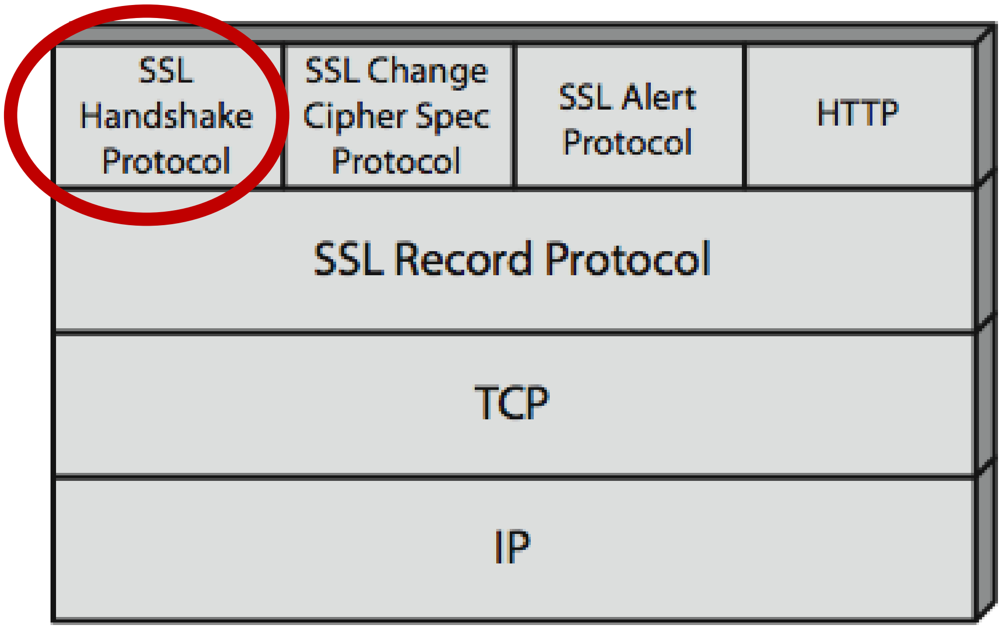
Strong focus on randomness

- Use “true randomness”
- Use specially designed PRNG for crypto
- NSA got involved in creating PRNG standard
- Based on elliptic curve cryptography
  - Dual Elliptic Curve Deterministic Random Bit Generator
  - Dual_EC_DRBG

SSL (Secure Socket Layer)

- Taher Elgamal “father” of SSL
- SSL originally developed by Netscape
- Subsequently became Internet standard known as TLS (Transport Layer Security)
- HTTPS: HTTP over SSL (or TLS)
  - Typically on port 443 (regular http on port 80)
- SSL has two layers of protocols

SSL Architecture



SSL Handshake protocol message types

Message Type	Parameters
`hello_request`	(null)
`client_hello`	version, random, session id, cipher suite, compression method
`server_hello`	version, random, session id, cipher suite, compression method
`certificate`	chain of X.509v3 certificates
`server_key_exchange`	parameters, signature
`certificate_request`	type, authorities
`server_done`	(null)
`certificate_verify`	signature
`client_key_exchange`	parameters, signature
`finished`	hash value

client_hello, server_hello

- Version:
  - highest SSL version understood by client
- Random:
  - 32-bit timestamp, 28 pseudo-random bits
- Session ID
  - Zero for new session, non-zero to update existing session
- CipherSuite
  - Key exchange method & cipher spec
- Compression method:
  - Compression methods supported

Key exchange methods

- RSA
- Diffie-Hellman
  - Fixed (DH parameters signed by CA)
  - Ephemeral (DH parameters signed w/public keys)
  - Anonymous (can be attacked with MITM)
- EC Diffie-Hellman
- Big issue: forward secrecy
- + others

CipherSpec

- CipherAlgorithm
  - No encryption, AES, DES, 3DES, IDEA, + others
- MACAlgorithm
  - MD5 (!), SHA1, SHA-2 (256, 384) + others
- Some other fields

client_key_exchange

- RSA
  - Client computes a 48-byte pre-master secret
  - Encrypts pre-master secret in server public key
  - Sends to server
- DH
  - Client & server compute a DH shared key
  - Shared key is pre-master secret

Master secret

```
master_secret = MD5(pre_master_secret +
                    SHA('A' + pre_master_secret +
                               ClientHello.random +
                               ServerHello.random)) +
                MD5(pre_master_secret +
                    SHA('BB' + pre_master_secret +
                               ClientHello.random +
                               ServerHello.random)) +
                MD5(pre_master_secret +
                    SHA('CCC' + pre_master_secret +
                               ClientHello.random +
                               ServerHello.random));
```

SSL problems

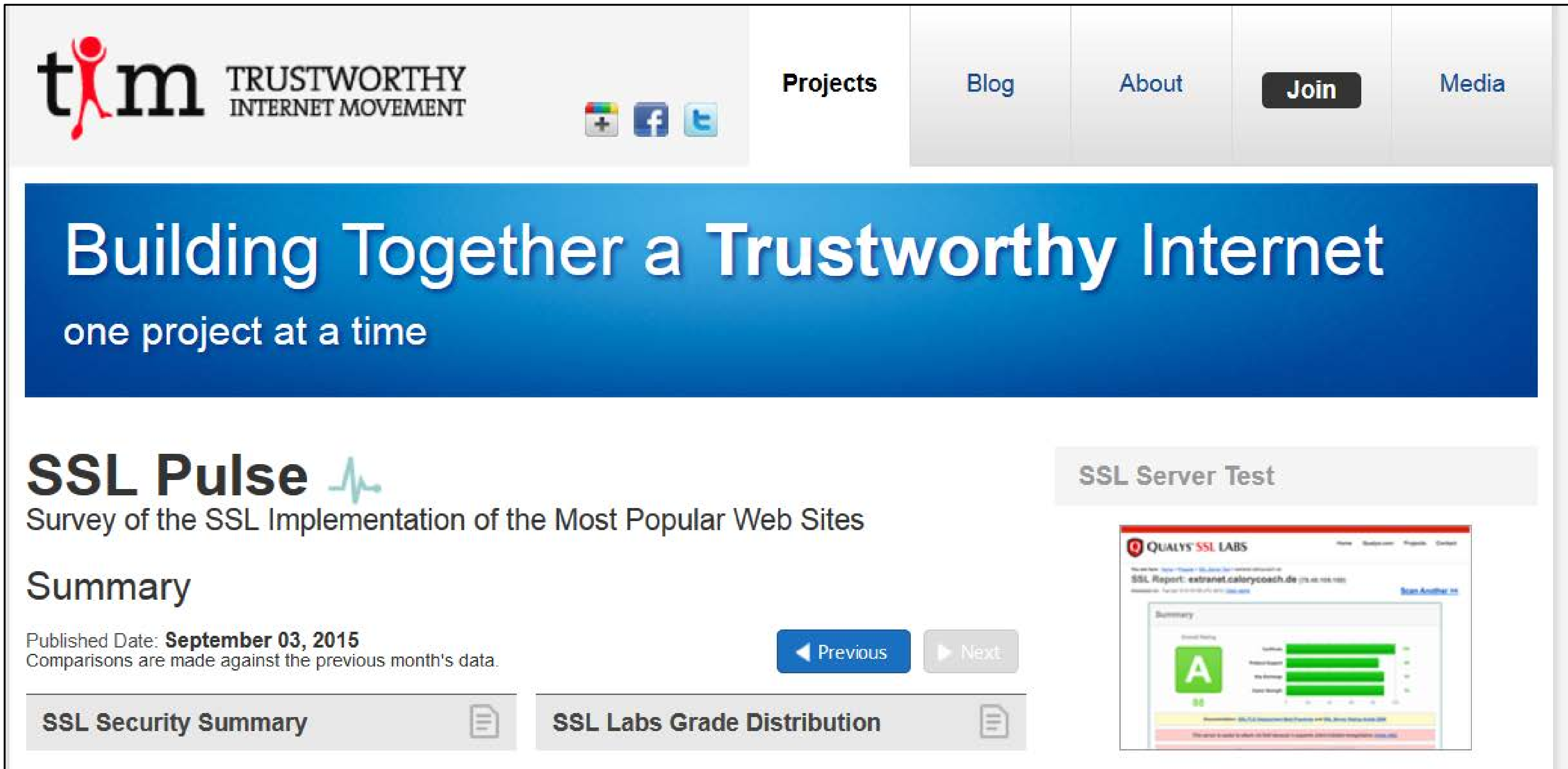
- SSL 2.0 broken
  - Message authentication uses MD5
  - Handshake messages not protected
    - Man in the middle attack forces weak cipher suite
  - Same key for message integrity/encryption
    - If one of those protocols weak, the other is no good
  - Man-in-the-middle can terminate
    - TCP FIN

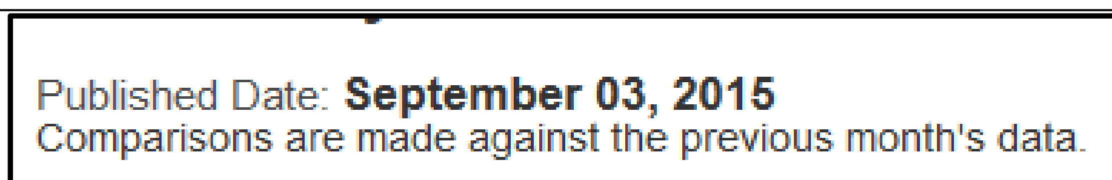
SSL problems

- SSL 3.0 broken
- TLS 1.0 broken
  - BEAST
  - Browser Exploit Against SSL/TLS Tool
  - Prof. David Wagner warned about this in 1999!
  - Man in the middle attack – uses weakness in CBC (will discuss next week)

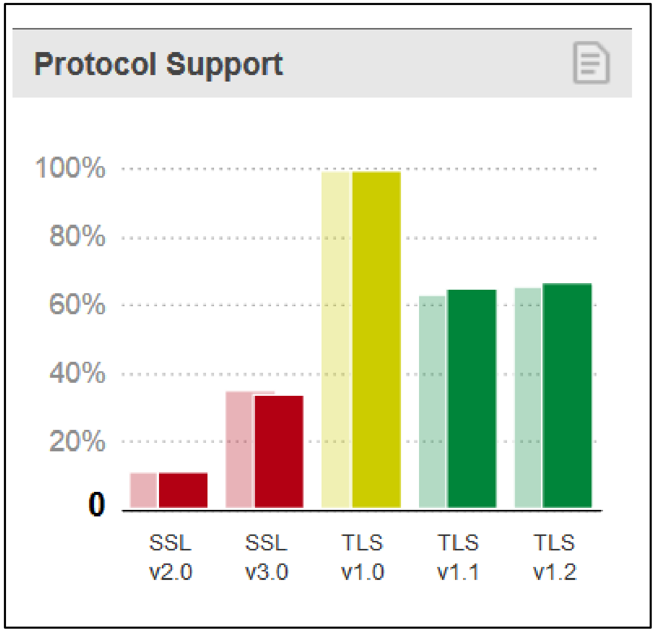
SSL weaknesses in wild

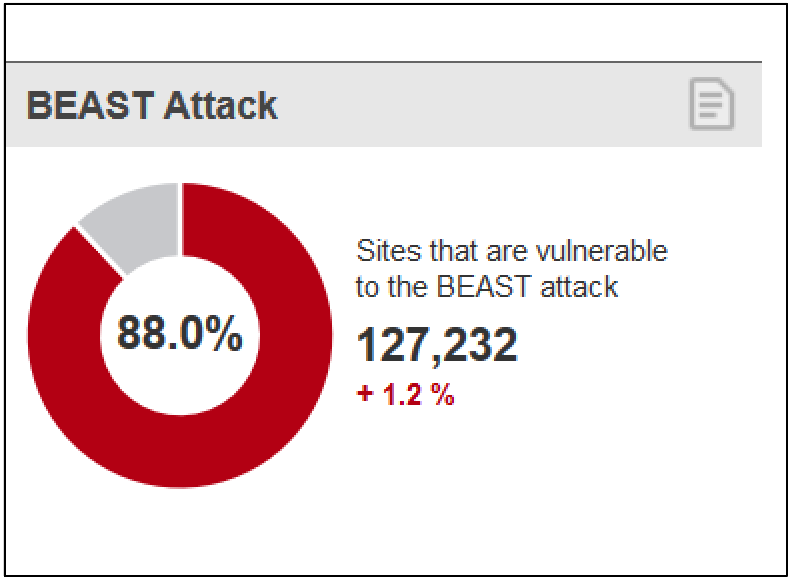
- https://www.trustworthyinternet.org/ssl-pulse/

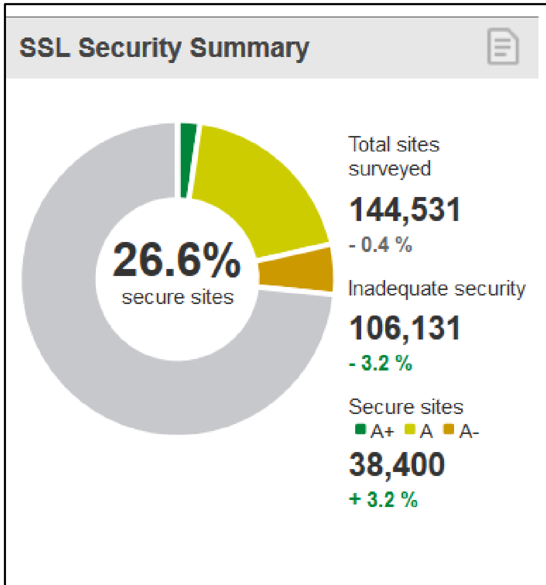


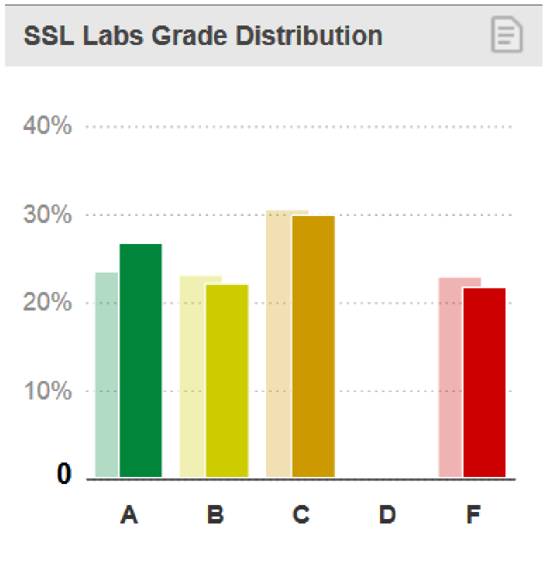


SSL weaknesses in wild

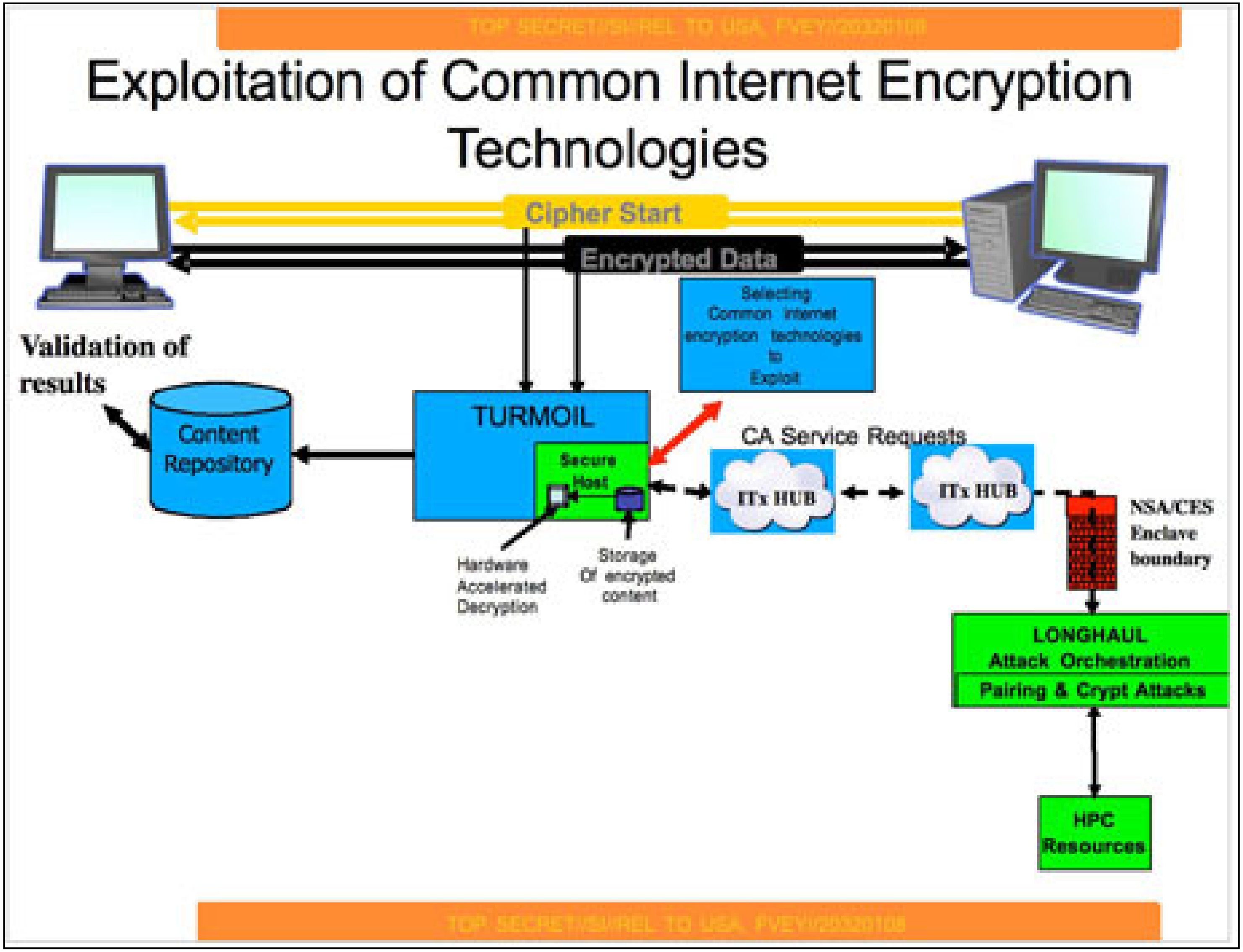








NSA Bullrun



TOP SECRET//SI//REL TO USA, FVEY//20320108
Exploitation of Common Internet Encryption Technologies
Cipher Start
Encrypted Data
Selecting Common internet encryption technologies to Exploit
Validation of results
Content Repository
TURMOIL
Secure Host
CA Service Requests
ITx HUB
ITx HUB
NSA/CES Enclave boundary
Hardware Accelerated Decryption
Storage Of encrypted content
LONGHAUL
Attack Orchestration
Pairing & Crypt Attacks
HPC Resources
TOP SECRET//SI//REL TO USA, FVEY//20320108

NSA has long history of backdoors

- NSA put backdoors in Cisco equipment
    - All routers built for export reportedly include NSA backdoor
- NSA has put backdoors in Crypto AG since 1957!

- Dual Elliptic Curve Deterministic Random Bit Generator
- Dual_EC_DRBG

Dual_EC_DRBG

- Algorithm specifies a particular (cyclic) curve
- Algorithm specifies a particular P, Q point pair
  - How were the points generated?
- Standardized in NIST SP800-90A

$$r_i \leftarrow number(s_i P)$$

$$s_{i+1} \leftarrow number(r_i P)$$

$$Output(bitstring(r_i Q))$$

Dual_EC_DRBG

$r_i \leftarrow number(s_i P)$

$s_{i+1} \leftarrow number(r_i P)$

$Output(bitstring(r_i Q))$

- But $P = eQ$ for some e
- Finding that e would require solving EC Discrete Log Problem
- But for generated P, Q NSA could know e

Dual_EC_DRBG

$r_i \leftarrow number(s_i P)$

$s_{i+1} \leftarrow number(r_i P)$

$Output(bitstring(r_i Q))$

$P = eQ$

- Suppose NSA can observe just <u>one</u> output from PRNG $Output(bitstring(r_i Q))$
- Find $r_i Q$
- Compute $er_i Q = r_i(eQ) = r_i P = s_{i+1}$
- Now you know state of PRNG

Dual_EC_DRBG

- Attack is easy (and impossible to prove)
- We know some e exists
  - Question is – does NSA know it?
  - Finding it would be hard – but generating it would be easy
  - Because finding it is hard – impossible to prove!

What the NSA did

REUTERS EDITION: U.S.

HOME BUSINESS MARKETS WORLD POLITICS TECH OPINION BREAKINGVIEWS

Exclusive: Secret contract tied NSA and security industry pioneer

BY JOSEPH MENN

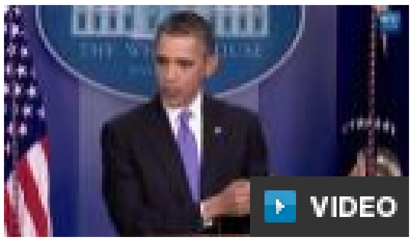
SAN FRANCISCO | Fri Dec 20, 2013 5:07pm EST

88 COMMENTS | Tweet Share this Email Print

A National Security Agency (NSA) data gathering facility is seen in Bluffdale, about 25 miles (40 km) south of Salt Lake

What the NSA did

RELATED VIDEO



Obama on surveillance: "There may be another way of skinning the cat"

RELATED TOPICS

Politics »

(Reuters) - As a key part of a campaign to embed encryption software that it could crack into widely used computer products, the U.S. National Security Agency arranged a secret $10 million contract with RSA, one of the most influential firms in the computer security industry, Reuters has learned.

Documents leaked by former NSA contractor Edward Snowden show that the NSA created and promulgated a flawed formula for generating random numbers to create a "back door" in encryption products, the New York Times reported in September. Reuters later reported that RSA became the most important distributor of that formula by rolling it into a software tool called Bsafe that is used to enhance security in personal computers and many other products.

What the NSA did

Undisclosed until now was that RSA received $10 million in a deal that set the NSA formula as the preferred, or default, method for number generation in the BSafe software, according to two sources familiar with the contract. Although that sum might seem paltry, it represented more than a third of the revenue that the relevant division at RSA had taken in during the entire previous year, securities filings show.

Next lecture

- Symmetric cryptography