SSL y TLS
— una larga lista de —
Eventos Desafortunados

About me...

- Es el tercer evento OWASP del que participo.
- Hace mas de 16 años que me dedico a la seguridad de forma exclusiva y un par adicionales a la tecnología
- Socio y Director en Porto, Trentalance, Antúnez y Asociados

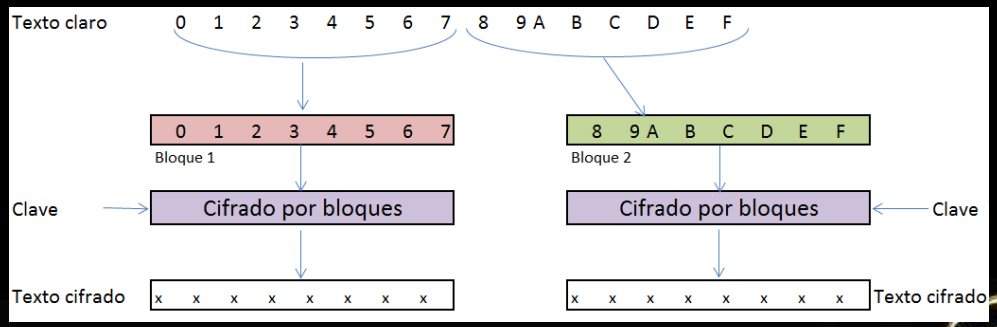
Intro

Conceptos

- Cifrado de flujo (Stream cipher)
  - Normalmente se trabaja con la representación a nivel de bits del texto claro, cifrando con una clave de la misma longitud que el texto claro (derivada de la clave brindada por el usuario)

	A								B								C							
Texto claro	0	1	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	0	0	0	1	1
Clave	0	0	1	1	1	1	1	0	0	0	1	0	1	0	1	0	0	1	0	0	1	1	1	0
Texto cifrado	0	1	1	1	1	1	1	1	0	1	1	0	1	0	0	0	0	0	0	0	1	1	0	1

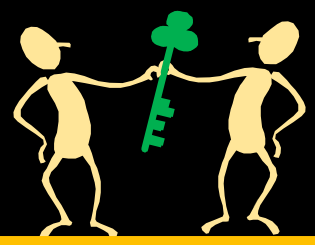
- Cifrado por bloques (Block cipher)
  - Se toman bloques de longitud fija de texto claro, se cifra con la clave (derivada de la clave brindada por el usuario), obteniendo un bloque de texto cifrado de la misma longitud.



Conceptos

Criptografía SIMETRICA

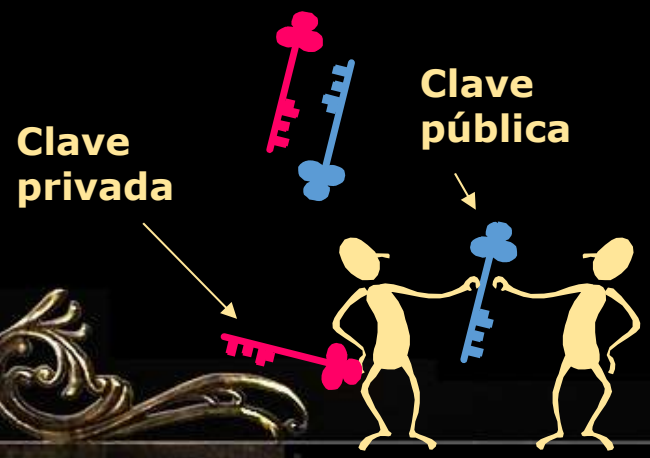
Se utiliza la misma clave tanto para cifrar como para descifrar. Requiere que ambos participantes de la comunicación conozcan esta clave. La clave debe informarse a través de un canal seguro (out-of-band).



Criptografía ASIMETRICA

Se utilizan pares de claves. Lo que es cifrado con una clave de un par, solo la complementaria puede descifrarlo.

Una de las claves se hace pública y puede distribuirse libremente. La otra es privada y solo el propietario tiene acceso a ella. No requiere que ambos participantes de la comunicación conozcan una clave secreta en común.



PKI (Public Key Infra)

- Certificados
  - Mecanismo usado para asociar una clave pública con un conjunto de campos para identificar unívocamente a su propietario.
  - Los campos usados en el certificado están definidos en el estándar X.509 V3 (2008)
  - La entidad que autentica (firma) los certificados digitales se denomina Autoridad Certificante (CA).
  - Los equipos clientes "confían" en autoridades certificantes reconocidas por defecto (certificados de CA preinstalados)

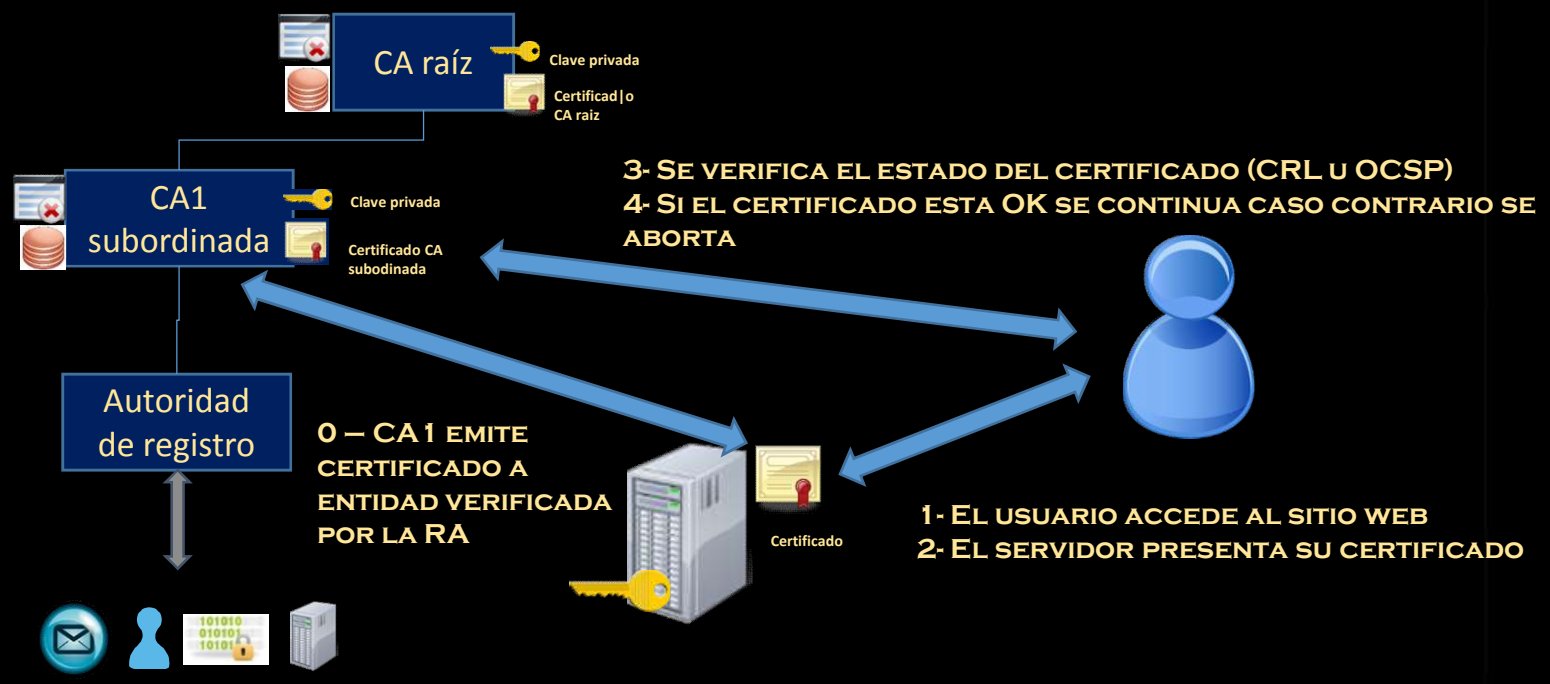
Version:3
Nro de serie: AB F1 0B DA
Algoritmo de firma: SHA1-RSA
Emisor: ZARAZA CA
Validez: Desde 01-11-2014 Hasta 30-10-2020

Este Certificado pertenece a a Javier Antunez
Organización: blackmantisecurity.com.ar
Unidad organizacional.: Educación
Provincia: Buenos Aires Pais: AR
Clave pública de Javier Antunez
Tipo de clave publica: RSA 2048 bits

Identificación de emisor (opcional)
Identificación de sujeto (opcional)
Extensiones (opcional)

Algoritmo de firma del certificado:
md5WithRSAEncryption
Firma Digital de la autoridad certificante
Punto de distribucion de CRL

PKI (Public Key Infra)



SSL/TLS

- Criptografía asimétrica y simétrica.
- Originado en Netscape en la década del 90
- Cifrado del canal de datos, autenticación del server, integridad de mensajes y opcionalmente autenticación del cliente
- Se basa en PKI (certificados digitales), para autenticar las partes intervinientes en una comunicación.

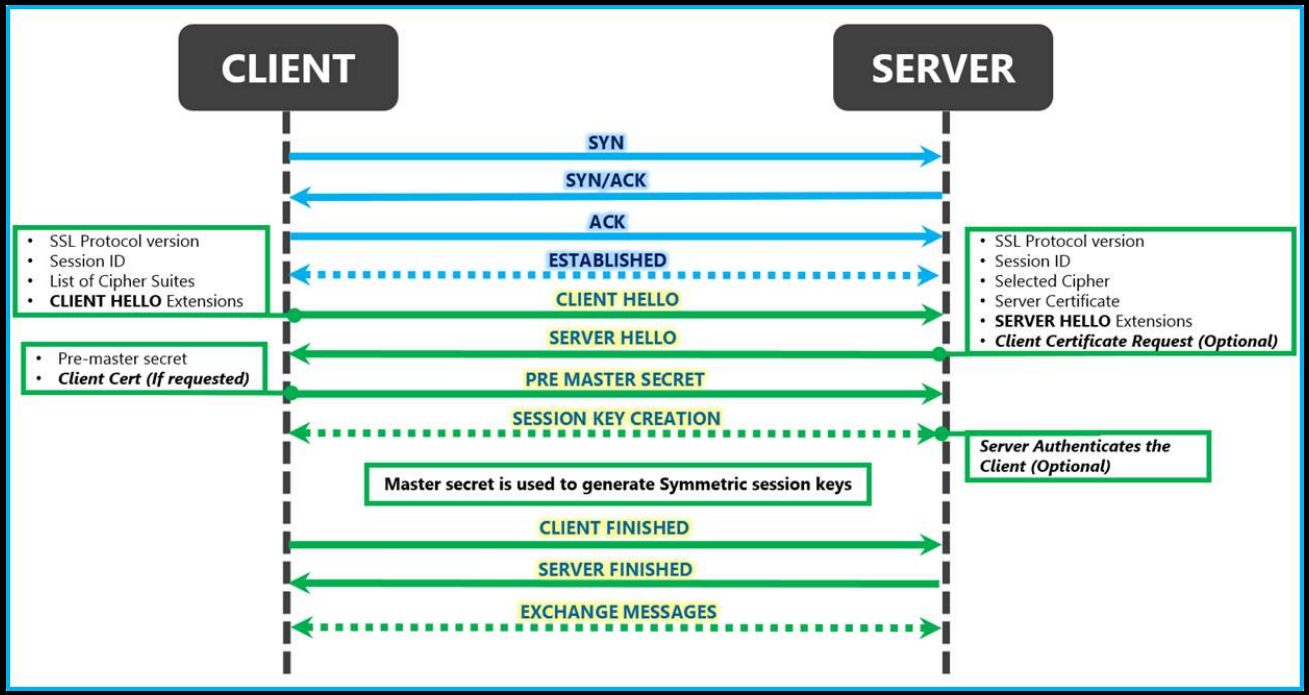
SSL/TLS

- La conexión (ej: HTTPS) implica una serie de pasos iniciales:
  - Handshake TCP
  - Negociación de algoritmos a utilizar entre las partes (ciphersuite)
  - Autenticación, generación e intercambio de claves
  - Cifrado del canal

Ciphersuite

- Es la definición de estándares utilizados para:
  - Algoritmo de intercambio de claves
    - RSA, Diffie-Hellman, ECDH, SRP, PSK
  - Algoritmo de cifrado
    - RC4, 3DES, AES, IDEA, DES o CAMELLIA
- Algoritmo de autenticación de mensajes
    - HMAC MD5, SHA-1, SHA-XXX – en TLS
    - MD5, MD4, MD2, SHA-1 – en SSL
- Función de generación de nros pseudo-aleatorios

Handshake SSL/TLS



SSL y TLS

— Una larga lista de—

Eventos Desafortunados

WHEN DID
YOU
LAST
EAT?

Un inicio accidentado

- Evento desafortunado SSL 1.0
  - No vio la luz, ya que tenia varias limitaciones y debilidades (por ejemplo no proveía integridad de datos, era vulnerable a ataques por replay).
- Evento desafortunado SSL 2.0
  - Se libera en 1995
  - Resulta ser vulnerable (downgrade de cipher suites y terminación de conexiones). Misma clave de cifrado e integridad (si la clave es débil ambas cosas lo serán).
- Primer versión "segura"(?) SSL 3.0 1996

Evento desafortunado Abuso de confianza

- 2001: Emitió certificados a alguien que se hizo pasar por Microsoft
- 2011: Un partner fue comprometido y se emitieron certificados sin validación
- 2011: Fue victima de una intrusión y se generaron decenas de certificados
- 2012: Se emitieron certificados de CA subordinadas por error, sin extensión CRL ni OCSP y un wildcard para google.com
- 2015: Se emitieron certificados para live.fi de forma fraudulenta
- 2015: Se emitieron certificados Extended Validation para www.google.com y google.com
- 2015: Se emitieron certificados con nombres prohibidos o Ips Internas

Detección

- http://www.certificate-transparency.org/
- Framework abierto para monitoreo y auditoria de certificados.
- Cercano al tiempo real
- Permite detectar certificados SSL emitidos erroneamente por una CA o adquiridos maliciosamente
- Hace possible identificar CAs que estan emitiendo certificadios maliciosamente
- El caso de google fue detectado por esta via

Evento desafortunado Abuso de confianza

- Superfish - lenovo (2015)
  - Aplicación pre instalada
  - Instalaba un certificado autofirmado con su clave privada
  - Interceptaba trafico HTTPS
- eDellRoot/DSDTestProvider - Dell (2015)
  - Certificados pre instalados como trusted root
  - Incluian la clave privada de los mismos :-)
- Ambos permitían realizar ataques MITM

Evento Desafortunado
Uso de algoritmos débiles o rotos

- 2004 MD5 roto
- 2004 SHA-1 casi roto
- 2008 en 25th CCC en Berlin. Usando colisiones de MD5 se podían generar certificados arbitrarios
- El ataque flame utilizó esta técnica para firmar actualizaciones falsas de Microsoft

Evento Desafortunado
Uso de algoritmos débiles o rotos

- 2011 se recomendaba utilizar RC4 como mitigación a BEAST
- 2013 RC4 algoritmo roto completamente sobre TLS
- 2012 Se identifica falta de aleatoriedad en generación de nros pseudo aleatorios (en RSA 1024 bit).

Evento Desafortunado
Fallas de diseño

OCSP Soft Fail

Moxie Marlinspike (2009)

- Falla de diseño especificada en RFC
- Respuestas OCSP sin firmar (Rep. fallas)
- Falsifica la respuesta código 3 (try again)

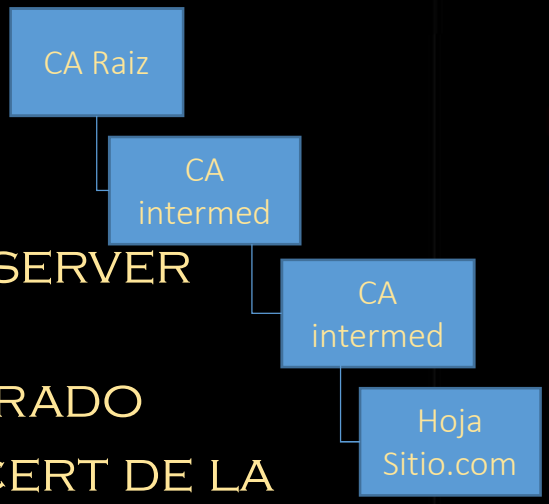
Adam Langley (2011)

- Identifica un ataque similar, pero a nivel de red
- Bloqueo de conexiones OCSP o devolución de errores HTTP al cliente

Evento Desafortunado Fallas de implementación

Moxie Marlinspike (2009)

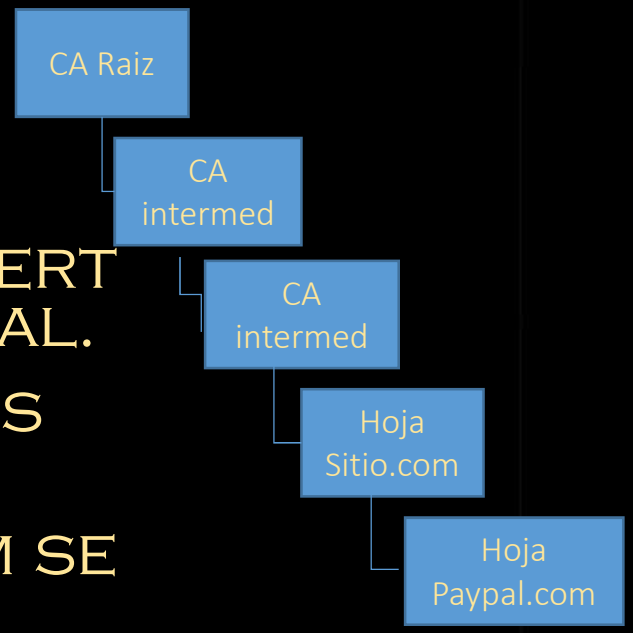
- Validación de certificados:
  - Verificar que el CN del cert y el server coincidan.
  - Verificar que el cert no haya expirado
  - Verificar la firma del cert. Si el cert de la Ca que firmo esta en nuestro trust de Cas.
  - Si el certificado de la CA esta en nuestra lista finalizar
  - En caso contrario moverse un paso hacia arriba en la cadena y repetir la secuencia



Evento Desafortunado Fallas de implementación

Moxie Marlinspike (2009)

- Que pasaría si…
- Con la clave privada de un cert valido firmo otro como paypal.
- La cadena de certificación es correcta
- El certificado de paypal.com se consideraba válido
- El problema es que no se revisaba el campo CA=False



Evento Desafortunado
Fallas de implementación

Moxie Marlinspike (2009)

- Demostró que podían solicitarse certificados con CN de este estilo:
  - www.Paypal.com\0.midominio.com
  - La Autoridad de registro identfica midominio.com, me valida y emite el certificado.
  - Del lado del navegador, la interpretación del CN es www.paypal.com (el \0 se interpreta como el final del string)

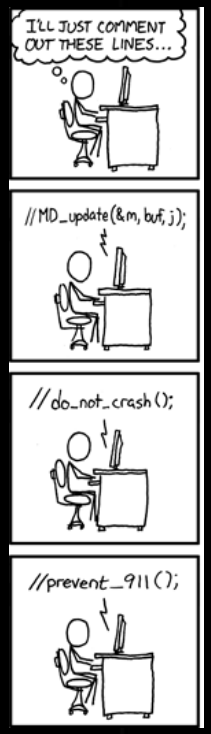
Evento Desafortunado
Fallas de implementación

Moxie Marlinspike (2009)

- Un lindo caso podría ser (wilcards)
  - *\0.midominio.com
  - *~.midominio.com
  - www.paypal.com|mail.google.com|www.etrade.com|www.bankofamerica.com|www.wachovia.com|www.pnc.com|www.wellsfargo.com)\0.midominio.com
- O enviar un exploit en el CN
  - AAAAAAAAAAAAAAAAAAAAAAAA\0overwrite).midominio.com

Evento desafortunado
Bugs de software

- OpenSSL sobre OpenBSD (2006)
  - Se comentaron ciertas líneas de código para evitar warnings de compilación.
  - La generación de nros aleatorios al momento de generar claves se volvía predecible.
- HeartBleed (2014)
  - Bug en OpenSSL HearTbeat
  - Permitia obtener segmentos de memoria del server en el que podian residir, contraseñas, datos sensibles u otras claves.



Evento Desafortunado
Bugs de software

- Apple GoToFail(2014)

```
static OSStatus
SSLVerifySignedServerKeyExchange(SSLContext *ctx, bool isRsa, SSLBuffer signedParams,
                                 uint8_t *signature, UInt16 signatureLen)
{
    OSStatus        err;
    SSLBuffer       hashOut, hashCtx, clientRandom, serverRandom;
    uint8_t         hashes[SSL_SHA1_DIGEST_LEN + SSL_MD5_DIGEST_LEN];
    SSLBuffer       signedHashes;
    uint8_t         *dataToSign;
    size_t          dataToSignLen;

    ...
    if ((err = ReadyHash(&SSLHashSHA1, &hashCtx)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
        goto fail;
        goto fail;
    if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
        goto fail;

    err = sslRawVerify(ctx,
                       ctx->peerPubKey,
                       dataToSign,              /* plaintext */
                       dataToSignLen,           /* plaintext length */
                       signature,
                       signatureLen);
    if(err) {
        sslErrorLog("SSLDecodeSignedServerKeyExchange: sslRawVerify "
                    "returned %d\n", (int)err);
        goto fail;
    }
```

"Understanding the Apple goto fail vulnerability", Amit Sethi, Cigital, Feb 25. 2014
http://www.cigital.com/justice-league-blog/2014/02/25/understanding-apple-goto-fail-

Evento Desafortunado
Bugs de software

- OpenSSL (Julio 2015)
  - Problema de validación de certificados
  - Permitía usar certificados confiables como si fueran de una CA valida (simil moxie en 2009)
- OpenSSL (Diciembre 2015)
  - 4 vulns, una de ella para explotar contra RSA, DSA y Diffie–Hellman. (caro explotar)
- OpenSSL (Enero 2016)
  - Key Recovery Attack on DH small subgroups (CVE-2016-0701)

Evento Desafortunado
Bugs de software

- OpenSSL (Marzo 2016)
  - Cross-protocol attack on TLS using SSLv2 (DROWN) (CVE-2016-0800)
  - Divide-and-conquer session key recovery in SSLv2 (CVE-2016-0703)
    - Facilita determinar las claves de sesión (en 16 conexiones)
    - Bug ayuda a ataque DROWN permitiendo ataques MITM en línea
  - Bleichenbacher oracle in SSLv2 (CVE-2016-0704)
    - Permite realizar ataque DROWN aun mas eficientemente

Evento Desafortunado BEAST (2011)

BEAST Browser Exploit Against SSL/TLS

- Afecta SSL 3.0 y TLS 1.0
- Es un ataque client side
- Se basa en un ataque "chosen plain text" "cookie="
- Solo afecta los ciphers basados en bloques en modo CBC (AES, DES)
- Se puede "adivinar" porciones pequeñas de datos
- Se mitiga utilizando TLS 1.1 o 1.2

Evento Desafortunado CRIME (2012)

CRIME Compresion Ratio Info-leak Made Easy

- Afecta todas las versiones de SSL/TLS
- Solo puede atacarse si se está utilizando compresión de TLS
- Se basa en un ataque "chosen plain text" donde se conoce parte del texto claro (ej: cookie:secret=)
- Se puede mitigar desactivando la compresión de SSL/TLS

Evento Desafortunado BREACH (2013)

BREACH: Browser Reconnaissance and Exfiltration via Adaptative Compression Hypertext

- Afecta a todas las versiones de SSL/TLS
- Es independiente del ciphersuite
- Mas simple si el cipher es stream
- Aprovecha la compresión HTTP (<> CRIME compresión TLS)

Evento Desafortunado Lucky13 (2013)

- Afecta a todas las versiones de TLS
- Es un ataque similar a BEAST
- Se pasan paquetes con bloques de relleno (padding) escogidos
- Los bloques que tienen el relleno (padding) correcto son respondidos mas rápido.
- Se realiza un muestreo estadístico para "adivinar" el contenido plano. Es muy sensible a tiempos (requiere poca latencia)
- Solo afecta los ciphers basados en bloques en modo CBC (AES, DES)

Evento Desafortunado Poodle (2014)

Padding Oracle On Downgraded Legacy Encryption

- Afecta solo a SSL 3.0
- Requiere ataque MitM para ejercer un ataque de downgrade de protocolos de autenticacion.
- En promedio se requieren 256 requests para descifrar un byte cifrado
- Funciona si el modo de cifrado es CBC
- Evento desafortunado SSL 3.0 R.I.P

Evento Desafortunado TLS padding (2015)

Padding Oracle On Downgraded Legacy Encryption (Reloaded)

- Variante de Poodle
- Afecta ciertas implementaciones de TLS (vendors en particular)
- Bajo ciertos supuestos se puede descubrir un caracter en 256 intentos
- No requiere el MITM para hacer downgrade a SSL 3.0

Evento Desafortunado Freak (2015)

FREAK
SSL/TLS Vulnerability

Factoring Attack on RSA-EXPORT Keys

- Se basa en forzar que los clientes hagan un downgrade para aceptar cifrado débil a traves de un ataque MiTM
- Se fuerza el uso de "export grade" cryptography (NSA)
- Por ejemplo se podría hacer que el cliente acepte utilizar RSA con claves de 512 bits
- Luego se factorizan las claves (7 horas en Amazon EC2)

Evento Desafortunado LogJAM (2015)

LOGJAM

- Similar a FREAK
- Requiere entorno de ataque MiTM.
- Permite forzar que se haga un downgrade para aceptar cifrado débil
- Se realiza un downgrade con claves Diffie Hellman de 512 bits y luego se factorizan
- Afecta a todos los navegadores y a servidores que acepten DIFFIE HELLMAN

Evento Desafortunado Bar Mitzvah (2015)

- Aprovecha debilidades de RC4
- Permite obtener segmentos de texto plano a partir de texto cifrado
- Permite obtener parte de las claves de sesión
- Si las claves utilizadas son débiles, permite recuperar todo el texto claro.

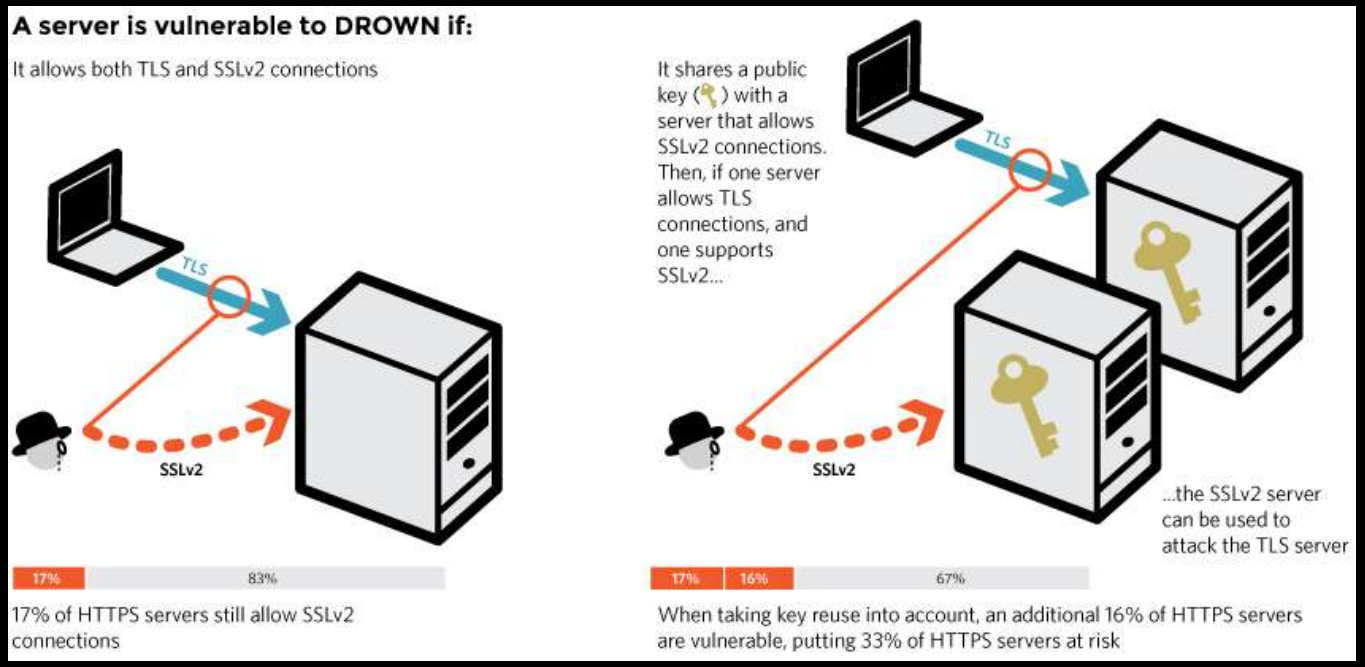
Evento Desafortunado DROWN (2016)

Decrypting RSA with Obsolete and Weakened eNcryption

- Afecta a Servidores:
  - con SSL V2
  - que comparten clave privada con servidores con SSL V2
- A la fecha de publicación 33% de los web servers de Internet.
- Permite descifrar comunicaciones y ataques MITM

Evento Desafortunado DROWN (2016)

Decrypting RSA with Obsolete and Weakened eNcryption



Evento Desafortunado rupture (2016)

Rupture framework

- BH Asia 2016 – Muestran formas de optimizar el ataque Breach
- Utilizan técnicas de división y paralelizacion de las tareas.
- Liberan framework "RUPTURE"
- La idea es generalizarlo para varios de los ataques de esta presentación

Pensamientos finales

¿Preguntas?

Muchas gracias

Para la billetera del Caballero o cartera de la dama

https://wiki.mozilla.org/Security/Server_Side_TLS

https://www.ssllabs.com/ssltest