



#XIVJORNADASCCNCERT






TALLER

**Protección y ataque en la autenticación criptográfica
de información y password hashing**

Dr. Alfonso Muñoz - @mindcrypt

*Head of Cybersecurity Unit & Cybersecurity Research – Inetum Digital Risk
(Co) editor – Red Temática Criptored*



-  alfonso@criptored.com
-  t.me/criptored
-  [http://github.com/mindcrypt](https://github.com/mindcrypt)
-  <https://es.linkedin.com/in/alfonsomuñoz>
-  @mindcrypt

- 18 años de experiencia
- Doctor Ingeniero Telecomunicación (UPM)
- Red Temática Criptored
- Expert security member – Europol (EC3)
- Libros, patentes, certificaciones de seguridad, ponente en conferencias de prestigio, tools, premios académicos e industriales, profesor de máster de seguridad (+4), papers (+70), security bulletins (Microsoft, Foxit, Hall of fame – Google, ...), etc.

Perfil:

- Offensive security (sw/hw)
- Cryptography & Covert channels/Stego
- Cutting-edge research (defensive & offensive)

Índice

- 1. Principios de la criptografía**
- 2. Un poco de teoría - ¿Cómo autenticar información/usuarios?**
 - a. Cifrado autenticado/padding**
 - b. Protección y almacenamiento de claves criptográficas**
- 3. Práctica - Atacando a funciones hash. Password hashing**
- 4. Conclusiones**

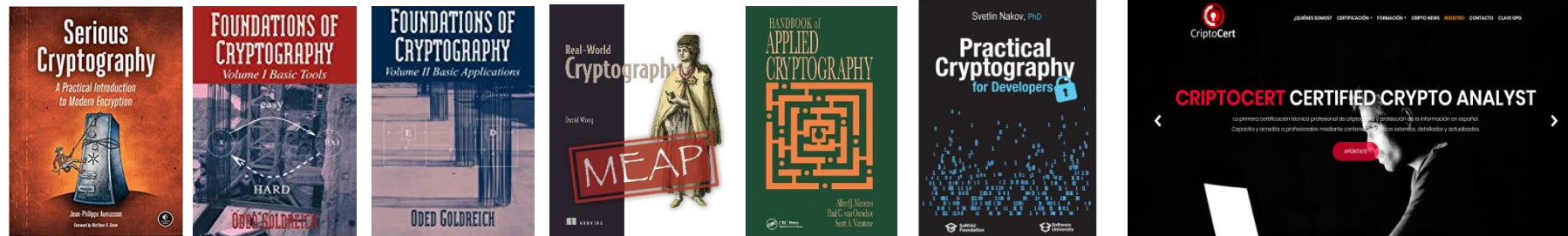
Criptología: Criptografía + Criptoanálisis

Criptología. Del gr. κρυπτός kryptós 'oculto' y -logía.
Estudio de los sistemas, claves y lenguajes ocultos o secretos.

- **Conceptos básicos - Tipo de criptografía**

- Principios de Kerckhoffs (1883) y Teoría de la información de Claude Shannon (1948-49)
- Bloques de bits o flujo de bits + una o más claves criptográficas
- Clave criptográfica = Información que permite realizar una acción específica a su poseedor
- **Criptografía simétrica** = Emisor y receptor comparten la misma clave privada
- **Criptografía asimétrica** = Cada participante tiene dos claves, una pública (cifrado) y una privada (descifrado/autenticación)
- **Tendencias criptográficas** – Criptografía cuántica, Criptografía postcuántica, Criptografía homomórfica, lightweight cryptography, whitebox cryptography, searchable encryption, criptografía diferencial, format-preserving encryption (FPE)...

- **Cryptographic Best Practices** - <https://gist.github.com/atoponce/07d8d4c833873be2f68c34f9afc5a78a>



https://www.cryptocert.com/CriptoCert_Analyst.html

Criptografía aplicada vs Criptografía teórica

Errores frecuentes

- Crear tu propio algoritmo o implementar uno existente
- Mal uso de librerías o algoritmos
 - **Cryptographic Best Practices** - <https://gist.github.com/atoponce/07d8d4c833873be2f68c34f9afc5a78a>
 - <https://security.googleblog.com/2018/08/introducing-tink-cryptographic-software.html>
 - <https://github.com/google/wycheproof>

- Incorrecta protección de claves criptográficas
- Reutilizar código (stackoverflow)
- La criptografía se “evita” (canales laterales)
 - <https://i.blackhat.com/us-18/Wed-August-8/us-18-Guri-AirGap.pdf>
- Criptografía no aislada. Ej. SSL/TLS

Seguridad del protocolo SSL/TLS: Ataques criptoanalíticos modernos - <https://github.com/mindcrypt/libros>



Seguridad del protocolo SSL/TLS.
Ataques criptoanalíticos modernos

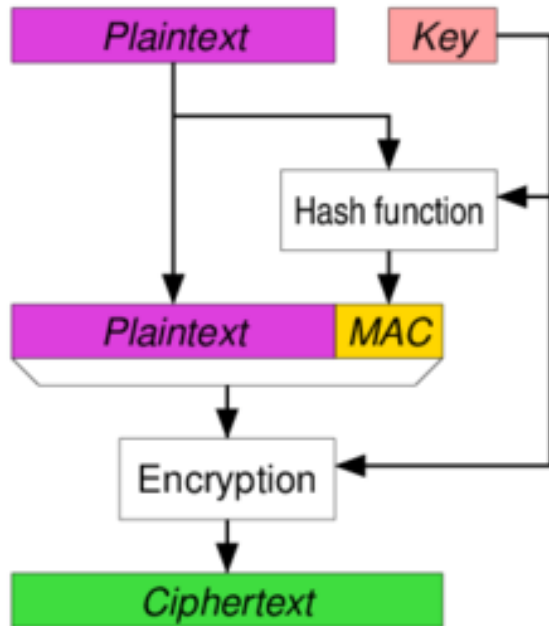
Autor: Dr. Alfonso Muñoz (@mindcrypt)
Prólogo: D. Juliano Rizzo



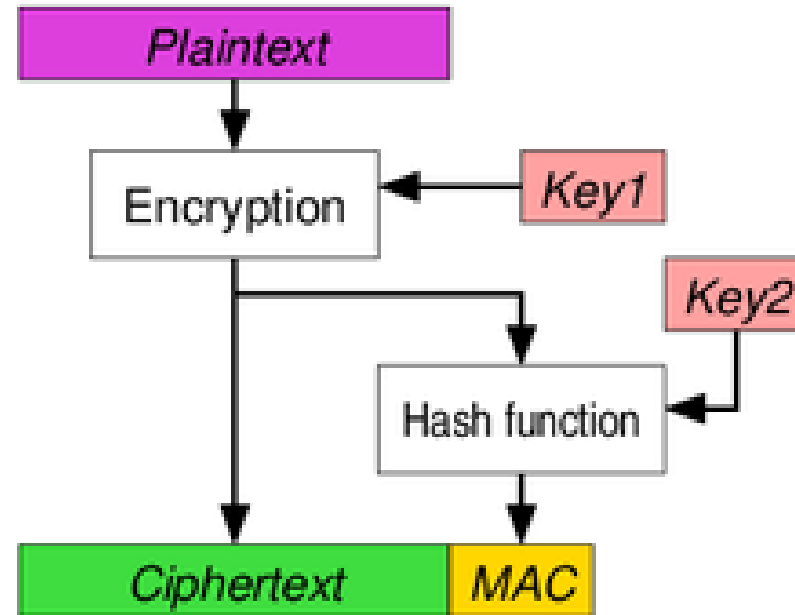
Reconocimiento-NoComercial-SinObraDerivada
CC BY-NC-ND
Madrid, 27/09/2020

¿Cómo autenticar información de manera segura?

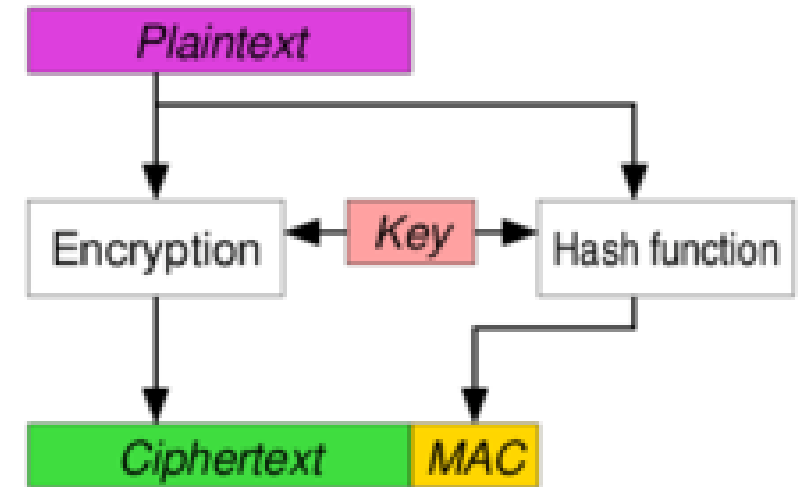
Cifrado autenticado - https://en.wikipedia.org/wiki/Authenticated_encryption



OPCION A. MAC-then-Encrypt (MtE)



OPCION B - Encrypt-then-MAC (EtM)



OPCION C. Encrypt-and-MAC (E&M)

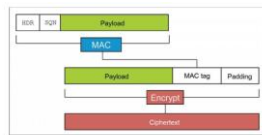


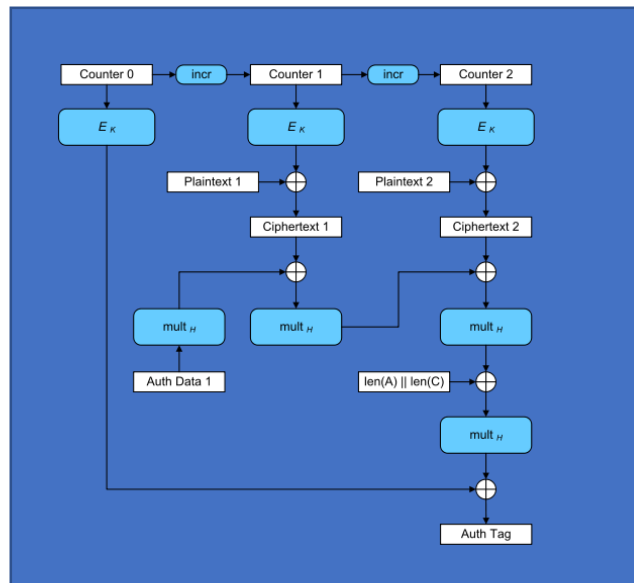
Figura 5. Proceso de cifrado DTLS

Existen diferentes maneras de crear funciones MAC. Ejemplos de creación de MACs son a partir de funciones hash (**HMAC**), a partir de cifradores en bloque (**CBC-MAC**) o a partir de Galois Counter Message Authentication Code (**GMAC**).

¿Cómo autenticar información de manera segura?

Cifrado autenticado - https://en.wikipedia.org/wiki/Authenticated_encryption

- **Authenticated Encryption (AE)** y **Authenticated Encryption with Associated Data (AEAD)** son una forma de cifrado que proporciona simultáneamente confidencialidad, integridad y autenticidad de los datos. Algunos de los modos de cifrado autenticado más famosos son: OCB 2.0, Key Wrap, CCM, EAX, EtM, GCM, ChaCha20-Poly1305 ...
- AEAD constituye el método de cifrado simétrico de referencia y recomendado en base a las buenas prácticas criptográficas actuales.



- AES-GCM - https://en.wikipedia.org/wiki/Galois/Counter_Mode
- Actúa como **un modo contador**, similar conceptualmente a AES-CTR, pero **garantizando autenticación de la información**.
- La información cifrada se utiliza para generar una **etiqueta de autenticación del mensaje**

¿Cómo autenticar información de manera segura?

Recordatorio: cuidado con el padding (padding oracle attacks)...

In [cryptography](#), **padding** refers to a number of distinct practices which all include adding data to the beginning, middle, or end of a message prior to encryption. In classical cryptography, padding may include adding nonsense phrases to a message to obscure the fact that many messages end in predictable ways, e.g. *sincerely yours*.

ANSI X9.23 ... | DD DD DD DD DD DD DD DD | DD DD DD DD **00 00 00 04** |

ISO 10126 ... | DD DD DD DD DD DD DD DD | DD DD DD DD **81 A6 23 04** |

PKCS#5 and PKCS#7 ... | DD DD DD DD DD DD | DD DD DD DD **04 04 04 04** |

ISO/IEC 7816-4 ... | DD DD DD DD DD DD DD DD | DD DD DD DD **80 00 00 00** |

Zero padding ... | DD DD DD DD DD DD DD DD | DD DD DD DD **00 00 00 00** |

Public key cryptography

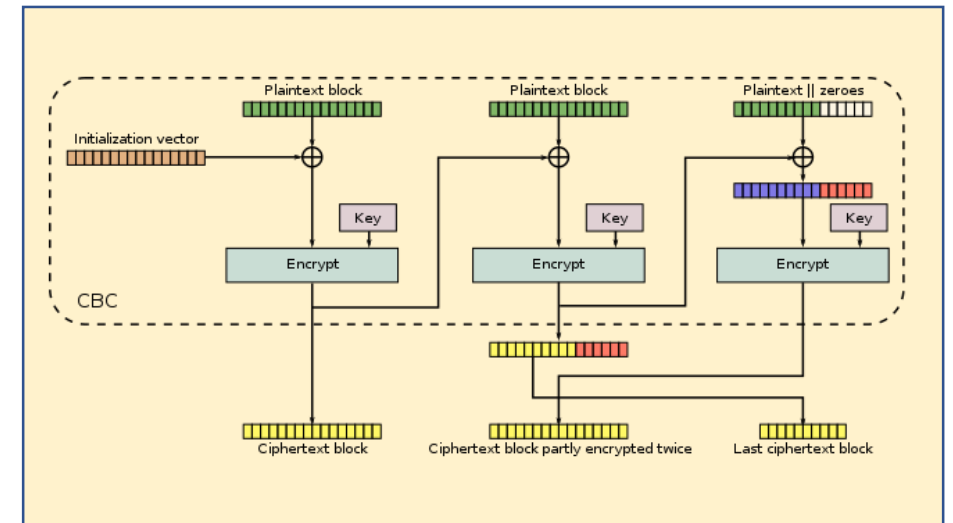
In [public key cryptography](#), padding is the process of preparing a message for encryption or signing using a specification or scheme such as [PKCS#1 v1.5](#), [OAEP](#), [PSS](#), [PSSR](#), [IEEE P1363](#)

[EMSA2](#) and [EMSA5](#). A modern form of padding for asymmetric primitives is [OAEP](#) applied to the [RSA algorithm](#), when it is used to encrypt a limited number of bytes.

Ataques SSL/TLS: Lucky13, New Bleichenbacher side Channels and attacks, Poodle, Drown, Robot, Zombie Poodle, GoldenPoodle...

Seguridad del protocolo SSL/TLS: Ataques criptoanalíticos modernos - <https://github.com/mindcrypt/libros>

[https://en.wikipedia.org/wiki/Padding_\(cryptography\)](https://en.wikipedia.org/wiki/Padding_(cryptography))
https://en.wikipedia.org/wiki/Disk_encryption_theory#XTS



https://en.wikipedia.org/wiki/Ciphertext_stealing
 Ej/ "RSA" $M=\{0...255\} \mid C = M^e \bmod n$

Protección y almacenamiento de “claves criptográficas”

Claves criptográficas

- Mecanismos de protección en función de si las claves se almacenan o se utilizan en tiempo de ejecución (alojadas en memoria RAM).
- Almacenamiento de claves robusta:
 - Mejoras algorítmicas: Key stretching, multiple encryption...
<https://blog.cryptographyengineering.com/2012/02/02/multiple-encryption/>
 - Almacenamiento software (gestor de contraseñas): KeePass, LastPass, 1Password.
 - Almacenamiento en hardware: HSM, YubiKey, etc.
- Protección de contraseñas en tiempo de uso: Whitebox cryptography

Almacenamiento “seguro” de contraseñas

Password hashing – KDF (Key Derivation Function)

- El uso de KDF es común para la verificación de contraseñas que se almacenarán mediante la aplicación de un hash criptográfico.
- El “key stretching” propone usar un salt para almacenar contraseñas de manera segura, frente a los ataques de fuerza bruta o diccionario, incrementando los recursos del atacante (tiempo y almacenamiento) necesario para probar cada posible clave. Esto se hará tradicionalmente con “iteraciones” de diversos algoritmos criptográficos.
- La mejor manera de almacenar o derivar una contraseña:

Derivation Key: $DK = KDF(\text{Key}, \text{Salt}, \text{Iterations})^*$

Key – La clave principal de la que se desea generar una versión más robusta frente a ataques de adivinación (fuerza bruta, diccionario o rainbowtables).

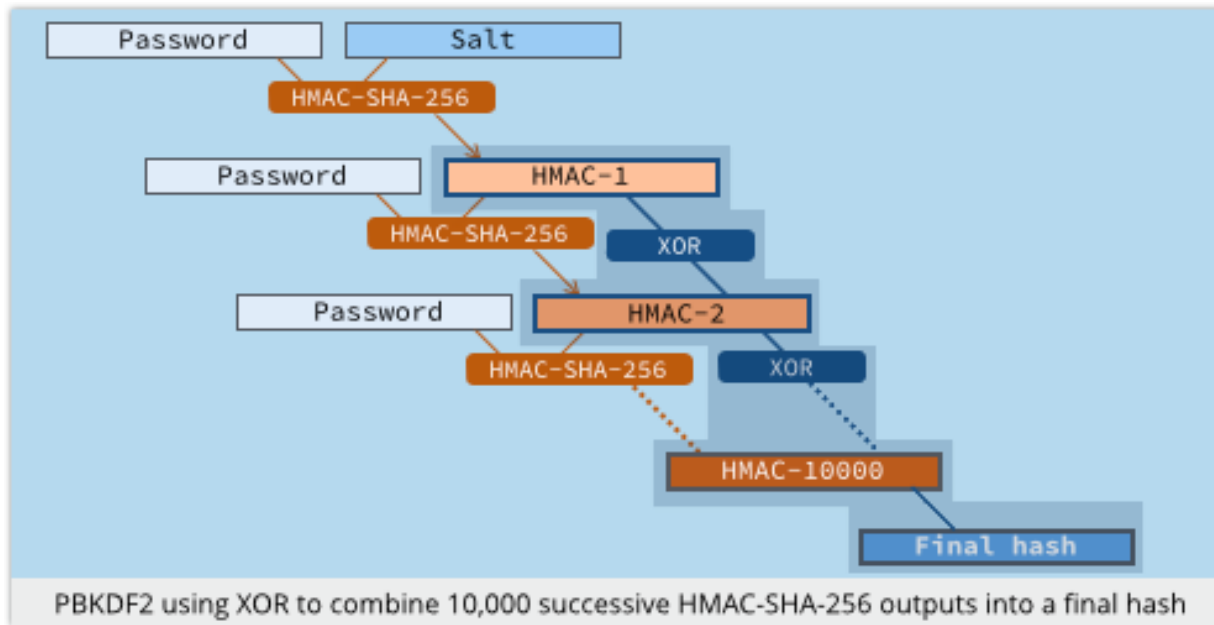
Salt – Número aleatorio, o semilla, que dificulta los ataques por diccionario (o rainbowtables) en la adivinación de la clave derivada en función de la clave principal. El atacante necesitará utilizar el salt específico para cada clave derivada para poder realizar un ataque sobre la clave.

Iterations – Este valor será mayor o igual a 1 y su principal utilidad es introducir retardos en el atacante en la fase de adivinación (key stretching).

Key stretching – PBKDF2

Password-Based Key Derivation Function 2

- PBKDF2 es una función pseudoaleatoria de derivación de claves de hasta 160 bits que usa como entrada una contraseña o frase de paso unido a un valor aleatorio (salt). El proceso se repetirá varias veces para producir la clave derivada. El NIST [1] recomienda valores en un rango de 1000 a 10.000.000



- Algoritmo muy utilizado: redes wifi (wpa y wpa2), software de cifrado (Filevault, Luks, Truecrypt, Veracrypt, ...), gestor de arranque (grub2, ...), sistemas operativos móviles (Android), etc.-
- Implementaciones de PBKDF2.
- https://en.wikipedia.org/wiki/List_of_PBKDF2_implementations

[1] Recommendation for Password-Based Key Derivation Part 1: Storage Applications. NIST.
<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-132.pdf>

Key stretching – scrypt / argon2

The scrypt Password-Based Key Derivation Function – <https://tools.ietf.org/html/rfc7914>

- Propuestas como PBKDF2, bcrypt o crypt basadas en key-stretching por uso de iteraciones son viables de atacar mediante hardware específico (ASICs o FPGA).
- El diseño de scrypt fuerza al uso de gran cantidad de memoria comparada con sus antecesores. Esto dificulta, por coste y tamaño, implementaciones hardware de ataque y su paralelización. Se ha usado como *proof-of-work* en diversas criptodivisas.

<https://password-hashing.net/argon2-specs.pdf> (ganadora del Password Hashing Competition – 2015)

- Argon2 es liberado en 3 versiones que gestionan **parámetros de memoria requerida, tiempo de ejecución (salt, iteraciones, ...) y grado de paralelismo**:
 - Argon2d: maximiza la resistencia a ataques de cracking utilizando GPUs.
 - Argon2i: optimiza la resistencia a ataques de canal lateral.
 - Argon2id: versión híbrida. El borrador actual de RFC recomienda este modo salvo alguna excepción que haga preferir alguno de los anteriores.

CyberChef

+

https://gchq.github.io/CyberChef/

Download CyberChef

Last build: 6 months ago - v9 supports multiple inputs and a Node API allowing you to program with CyberChef!

Options

About / Support

Operations

Search...

Favourites

To Base64

From Base64

To Hex

From Hex

To Hexdump

From Hexdump

URL Decode

Regular expression

Entropy

Fork

Magic

Data format

Encryption / Encoding

Public Key

Recipe

Input

length: 0

lines: 1

Output

STEP

BAKE!

Auto Bake

CN-CERT
centro criptológico nacional

← → ↻

attack.mitre.org/techniques/T1003/

🔍 ☆ 🌙 🏠 A ⋮

MITRE

ATT&CK™

Matrices

Tactics ▾

Techniques ▾

Mitigations ▾

Groups

Software

Resources ▾

Blog ↗

Contribute

Search site

ENTERPRISE ▾

TECHNIQUES

All

Initial Access +

Execution +

Persistence +

Privilege Escalation +

Defense Evasion +

Credential Access +

Account Manipulation

Bash History

Brute Force

Cloud Instance Metadata API

Credential Dumping

Credentials from Web Browsers

Credentials in Files

Home > Techniques > Enterprise > Credential Dumping

Credential Dumping

Credential dumping is the process of obtaining account login and password information, normally in the form of a hash or a clear text password, from the operating system and software. Credentials can then be used to perform Lateral Movement and access restricted information.

Several of the tools mentioned in this technique may be used by both adversaries and professional security testers. Additional custom tools likely exist as well.

Windows

SAM (Security Accounts Manager)

The SAM is a database file that contains local accounts for the host, typically those found with the 'net user' command. To enumerate the SAM database, system level access is required. A number of tools can be used to retrieve the SAM file through in-memory techniques:

- [pwdumpx.exe](#)
- [gsecdump](#)
- [Mimikatz](#)
- [secretsdump.py](#)

Alternatively, the SAM can be extracted from the Registry with [Reg](#):

ID: T1003

Tactic: Credential Access

Platform: Windows, Linux, macOS

Permissions Required: Administrator, SYSTEM, root

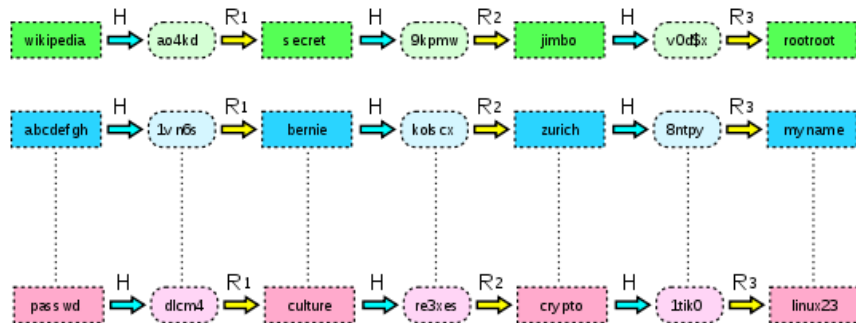
Data Sources: API monitoring, Process monitoring, PowerShell logs, Process command-line parameters

Contributors: Vincent Le Toux; Ed Williams, Trustwave, SpiderLabs

Version: 1.1

Tipos de ataques – Diccionarios, fuerza bruta, ataques híbridos

- Los ataques basados en diccionario, aunque no garantizan el éxito, son soluciones prácticas. La generación de diccionarios, tablas precalculadas, se complica si el defensor añade mecanismos de protección basados en salt e iteraciones.
- Rainbow tables** son una solución adecuada en situaciones reales para la creación de diccionarios (tradeoff entre tamaño del diccionario y tiempo de ejecución de prueba de cada potencial clave).



https://en.wikipedia.org/wiki/Rainbow_table

length	1-4	5	6	7	8	9	10	11	12	13	14	15	16
german	xp_german(7.4GB)												
special	xp_special(7.5GB)												
mixedalphanumeric	xp_free_small(380MB) and xp_free_fast(703MB)												

XP free small (380MB)
formerly known as SSTIC04-10k

Success rate: 99.9%
Charset: 0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
md5sum: 17cfa36c13e275236c1f23eb241bc86

Tipos de ataques – Computación

GPUs, HW específico y herramientas

- Uso de la potencia de la **GPU** (*Graphics Processing Unit*) para realizar cálculos a una mayor velocidad
- Herramientas (hashcat, Elcomsoft, Passware, ...)



It achieves the 350 billion-guess-per-second speed when cracking password hashes generated by the NTLM cryptographic algorithm that Microsoft has included in every version of Windows since Server 2003. As a result, it can try an astounding 95^8 combinations in just 5.5 hours, enough to brute force every possible eight-character password containing upper- and lower-case letters, digits, and symbols.

<http://arstechnica.com/security/2012/12/25-gpu-cluster-cracks-every-standard-windows-password-in-6-hours/>

<https://github.com/Coalfire-Research/npk>

Serverless distributed hash cracking platform – “NPK lets you leverage extremely powerful hash cracking with the 'pay-as-you-go' benefits of AWS. For example, you can crank out as much as 1.2TH/s of NTLM for a mere \$14.70/hr. NPK was also designed to fit easily within the free tier while you're not using it! Without the free tier, it'll still cost less than 25 CENTS per MONTH to have online!...”

Price Comparison			
Enthusiast Rig	8x GTX 1070	\$3,600 Upfront	240GH/s
Professional-Grade Rig	8x 1080Ti	\$11,000 Upfront	416GH/s
NPK with 1x g3.16xlarge	4x Tesla M60	\$1.37/hr	73GH/s
NPK with 1x p2.16xlarge	16x Tesla K80	\$4.32/hr	136GH/s
NPK with 1x p3.16xlarge	8x Tesla V100	\$7.34/hr	632GH/s



Acabamos de pasar el rockyou.txt entero para crackear un token JWT con una Tesla V100 en AWS en 2 segundos. Ah, y nos ha costado 1,5€ contando con encender y apagar la máquina.

#miedito 😊

```

Dictionary cache built:
* Filename...: rockyou.txt
* Passwords...: 14344391
* Bytes...: 139921497
* Keyspace...: 14344384
* Runtime...: 2 secs

Approaching final keyspace - workload adjusted.

Session.....: hashcat
Status.....: Exhausted
Hash.Type.....: JWT (JSON Web Token)
Hash.Target.....:
Time.Started....: Fri Jul 26 11:20:01 2019 (4 secs)
Time.Estimated...: Fri Jul 26 11:20:05 2019 (0 secs)
Guess.Base.....: File (rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 4391.5 kH/s (5.89ms) @ Accel:256 Loops:1 Thr:64 Vec:1
Recovered.....: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.....: 14344384/14344384 (100.00%)
Rejected.....: 0/14344384 (0.00%)
Restore.Point...: 14344384/14344384 (100.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1...: $HEX[31393132383873616c] -> $HEX[042a0337c2a156616d6f732103]
Hardware.Mon.#1...: Temp: 34c Util: 20% Core:1447MHz Mem: 877MHz Bus:16
  
```

1:27 p. m. · 9 sept. 2019 ·

Tesla V100 – 7300 euros
1× Tesla V100: **52 729,6 MH/s (!)**, p3.16xlarge
instance with 8× GPU does 421,8 GH/s (!!!)

17

Demo

- Creación y uso de diccionarios
- Expansión de diccionarios (JohnTheRipper)
- Cracking autenticación en línea (Hydra)
- Hashcat
 - Modos de uso
 - Cracking clave usuarios UNIX
- JohnTheRipper
 - Recuperando la clave privada de un certificado protegido con contraseña
- Software variado de cracking

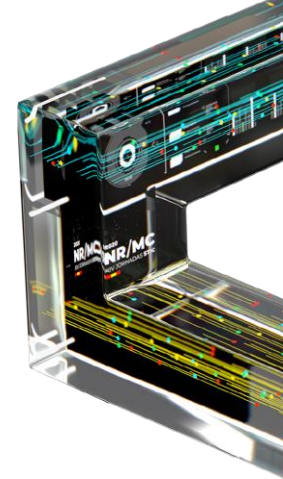
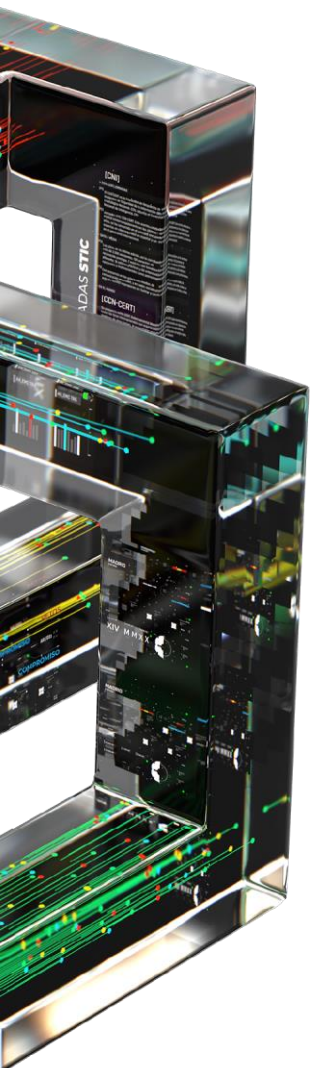
<https://github.com/mindcrypt/cracking>

<https://github.com/mindcrypt/Cryptanalysis>

michael	shadow	000000	myspace	inuyasha	mustang
ashley	melissa	diamond	rebelde	peaches	isabel
qwerty	eminem	carolina	angel1	veronica	natalie
111111	matthew	steven	ricardo	chris	cuteako
iloveu	robert	rangers	babygurl	888888	javier
000000	danielle	louis	heaven	adriana	789456123
michelle	forever	orange	55555	cutie	123654
tigger	family	789456	baseball	james	sarah
sunshine	jonathan	999999	martin	banana	bowwow
chocolate	987654321	shorty	greenday	prince	portugal
password1	computer	11111	november	friend	laura
soccer	whatever	nathan	alyssa	jesus1	777777
anthony	dragon	snoopy	madison	crystal	marvin
friends	vanessa	gabriel	mother	celtic	denise
butterfly	cookie	hunter	123321	zxcvbnm	tigers
purple	naruto	cherry	123abc	edward	volleyball
angel	summer	killer	mahalkita	oliver	jasper
jordan	sweetie	sandra	batman	diana	rockstar
liverpool	spongebob	alejandro	september	samsung	january
justin	joseph	buster	december	freedom	fuckoff
		george	morgan	angelo	alicia
			marinosa	kenneth	nicholas



Conclusiones



MUCHAS GRACIAS

#XIVJORNADASCCNCERT



alfonso@criptored.com



[@mindcrypt](https://twitter.com/mindcrypt)



t.me/criptored



[http://github.com/mindcrypt](https://github.com/mindcrypt)



<https://es.linkedin.com/in/alfonsomuñoz>