Computer Security Bootcamp

Alessandro Barenghi

Politecnico Open unix Lab

18 novembre 2021

Un security bootcamp

. . . per 10_2 tipi di persone

0_2 per non appassionati di informatica

- ... che devono comunque conviverci

1_2 per appassionati di informatica

- ... che sono ben felici di conviverci

Sicurezza informatica

Qualche domanda prima di partire

- Cosa vogliamo difendere?
  - Dati che riteniamo personali
  - Modi di provare la propria identità digitale
- Da chi o cosa lo vogliamo difendere?
  - *the spooks*: il mondo post Snowden revelations (2013-6-7)
  - *the crooks*: malintenzionati (contro proprietà) classici, v.2.0
  - *the swamp*: malintenzionati (contro la persona) classici, v.2.0
  - *the geeks*: curiosi, for fun, or fame

Sicurezza informatica e FLOSS[1]

Four Freedoms e sicurezza informatica

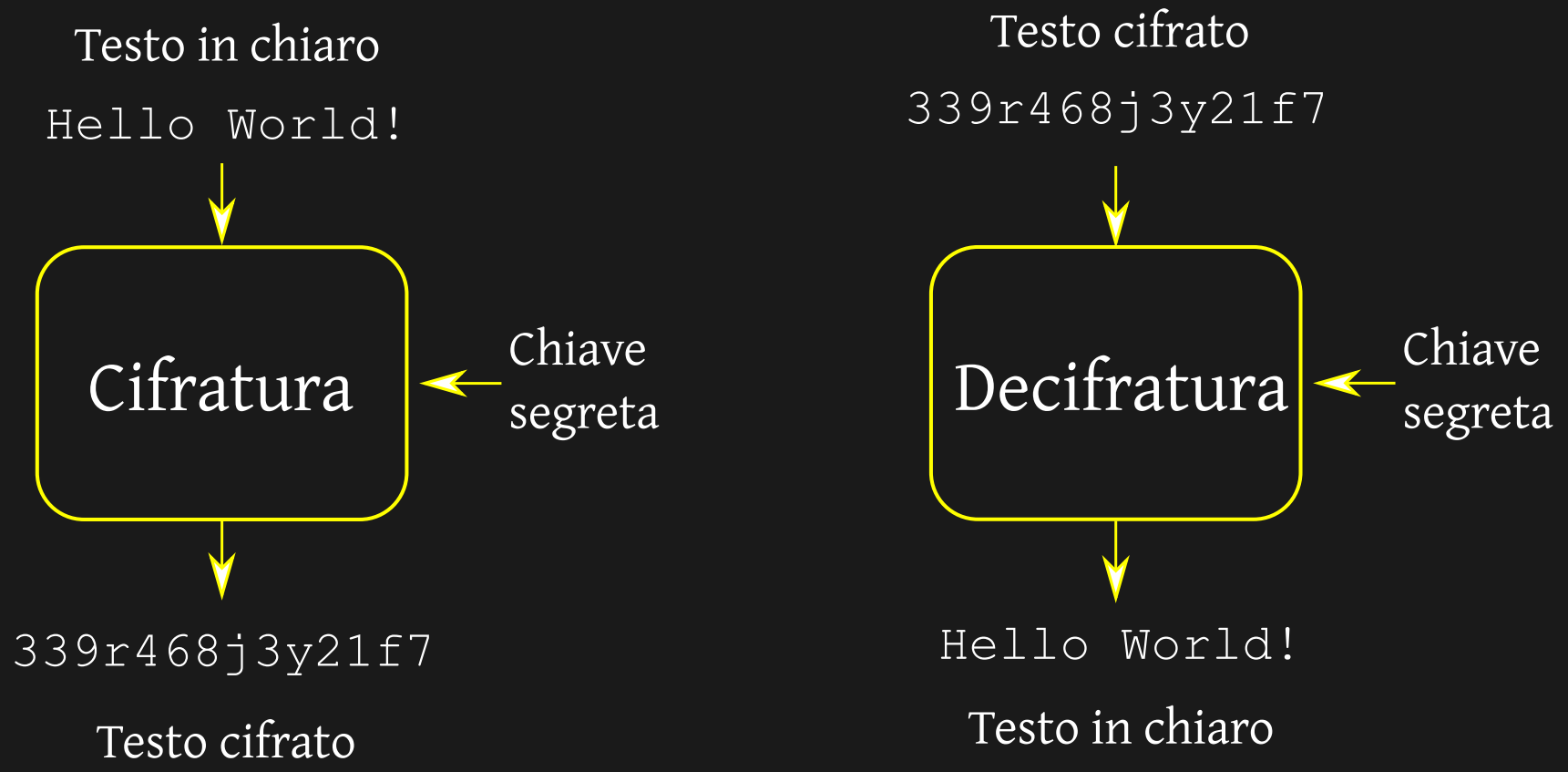
0. Libertà di esecuzione: fornisce a tutti gli strumenti informatici
1. Disponibilità del sorgente: consente a tutti di ispezionare/modificare il programma
2. Libertà di ridistribuzione: consente di distribuire copie a chi ne ha necessità
3. Libertà di ridistribuzione delle modifiche: consente di contribuire versioni migliori

[1] non il filo interdentale, il Free/Libre Open Source Software

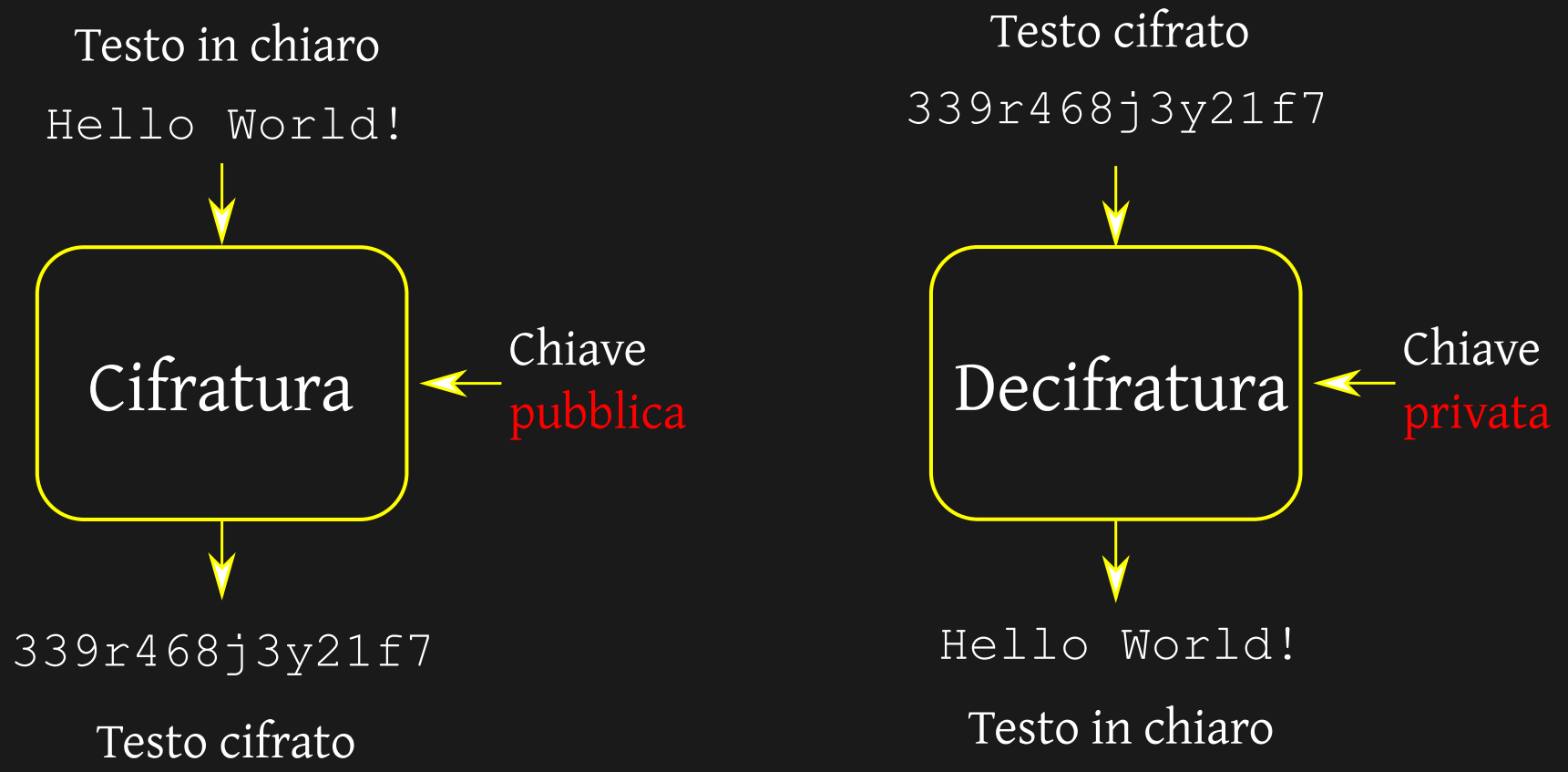
Sommario

- μintroduzione alla crittografia
- Password, chiavi, portafoglio e altro che spiace perdere
- Comunicare in modo sicuro
  - per procurarci informazioni: navigazione
  - con altri esseri umani, a distanza

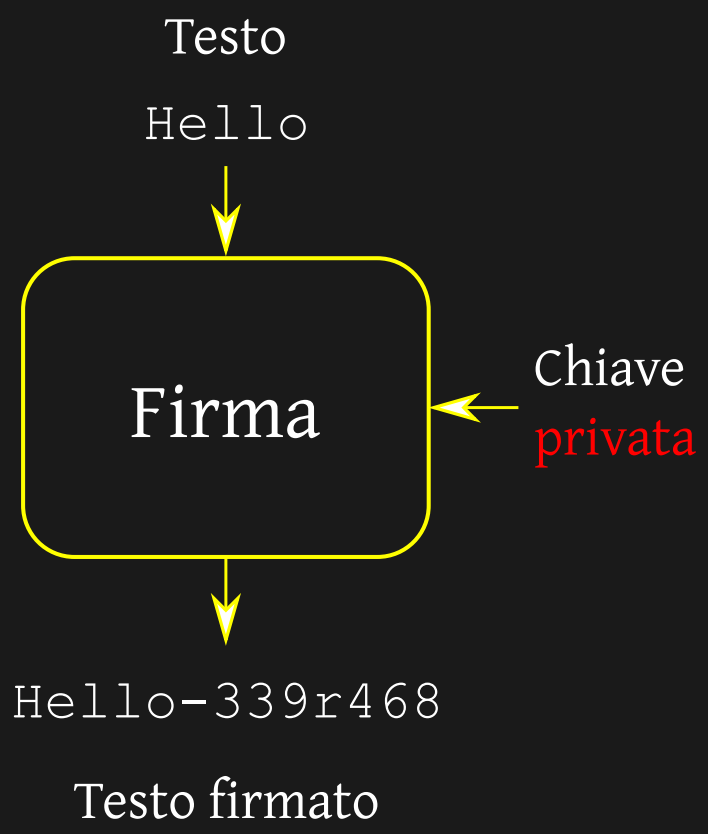
Cifrario simmetrico

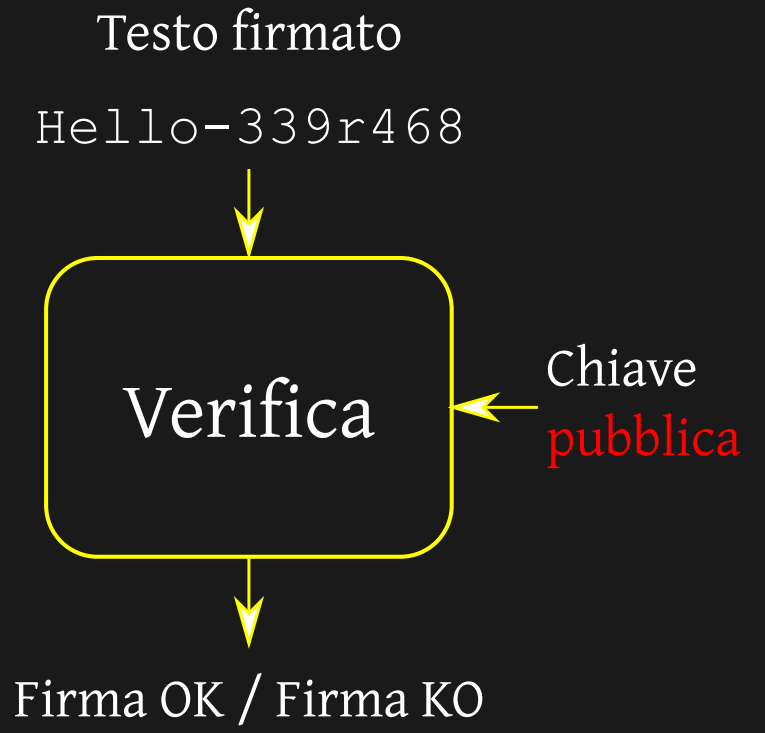


Cifrario asimmetrico



Firma digitale





Funzione di Hash crittografica

- Il digest di un testo è indistinguibile da uno sorteggiato a caso, se non conosco il testo
- Digerire due volte lo stesso testo, produce lo stesso digest
- É difficile™ trovare due testi con lo stesso digest

Password, chiavi, portafoglio e altro che spiace perdere

Password e categorie mentali

- *Spesso (prima di questo talk)*: una scocciatura necessaria
- *Dopo il talk*: la versione digitale delle chiavi di casa

Usi (in brevissimo)

- Provare la propria identità
- Derivare una chiave segreta (e.g. `hash(password)`)

Password: come sceglierle?

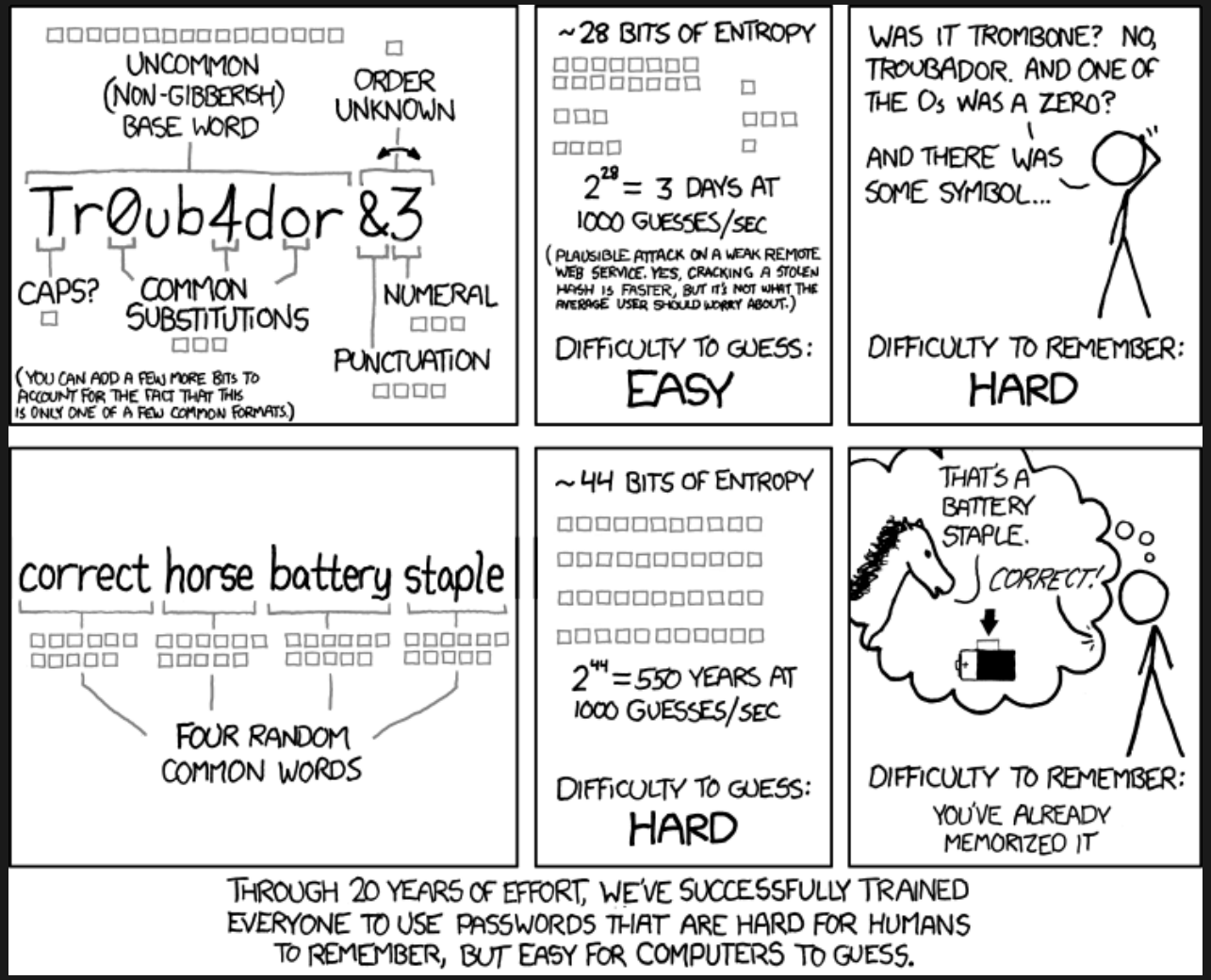
Ideale per massimizzare la fatica dell'attaccante

- Uniformemente a caso da un insieme enorme di possibilità
- Una password = un utilizzo: tutte diverse tra loro

Ideale per minimizzare la fatica utente

- Facili da ricordare
- Il minor numero possibile

Password facili da ricordare



https://xkcd.com/936

Diceware

Scegliere a caso

- Generatori di numeri casuali
  - un generatore di numeri casuali digitali fidato
  - i dadi! (consigliati d6 o d20)

Quante parole?

- Sorteggiando da una lista da 4000 parole:
  - 3 parole: $64 \cdot 10^9 \approx 2^{36}$ possibilità (ok contro guessing online)
  - 6 parole: $4 \cdot 10^{21} \approx 2^{72}$ possibilità (ok contro bruteforcing)
  - 9 parole: $256 \cdot 10^{30} \approx 2^{108}$ possibilità (ok.)
- Usando la capacità massima storica di calcolare hash dell'intera rete di Bitcoin ($191 \cdot 10^{18}$ Hash/s)
  - indovino una password da 6 parole in 5.2s (1 Hash per test)
  - ottengo, in 5.2s, $5.2 \times 6.25 = 32.5$ Bitcoin ≈ 1.95M\$

Hardware o software free?

Pen – dice – and paper

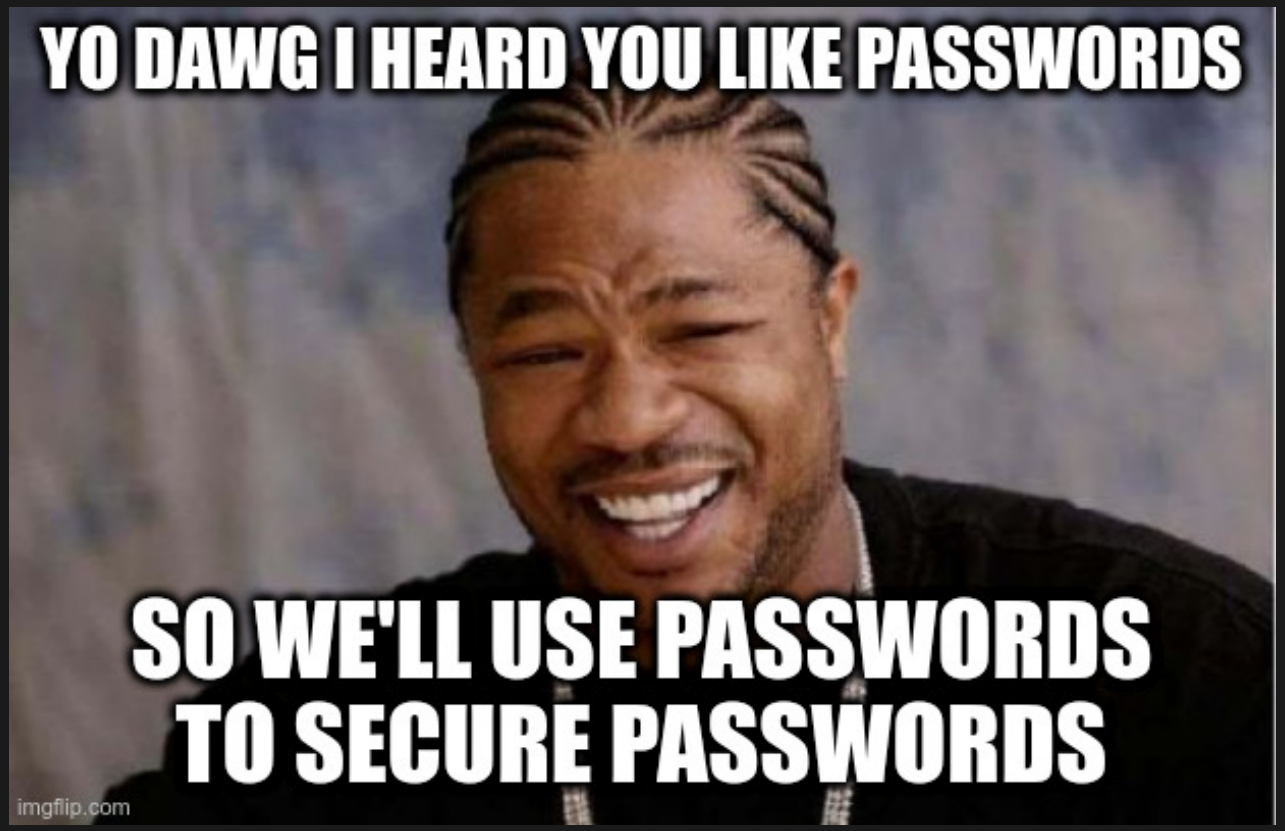
- Carta, penna e 5d6 (o 3d20)
- Lista di parole indicizzate con cifre delle facce del dado:

```
12115 antacid
12116 antarctic
12121 anteater
12122 antelope
```

Software free

- Generatore FLOSS: https://www.rempe.us/diceware/
  - In Javascript, gira in locale, supporto per 10+ lingue
  - https://github.com/grempe/diceware, codice per self hosting

Poche password (da ricordare)



Password Managers (a.k.a. password wallets)

Caratteristiche

- Mimano i password wallet fisici: taccuini in carta
- Struttura:
  - password sono salvate in un file cifrato con cifrario simmetrico
  - la chiave del cifrario è ricavata da una master password
- Serve ricordare e scegliere solo la master password

Implementazioni consigliate

- Keepass
- Bitwarden

Keepass

Caratteristiche

- Logica KISS: singolo file cifrato, no temporanei, scelta di algoritmi solidi
- Codebase ridotta da ispezionare
- Port disponibili per Linux/MacOS (KeepassX), Android, iOS
- Puramente locale:
  - nessuna registrazione a servizi esterni...
  - ... e nessun backup remoto di default

Bitwarden

Caratteristiche

- Combinazione di programma + servizio remoto
- Codebase con ispezioni di terze parti, certificate, periodiche
- Salva localmente come Keepass, sincronizza tramite stoccaggio remoto
- Implementazioni disponibili per Windows/Linux/MacOS, Android, iOS, browser plug-in
- Server self-hostable per ridurre al minimo la fiducia da avere in terze parti

Comunicazioni sicure

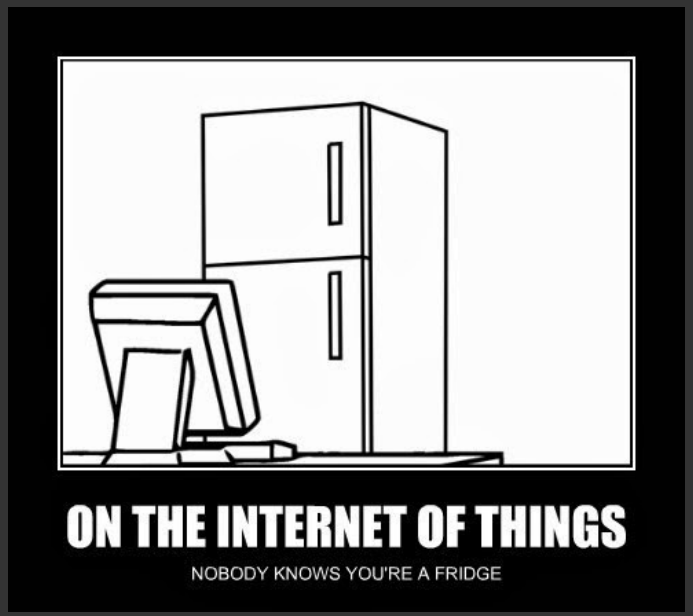
Comunicazioni digitali

- La sicurezza delle comunicazioni si regge sul fatto che le chiavi private o segrete siano note solo a chi si vuole che decifri
- Crittografia end-to-end: solo gli estremi in comunicazione sanno decifrare
  - Le chiavi sono decise dagli estremi in comunicazione
  - Serve verificare che le chiavi siano autentiche

Casi pratici

Navigazione (o simili)

- Distribuzione di contenuti da un server “non senziente”
- Anche il client potrebbe non essere senziente



Casi pratici

Comunicazione con altri esseri umani

- Comunicazione sincrona (chiamate audio/video)
  - posso sfruttare la presenza di entrambi i coinvolti per l'autenticazione
- Comunicazione asincrona (messaggistica/e-mail)
  - serve un intermediario che agisca da "parcheggio" dati

Cifratura ibrida

- Transport Layer Security usa uno schema di cifratura ibrida
- Ubiquo, praticamente il miglior sostituto del Cryptonium pipe[2]

[2] https://cseweb.ucsd.edu/classes/fa19/cse107-a/s-intro.pdf

TLS ovunque

Per HTTP

- (Sempre meno) spesso, TLS è offerto come opzione, traffico di default in chiaro
- HTTPS Everywhere: plugin per tutti i *browser*, tenta sempre di usare TLS per le connessioni, prima di passare al chiaro
  - https://www.eff.org/https-everywhere
- HTTP Strict Transport Security (HSTS): impostazione lato *server*: evitare che i client possano comunicare in chiaro
  - https://bettercrypto.org/#hsts

Per altro

- Stunnel (https://www.stunnel.org/) consente di fare wrapping trasparente di qualunque protocollo legacy su TCP in TLS

Messaggistica asincrona

- Posta elettronica (1971)
- SMS (1992): solo in chiaro, strutturalmente
- Messaggistica asincrona su IP
  - WhatsApp (2009)
  - WeChat (1/2011)
  - SnapChat (5/2011)
  - Telegram (2013)
  - Signal (2014)

Posta elettronica e OpenPGP

- La posta elettronica trasmette testo e allegati in chiaro
  - è nata prima della crittografia asimmetrica
- 1991: Phil Zimmermann inventa Pretty Good Privacy (PGP), cifratura ibrida per e-mail
- 1999: Werner Koch scrive GNU Privacy Guard (GPG)
- GPG è lo standard de-facto per la cifratura di e-mail
  - Ad ogni utente sono associate coppie di chiavi per cifrare messaggi e firmare messaggi o chiavi pubbliche di altri
  - Integrato in Mozilla Thunderbird (was Enigmail)
  - https://ssd.eff.org/module-categories/tool-guides
- Il formato (OpenPGP) è usato anche al di là delle e-mail
- Firme OpenPGP usate per autenticare anche i pacchetti delle distribuzioni Linux

WhatsApp

- All'inizio: tutte le comunicazioni in chiaro
- Ad aprile 2016 adotta il protocollo di Signal per la cifratura delle chat e delle chiamate
  - default: chiavi non autenticate, autenticazione da op. utente
- È closed source, ma qualche analisi è stata fatta.[3] TL;DR
  - L'adozione del protocollo di Signal è effettiva
  - Riusa librerie crittografiche solide ...
  - ... con qualche bug nel controllo di integrità del traffico

[3] https://medium.com/@schirrmacher/analyzing-whatsapp-calls-176a9e776213

Telegram

- Pensato per cifratura end-to-end nelle chat tra utenti singoli (secret chats)
- Protocollo crittografico pensato da zero (MTProto v1)
  - Uso di cifratura simmetrica relativamente poco comune (ma non problematica)
  - Qualche difetto di progettazione (hash vecchie, padding)
- MTProto v2 corregge alcuni problemi ma ha vulnerabilità in autenticazione
  - Analisi (7/2021) in: https://mtpsym.github.io/paper.pdf

Signal

- Cifratura end-to-end (con verifica delle chiavi)
- Contact discovery usando hash dei numeri di telefono
- Metadati ridotti alla data dell'ultima connessione
- Code audit di terze parti & protocol audit
  - Rivisto anche da alcuni degli autori di TLS
- Forward-secure

(Video)chiamate sicure

Dati trasmessi

- Secure Real-time Transport Protocol (SRTP) / SRTP Secure Control Protocol (RTCP)
  - "Simile" a TLS, delega autenticazione delle chiavi a chi lo usa

Autenticazione delle chiavi

- Si approfitta della comunicazione sincrona: viene chiesto di leggere un messaggio ad alta voce (ZRTP)
  - sintetizzare al volo la voce dell'interlocutore con un messaggio diverso è ritenuto "abbastanza difficile" (RFC 6189 - Sec. 15)
  - la difficoltà aumenta con una videochiamata

(Video)chiamate

Signal

- Supporta chiamate e videochiamate cifrate end-to-end
- Inizialmente, autenticazione chiavi solo tramite ZRTP
- Attualmente, usa l'autenticazione long-term, se disponibile, in modo preferenziale

Jitsi

- Opera sia in modalità peer-to-peer che centralizzata (Jitsi Videobridge)
- Policy di no-logging del traffico in modalità centralizzata
- Cifratura a livello di trasporto (dai client al server) sempre abilitata (usa SRTP+DTLS)
- Cifratura end-to-end disponibile in modalità peer-to-peer
  - Supporto recente, inserimento chiave simmetrica... a mano :)
  - https://jitsi.org/e2ee-in-jitsi/
- Disponibili server pubblici, oppure possibilità di self-hosting completo

(Video)chiamate

Stato delle soluzioni proprietarie (e closed source)

- Microsoft Teams: default trasporto cifrato senza verifica chiavi estremi, dal 21 Ott. 2021, end-to-end encryption disponibile
- Zoom: end-to-end encryption... tra il client e la *cloud Zoom*
- Cisco Webex Meetings: end-to-end encryption abilitabile da chi crea la riunione (usa ZRTP)

Furti

... fisici

- I dispositivi elettronici sono spesso bersaglio di furti
  - con loro vengono rubati anche i dati che contengono

... e digitalizzati

- Nel mondo fisico è più facile sottrarre un bene, che non impedire l'accesso senza spostarlo
- Per beni digitali, un attaccante può:
  - cifrare i dati sul posto (con cifratura ibrida)
  - chiedere un riscatto per rivelare la chiave privata
  - È il principio alla base del *ransomware*

Cifratura dei dati su dispositivo

Laptop (e desktop?)

- Linux supporta cifratura dell'intero disco fisso
  - Specifica: LUKS, implementazione: cryptsetup/dm-crypt
  - Deriva una master-key da password (ibrido simm.-simm.)
  - Overhead praticamente impercettibile

Smartphone

- Android $\leqslant 9$ utilizza lo stesso meccanismo di $\uparrow$
- Android $\geqslant 10$ cifra individualmente i file
  - `fscrypt`: parte nel gestore del FS, utility omonima
  - Protegge contenuti e nome dei file, non altri metadati (data)
- Master key derivata da pin/pattern di sblocco/password

Riprendersi da un ransomware

Due approcci

- Elegante: Tentare di capire, aumentando il sistema operativo, quando qualcosa inizia a cifrare masse di dati e fermarlo
- Grezzo: Fare dei backup, periodici, separati

Backup come/dove?

- Su un disco, rimov
- ~~Nel cloud~~ Sul computer di altri
  - `rclone`: come rsync, verso servizi computer-di-altri based, cifratura a monte integrata
- ~~In un private cloud~~ Su un altro mio computer
  - ZFS: filesystem con capacità di versioning
  - Nextcloud: self-hosted cloud storage

Grazie per l'attenzione!

Domande?

Ask Me Anything

Riferimenti e licenza

Riferimenti

- Lista diceware (d6) con parole in inglese (15625 termini): https://www.eff.org/files/2016/07/18/eff_large_wordlist.txt
- Lista diceware (d6) con parole in italiano (15625 termini): `https://www.taringamberini.com/downloads/diceware_it_IT/lista-di-parole-diceware-in-italiano/4/word_list_diceware_it-IT-4.txt`
- Liste diceware (d20) con parole in inglese (4000 termini unici): https://www.eff.org/deeplinks/2018/08/dragon-con-diceware

Licenza