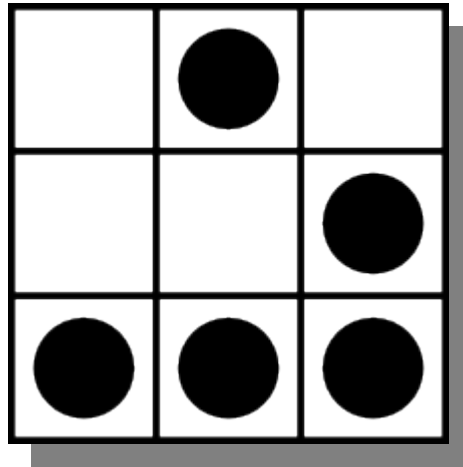


# Sicurezza - Manuale d'uso

---



Giacomo Rizzo [ a.k.a. alt-os ]

alt@free-os.it



# Che cos'è la Sicurezza?

---

- La maggior parte delle persone associa alla Sicurezza un concetto simile a: “Assenza di violazioni”, “niente virus” e via dicendo.

**SBAGLIATO**

# La sicurezza

---

- **Confidenzialità [violazione]**
  - Significa non dare accesso ai dati a chi non ne ha diritto.
  - Riguarda sia ai dati residenti su disco, che a quelli in transito sulla rete (autenticazione, cifratura del canale)
- **Integrità [defacement]**
  - Significa che i dati non vengono manomessi
  - Questo può accadere sia durante il transito in rete, sia nella locazione originale dei dati (su disco)
- **Disponibilità [denial of service]**
  - Se non ci sono dati, a cosa serve che siano confidenziali e integri?
  - Nella società dell'informazione è il bene più prezioso

# I problemi della Sicurezza

---

- Quello di identificare cosa sia la Sicurezza è parte di uno dei problemi che ha chi lavora nel campo. Altri problemi, ad esempio, sono:
  - L'idea che la sicurezza sia un obiettivo raggiungibile
  - L'illusione dello stato di sicurezza
  - Sacrificare la sicurezza in cambio di performance
  - Sacrificare la sicurezza in cambio di funzionalità
  - Sacrificare la sicurezza in cambio di comodità
  - Dimenticare della regola dell'Anello Debole
  - La crittografia risolve tutti i problemi
  - Pigrizia nella ricerca delle soluzioni

# La Sicurezza esiste?

---

- Molte persone sono convinte che “la sicurezza” sia una condizione effettivamente raggiungibile.
- In realtà la “sicurezza” così come queste persone la intendono, non esiste.
- Si deve pensare al processo di Sicurezza come quello di definizione di un “compromesso”, tra l'usabilità, integrità, confidenzialità e disponibilità.
- A volte sento dire “l'unico sistema veramente sicuro è quello spento, sottoterra, sorvegliato da una guardia armata”.
- Beh, vi state fidando della guardia armata.
- Di qualcuno, prima o poi, bisogna sempre fidarsi, fossimo anche solo noi stessi (alcool? minacce?)

# L'illusione della Sicurezza

---

- Un'altro dei rischi piu grandi, è quello del sentirsi sicuri.
- Sentendosi sicuri, si tende ad abbassare la guardia, a prestare meno attenzione, finendo con l'esporsi maggiormente.
- La sensazione di sicurezza, oltretutto, è solitamente data dall'introduzione di palliativi, come una crittografia male applicata, che contribuiscono ad abbassare ulteriormente il già precario livello di sicurezza del sistema oggetto.
- Un livello minimo di paranoia è assolutamente imprescindibile per le persone che si occupano di sicurezza.
- Per deformazione professionale, si passa la vita a cercare un modo per violare la sicurezza di tutto ciò che si vede.

# L'illusione della Sicurezza

---

- L'ambiente in cui lavora l'Ing. della Sicurezza è più difficile di quello in cui lavora l'Ing. Edile.
- Costruire un ponte pone molti problemi, ma la maggior parte di questi sono noti e/o prevedibili:
  - Il vento
  - Il carico
  - Il calore
  - L'usura
- L'ambiente circostante non è malevolo, ma quasi sempre omogeneo a se stesso (muta poco)
- Una volta fatti i calcoli necessari e trovate le relative soluzioni tecniche, ci si può considerare ragionevolmente al sicuro (conseguenze peggiori)

# L'illusione della Sicurezza

---

- Nell'ambiente della sicurezza informatica invece, le avversità non solo non sono prevedibili, ma sono molto variabili e (soprattutto) malintenzionate.
- E' come se, costruito il ponte, improvvisamente il vento cominciasse a soffiare dal basso verso l'alto, o ritmicamente, in modo da portare il ponte in risonanza e farlo crollare.
- Il nostro avversario, nel mondo della sicurezza è:
  - Intelligente
  - Capace
  - Malintenzionato
  - ha FANTASIA (trova soluzioni fino a ieri impensate)

# L'illusione della Sicurezza

---

- Difensore:
  - Deve difendere tutti i fronti
  - Non conosce con chiarezza gli obiettivi dell'ipotetito attaccante, ne i metodi ne tantomeno le competenze
  - Deve agire in anticipo, e una volta realizzata la soluzione, è quella e difficilmente la si può cambiare
- Attaccante
  - Deve trovare un solo punto debole
  - Ha dalla sua anni di studio
  - Sfrutta l'evoluzione tecnologica (strumenti, conoscenze)
  - Si può documentare e riprovarci
  - Sfrutta il lavoro di altri

# Barattare la sicurezza

---

- Spesso la sicurezza finisce con il passare in secondo piano nella progettazione nello sviluppo di un applicativo.
- Gli aspetti che la riguardano finiscono così con l'essere letteralmente “barattati” con:
  - Maggior performance
  - Maggiori funzionalità
  - Maggior comodità/semplifictà d'uso
- Si tratta di un errore piuttosto grossolano, perchè una sicurezza pensata a livello di progetto è spesso più facile da portare avanti del tentativo di “tappare i buchi”
- Vediamo nel dettaglio...

# Barattare la sicurezza

---

- Barattare la sicurezza in cambio di performance:
  - Nella scala delle priorità progettuali degli ingegneri del software, solitamente, le performance vengono MOLTO prima della Sicurezza.
  - Il clock di un computer desktop, oggi, è mediamente occupato al 99% dal ciclo IDLE. Su un server, questo scende ADDIRITTURA al 97.3%
  - I computer quindi non fanno NULLA per la maggior parte del tempo.
  - Sacrificare determinate operazioni basilari per la sicurezza perchè “rallentano il sistema” è assolutamente inutile.
  - Un computer che utilizzi il 90% della sua CPU per operazioni di sicurezza, sarebbe comunque utilizzabile e la maggior parte degli utenti non se ne renderebbe nemmeno conto.

# Barattare la sicurezza

---

- Barattare la sicurezza in cambio di funzionalità:
  - L'introduzione di funzionalità all'interno di uno stesso “applicativo” o “sistema” non fa che virtualmente abbassarne il livello di sicurezza.
  - Infatti maggiori sono le funzionalità disponibili, maggiore è la possibilità che un errore nella progettazione o nella realizzazione di una di esse introduca una possibile linea di attacco
  - A questo si aggiunge che l'interazione tra funzionalità diverse moltiplica le possibilità di attaccare un sistema.
  - La soluzione è nella filosofia KISS (Keep It Simple Stupid) nota anche come “modularizzazione”, che troppo spesso viene dimenticata da chi progetta il software.
  - E' piu facile mantenere, aggiornare e correggere problemi riprogettando il modulo, senza ritoccare l'intero sistema

# Barattare la sicurezza

---

- Barattare la sicurezza in cambio di comodità:
  - La sicurezza è scomoda e difficile.
  - L'utente che “non vuole far fatica”, è un punto debole nella “catena della sicurezza”
  - Usare come password il nome del proprio gatto, non è il solo problema (l'imposizione di password complesse può addirittura rivelarsi controproducente in questi casi)
  - Anche l'utilizzo di applicativi che consentono la “memorizzazione di passphrase per non doverla digitare tutte le volte” sono estremamente pericolosi e vanificano quasi del tutto l'uso della crittografia
  - Se si vuole essere ragionevolmente sicuri, bisogna essere disposti a sacrificare sia parte della semplicità d'uso sia (soprattutto) la comodità.

# Dimenticare l'anello debole

---

- “La sicurezza di un sistema coincide con la sicurezza del suo anello piu debole”
- E' fondamentale tenerlo a mente
- Inutile predisporre una porta blindata a protezione di una tenda di tela
- L'obiettivo primario della messa in sicurezza di un sistema sta nell'individuazione dell'anello piu debole del sistema, e nella messa in sicurezza di quello.
- L'anello piu debole non è sempre unico, e spesso varia a seconda di competenze, strumenti e piaceri dell'attaccante
- Spesso si trova tra il monitor e la sedia (Cultura della S.)

# Individuare l'anello piu debole

---

- Il processo di individuazione dell'anello piu debole di un sistema si basa sull'immedesimazione di chi lavora alla sicurezza con l'attaccante.
- Bisogna guardare al proprio sistema come al bersaglio, e tentare di trovare un modo per entrarvi.
- Questo porta alla deformazione professionale della “paranoia” di cui parlavo prima.
- Questo può portare la “battaglia” sul piano personale: ne sa piu il sysadmin o l'attaccante?
- E' un errore da evitare: quando si attacca un sistema, non si attacca il suo sysadmin. Se non si impara questo (ambo i fronti), non si sopravvive.

# Individuare l'anello piu debole

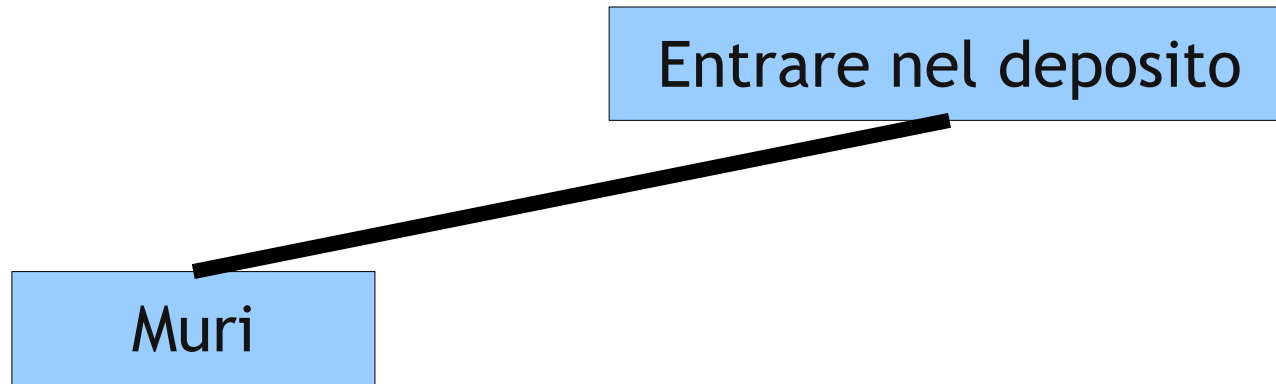
---

Entrare nel deposito

Il nostro obiettivo è entrare nel deposito di Zio Paperone...

# Individuare l'anello piu debole

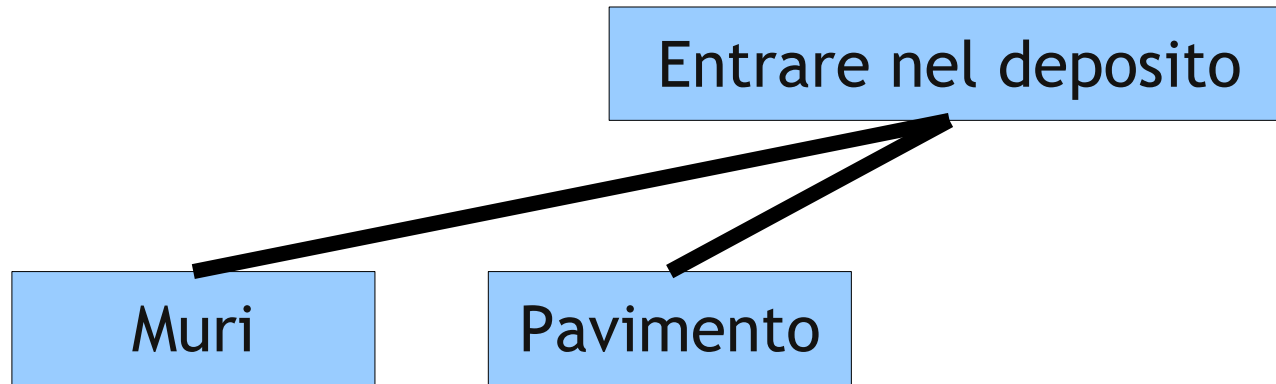
---



Possiamo passare attraverso i muri...

# Individuare l'anello piu debole

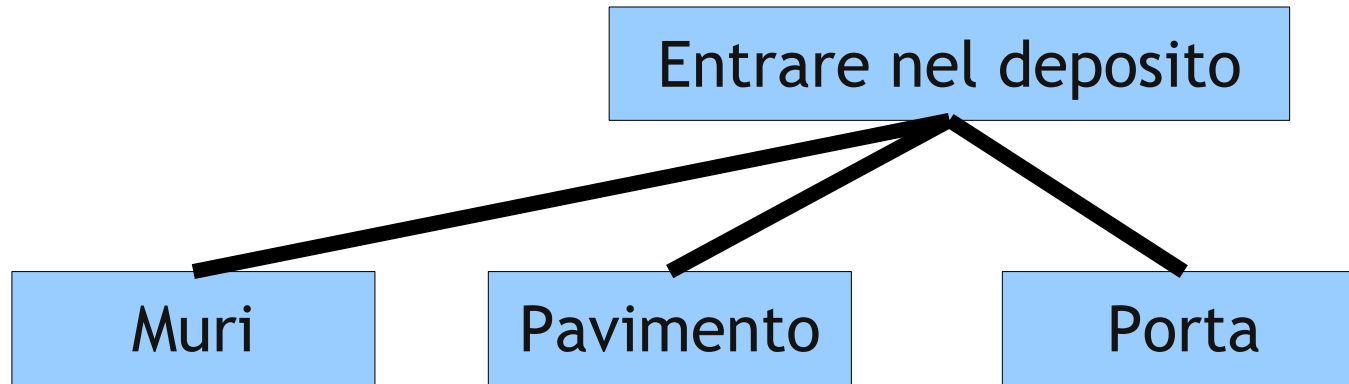
---



... attraverso il pavimento...

# Individuare l'anello piu debole

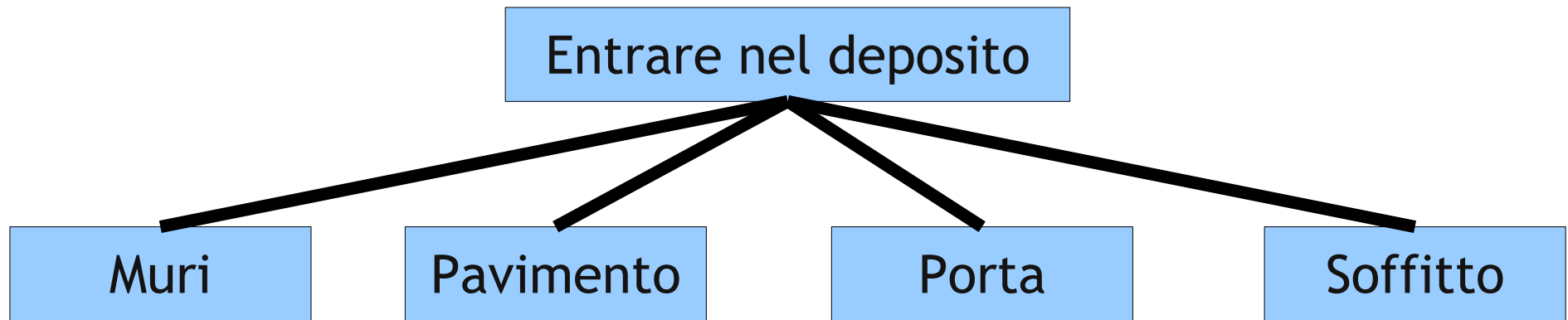
---



... attraverso la porta...

# Individuare l'anello piu debole

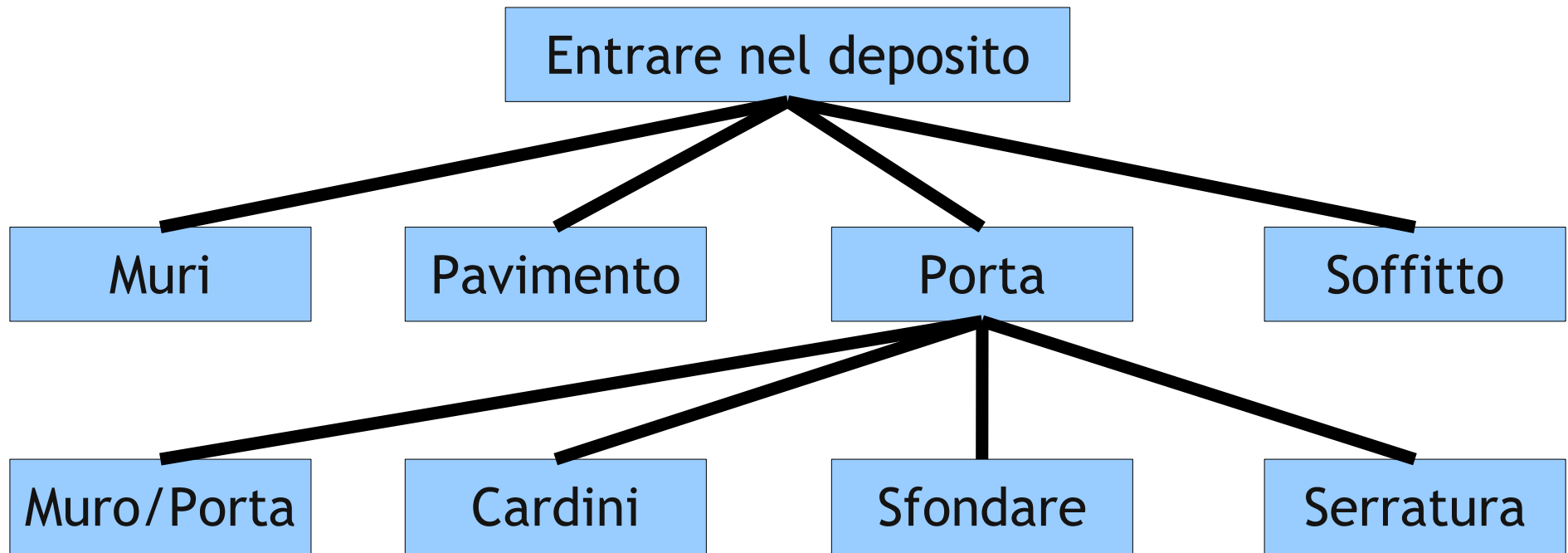
---



... o ancora calandoci dal soffitto. Tra queste, è sicuramente la porta il punto piu debole, quindi concentriamoci su quello.

# Individuare l'anello piu debole

---



Possiamo lavorare sull'interfaccia tra il muro e la porta, scardinare o sfondare la porta, oppure lavorare sulla serratura. Individuiamo il punto piu debole e procediamo.

# Individuare l'anello piu debole

---

- Questo genere di lavoro, che va ripetuto fino ad arrivare ai componenti singoli, si chiama “Albero d'attacco”.
- E' un modo pratico per tenere traccia dello stato di sicurezza del sistema, e per evitare di perdere tempo alzando continuamente il livello di sicurezza di alcuni componenti del sistema dimenticandone completamente altri.
- Bisogna inoltre ricordarsi che la maggioranza degli attacchi informatici proviene dall'interno della struttura:
  - Guardare prima in casa nostra che fuori
  - Guardare prima agli amici che hai nemici
  - Paranoia!

# La crittografia non risolve tutto

---

- Spesso la “risposta” ai problemi di sicurezza sta nell'introduzione della crittografia.
- Gli “utenti”, vedendo il lucchetto sul loro browser, si sentono al sicuro e non si lamentano più
- Ma la crittografia è solo la serratura della porta: a cosa può mai servire mettere una porta blindata all'ingresso di una tenda di tela? (Anello più debole)
- Allo stesso modo, ottenere cifratura e non ripudio dal canale di comunicazione non è una soluzione:
  - L'attaccante può introdursi ad uno dei due estremi del canale, magari semplicemente rubando la chiave privata
  - Ci sono alcuni errori comuni nell'implementazione della crittografia che la portano ad essere inutile

# La crittografia non risolve tutto

---

- Alice e Bob stanno comunicando, con un canale cifrato:
  - Eva intercetta i messaggi, e pur non potendoli leggere può senza problemi:
    - alterarne la sequenza
    - cancellarne alcuni
    - duplicarne altri
    - invertirne l'ordine
  - Nonostante la cifratura del canale, la trasmissione viene manomessa e non è quindi più affidabile, ne tanto meno sicura.
- La cifratura del canale NON E' una soluzione.
- E' una parte.

# La crittografia non risolve tutto

---

- Alice e Bob stanno comunicando, con un canale che tramite un numero di sequenza ed un hash di integrità, si può considerare integro e sequenziale:
  - Eve che intercetta i messaggi può solo interrompere la comunicazione, o far perdere alcuni messaggi.
  - A questo si trova facilmente soluzione con un meccanismo di richiesta di ritrasmissione dei pacchetti persi
  - Ma Eva può LEGGERE il messaggio.
  - Nonostante l'integrità, non c'è riservatezza.
- Non è una soluzione.
- E' una parte.
- La composizione di più parti, porta ad innalzare il livello di sicurezza del componente preso in considerazione.

# La crittografia non risolve tutto

---

- Oltretutto la crittografia è spesso la parte più semplice del sistema di sicurezza, in quanto le sue funzionalità ed i suoi limiti sono piuttosto ben definiti.
- Il resto del sistema, i cui limiti e le cui funzionalità sono molto meno definite, è la vera parte “difficile” del sistema di sicurezza.
- Per fare un esempio: recentemente Schneier spiegava sul suo blog il perché secondo lui la crittografia sia un pericoloso palliativo per le comunicazioni in Internet:
  - L'inviolabilità del canale trasmissivo sposta l'obiettivo dell'attacco dal canale alla sua origine.
  - Bucato il sistema in un altro punto, il fatto che il canale non sia violabile è assolutamente indifferente. Lo si può persino comodamente utilizzare a partire da “Alice”.

# Attacco al cifrario

---

- Spesso si immagina un attacco contro un algoritmo crittografico come il risultato di qualche scenziato pazzo che studiando il codice sorgente dell'implementazione dell'algoritmo, oppure il modello matematico che si trova alla base dello stesso, ha trovato un modo per decifrarlo avendo a disposizione il solo testo cifrato.
  - Si tratta sicuramente dell'immagine piu caratteristica e piu poetica, ma spesso e volentieri le cose non vanno cosi, e anzi, risultano essere dannatamente piu semplici
- Esistono diverse tipologie di attacchi ad una chiave:
  - Con solo testo cifrato
  - Con testo in chiaro noto
  - Con testo in chiaro scelto
  - A convergenza (o “delle collisioni”)

# Attacco alla crittografia

---

- Se pensate che sia difficile ottenere il testo in chiaro per portare a termine un attacco “con testo in chiaro noto”, beh, siete in errore.
  - Pensate ad esempio ai risponditori automatici:
    - Voi inviate un messaggio ad Alice, e questa vi risponde fornendovi il testo in chiaro
    - Poi intercettate il messaggio destinato a Bob, e avete il testo cifrato
- Lo stesso discorso vale per gli attacchi con “testo in chiaro scelto”: ci sono molti modi per portare un utente ad inviare un testo ben definito.
- Un algoritmo crittografico degno di questo nome (appena appena decente) non ha problemi a resistere a questo genere di attacchi anche per lungo tempo.

# Attacco alla crittografia

---

- Tutta un'altra storia invece è quella degli attacchi “a collisione”, o “del calendario”.
- Si tratta di attacchi piuttosto onerosi computazionalmente parlando da parte dell'attaccante, ma ricordatevi sempre che ha dalla sua parte tecnologia e tempo.
- In cosa consiste un attacco “a collisione”:
  - Supponiamo che l'algoritmo preveda una chiave di 64 bit
  - Essendo chiavi binarie, avremo  $2^{64}$  chiavi possibili (poche)
  - Per il paradosso del compleanno, trovare collisioni con questo numero di chiavi è sorprendentemente più semplice di quanto non appaia a prima vista

# Attacco alla crittografia

---

- “Paradosso del compleanno?”
  - Quante persone servono, perchè la probabilità che due abbiano la stessa data di nascita sia superiore al 50%?

# Attacco alla crittografia

---

- “Paradosso del compleanno?”
  - Quante persone servono, perchè la probabilità che due abbiano la stessa data di nascita sia superiore al 50%?

23

# Attacco al cifrario

- “Paradosso del compleanno?”
  - Quante persone servono, perchè la probabilità che due abbiano la stessa data di nascita sia superiore al 50%?

23

- Infatti questa probabilità ( $P_{(n)}$ ) è la complementare del fatto che tutte le date siano diverse ( $P_{1(n)}$ ). Essendoci 365 giorni in un anno, questa probabilità si calcola con:

$$P_{(n)} = 1 - P_{1(n)} = 1 - \frac{364}{365} * \frac{363}{365} * \dots = 1 - \frac{365!}{365^p (365 - p)!}$$

# Attacco al cifrario

---

- Approssimando (molto, ma non tanto da rendere i calcoli errati), possiamo dire che dato un insieme di  $N$  elementi, ci sono buone probabilità di imbattersi in una collisione prendendo  $\sqrt{N}$  elementi.
- Questo ci porta ad affermare che, osservando passare  $2^{32}$  transazioni identiche (header?) cifrate con chiavi diverse, ci si può aspettare che quella successiva utilizzi la stessa chiave di una già utilizzata.
- Una volta trovata la collisione, potremo banalmente riutilizzare i messaggi della transazione precedente per inserirli nella comunicazione, e questi verranno accettati in quanto firmati correttamente e non alterati nel loro contenuto.
- Questo attacco viene chiamato “attacco del compleanno”

# Attacco al cifrario

---

- Ma c'è di meglio!
- Potendo permettersi il lusso di generare  $2^{32}$  chiavi diverse, dopo  $2^{32}$  transazioni non solo ci potremmo imbattere in una collisione, ma sarà facile che si tratti di una di quelle da noi generate, e di cui quindi possederemo la chiave segreta!
- A questo punto non solo possiamo riciclare vecchie transazioni, ma anche di generarne di nuove!
- Tecnica di “attacco della convergenza”
- Naturalmente il fatto che abbiamo scelto una chiave di 64 bit influisce notevolmente. C'è una bella differenza tra  $2^{32}$  e  $2^{1024}$ . Ogni bit in più, raddoppia il numero di combinazioni.

# Reale forza della crittografia

---

- Tutto è craccabile, ma in quanto tempo? Con che risorse?
- L'idea di base della crittografia, simmetrica o asimmetrica che sia, sta nel rendere “computazionalmente troppo costosa” la decodifica brute force del testo cifrato.
- Se mi ci vogliono 2000 anni per decifrare il codice, posso considerarlo sicuro (?)
- Ci sono naturalmente modelli matematici e semplificazioni che consentono di ridurre anche piuttosto notevolmente il peso computazionale della decodifica “forza bruta” (che una volta semplificata, proprio “forza bruta” non è)
- Inoltre dobbiamo fare i conti con la crescente potenza computazionale degli elaboratori. Quanto tempo resta?

# Reale forza della crittografia

---

- Se vogliamo che le nostre chiavi simmetriche durino oltre il 2030, dobbiamo prevedere una lunghezza di almeno 256 bit.
- Per quanto riguarda invece la crittografia asimmetrica, le cose sono un po' più complicate.
- La stragrande maggioranza degli algoritmi di cifratura a chiave pubblica attualmente in circolazione si basano infatti sulla “fattorizzazione”

Fattorizzazione è la scomposizione di un numero primo (molto grande) in numeri primi.

- Computazionalmente parlando, questa operazione è estremamente costosa, al punto che fattorizzare un numero primo di grandi dimensioni può ad oggi dirsi impossibile (migliaia di anni).

# Reale forza della crittografia

---

- Numerose semplificazioni e nuovi modelli matematici hanno nel corso degli anni abbassato il costo computazionale della fattorizzazione, ma non c'è mai stato quel “salto di qualità” tale da rendere la decifratura forza bruta della cifratura a chiavi pubbliche tecnicamente fattibile.
- Però abbiamo una spada di Damocle sulla testa: (a patto che sia possibile) cosa succederà nel momento in cui un matematico scoprirà un modo realmente efficiente per fattorizzare un numero primo anche di grandi dimensioni?
- Tutta l'architettura a chiave pubblica attualmente in circolazione perderebbe immediatamente di valore.
- Non è naturalmente detto che sia possibile, ma la scienza fa passi sempre più lunghi...

# Reale forza della crittografia

---

- Una soluzione può venire dalla “crittografia a curve ellittiche”
- Si basa sui “logaritmi discreti”
- A fronte di un livello di sicurezza paragonabile a quello della crittografia a chiave pubblica più comune, consente l'utilizzo di chiavi estremamente più brevi:
  - 160 bit ECC = 1024 bit RSA
  - 210 bit ECC = 2048 bit RSA
  - 320 bit ECC = 5120 bit RSA
- Questo porta oltretutto ad una maggior efficienza dell'operazione di cifratura in se per se (più lunga è la chiave, più c'è da calcolare).
- Peccato che la ECC sia in parte coperta da brevetti.

# Reale forza della crittografia

---

- Il futuro, è la “crittografia quantistica”
- Grazie ai progressi della fisica nel campo della meccanica quantistica, è possibile monitorare il canale di trasmissione durante lo scambio della chiave, rilevando immediatamente qualsiasi tentativo di intrusione sul canale.
- Ad ogni buon conto, anche il modello crittografico della chiave pubblica, al momento attuale, deve fare i conti con l'incremento della potenza computazionale.
- Per poter ragionevolmente ritenere un testo cifrato inviolabile fino al 2030, dovremo utilizzare almeno chiavi da 2048 bit.

# Pigrizia nella ricerca delle soluzioni

---

- Spesso chi sviluppa un software si trova a dover far fronte a problemi di sicurezza.
- Si tratta di un problema comune e “normale”: tutti sbagliano.
- La soluzione dell'errore però, spesso risiede in una patch. Risolto il problema segnalato, si spera che il resto vada a posto da solo, senza rimettere in discussione la struttura stessa che ha portato all'errore.
- E' come se un ponte che vibra per effetto del vento fosse “messo in sicurezza” aggiungendo una ulteriore putrella di metallo. Con un vento debole non si muove piu, ma appena il vento aumenta siamo punto e a capo.

# Come si realizza la sicurezza

---

- Come si realizza allora il processo di sicurezza?
  - Prima di tutto tenendosi informati
    - Bugtraq
    - Nel momento in cui si scopre una vulnerabilità nel software che si sta utilizzando, lo si aggiorna (subito), tenendo ben chiaro come questo tassello va a modificare il sistema nel suo complesso
  - Eliminando tutti i servizi inutili e sostituendo quelli non sicuri
  - Riducendo l'esposizione (meno usato è...)
    - Piattaforme operative non standard (Windows/x86), e possibilmente decenti...
    - Utilizzando software alternativo (browser, posta, office...)
    - ...

# Come si realizza la sicurezza

---

- Come si realizza allora il processo di sicurezza?
  - Usando la testa quando si configurano i servizi ed i sistemi (andare al di là degli How-To)
    - Negare tutto, e specificare le eccezioni.
    - Evitare le configurazioni di default (supporto sshd v1)
    - Riutilizzare le informazioni che si ottengono tenendosi informati: sapendo quanto sicuro è Wu-FTPd, evitare di installarlo!
  - Lavorando costantemente per innalzare dove possibile il livello di sicurezza del sistema: avvengono quasi 150.000 attacchi ogni anno (di cui ne vengono registrati circa il 35%). Prima o poi capiterà anche a voi (paranoia!)

Fatevi trovare pronti! (e fate i backup!)