Appunti di Introduction to Computer and Network Security

Luca Facchini

Matricola: 245965

Corso tenuto dal prof. Raniese Silvio

Università degli Studi di Trento

A.A. 2024/2025

Sommario

Appunti del corso di Introduction to Computer and Network Security tenuto dal prof. Raniese Silvio presso l'Università degli Studi di Trento nell'anno accademico 2024/2025.

Indice

Capitolo 1

Basic Notions - Nozioni Base

1.1 The CIA Triad

1.1.1 Introduzione

L'acronimo L'acronimo CIA in inglese sta per **Confidentiality**, **Integrity** e **Availability**. Questi tre concetti sono alla base della sicurezza informatica e della protezione dei dati.

Confidentiality - Riservatezza Il principio della Riservatezza prescrive che i non autorizzati non devono poter accedere ai dati e i dati sono visibili solo a chi è autorizzato a vederli.

Integrity - Integrità Il principio di Integrità prescrive che i dati non devono essere modificati se non da persone autorizzate a farlo, i tentativi di modifica non autorizzati devono essere rilevati e prevenuti.

Availability - Disponibilità Il principio della Disponibilità prescrive che i dati non devono essere mantenuti inaccessibili ed inoltre bisogna permettere l'accesso agli autorizzati alle informazioni e ai servizi.

1.1.2 In Pratica

Confidentiality - Riservatezza L'accesso alle informazioni riservare può essere:

Intenzionale Quando un intruso cerca di accedere a dati sensibili.

Non Intenzionale Quando un utente autorizzato accede a dati sensibili senza volerlo causato dal fatto che chi detiene i dati non è competente e/o non gli importa della sicurezza.

Come garantire la riservatezza:

Crittografia La **crittografia** è una tecnica che permette di limitare l'accesso ai dati solo a chi possiede la chiave per decifrarli (e quindi è autorizzato).

Access Control Il **controllo degli accessi** è una parte integrante per mantenere la **riservatezza**, ciò andando a gestire quali utenti possono accedere a quali risorse.

Integrity - Integrità Per garantire l'integrità bisogna prevenire la modifica e/o la distruzione, inoltre bisogna assicurare informazioni autentiche e non invalidate.

Inoltre ulteriore prescrizione è quella di individuare anche le cosiddette "minute changes" ovvero le modifiche minime che possono essere fatte ai dati per alterarne il significato.

L'integrità può essere anche di tipo **data integrity** ovvero la garanzia che i dati siano corretti e completi, oppure di tipo **system integrity** ovvero la garanzia che il sistema sia protetto da attacchi e che funzioni correttamente.

L'itemize può essere compromessa da:

- errore umano
- attacchi come malware distruttivi e ransomware

Come garantire l'integrità:

Version Control Il **controllo delle versioni** assieme agli **audit trails** permettono ad una organizzazione di garantire la veridicità dei dati.

Compliance requirements I **requisiti di conformità** sono regole e normative che un'organizzazione deve seguire per garantire l'integrità dei dati.

Availability - Disponibilità Le violazioni alla disponibilità includono:

- Fallimento dell'infrastruttura
- Overload dell'infrastruttura
- Blackout
- Attacchi come DDoS

Come garantire la disponibilità:

Backup Bisogna sempre avere un piano di **backup** per mantenere disponibile in caso di disastro, attacco o altre minacce si verificassero.

Cloud Il **cloud** è una soluzione per garantire la disponibilità dei dati, in quanto i dati sono replicati su più server e quindi in caso di guasto di uno di essi i dati sono comunque disponibili.

1.2 Security Policy, Mechanism, Service

1.2.1 Security Policy

Definizione

Una **security policy** è un insieme di regole e requisiti stabilita da una azienda che definisce l'uso accettabile delle informazioni e dei servizi, inoltre il livello indica il livello di confidenzialità, integrità e disponibilità delle informazioni.

Esempi

Un esempio di possibili regole di una **security policy** potrebbero essere:

- I dipendenti devono completare il corso su sicurezza informatica e accettare la policy di sicurezza.
- I visitatori dell'azienda devono essere accompagnati da un dipendente autorizzato in tutti i momenti, questo dipendente deve mantenere i visitatori in una area appropriata senza dati sensibili
- I dipendenti devono mantenere una "scrivania pulita" dunque senza lasciare documenti sensibili accessibili e incustoditi.
- I dipendenti devono usare una password sicura e queste non devono essere usate in altri servizi esterni.
- Gli ex-dipendenti non devono mantenere alcun dato dell'azienda.

1.2.2 Security Service

Definizione

La capacità di supportare uno o più dei requisiti di sicurezza (Confidentiality, Integrity, Availability) e.s.: gestione chiavi, controllo accessi, autenticazione. Come già detto i **Security Services** implementano uno o più **Security Mechanism** che servono per far rispettare la **Security Policy**.

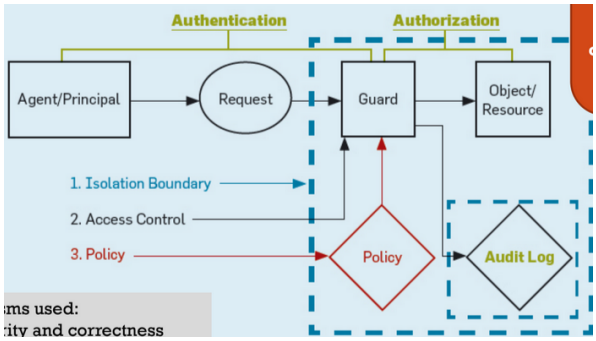
Esempi

HTTP(S)

HTTP Se si sua il protocolla **HTTP** per la trasmissione dei dati qualsiasi informazione trasmessa può venire intercettata da hacker nel mezzo.

HTTPS Per ovviare a questo problema si usa il protocollo **HTTPS** che è una versione sicura di **HTTP** che usa la crittografia per proteggere i dati, in questo modo i dati trasmessi vengono cifrati e se intercettati non possono essere letti.

Access Control



Audit Log è un registro importante che registra tutte le richieste di accesso ai dati e se la richiesta è stata accettata o rifiutata, in questo modo si può monitorare se una richiesta che è stata fatta ha avuto un risultato non aspettato e quindi che vadano modificate le **policy**.

1.2.3 Security Mechanism

Definizione

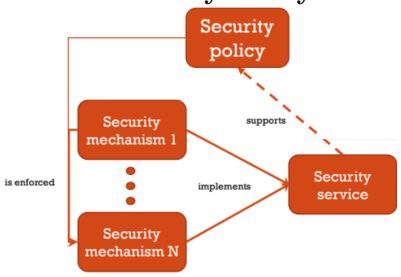
Un **security mechanism** un dispositivo o funzione designato per provvedere uno o più servizi di sicurezza classificate in termini di potenza

Inoltre è l'implementazione della **security policy**

Esempi

1.2.4 Come si relazionano

In conclusione i **Security Services** implementano uno o più **Security Mechanism** che servono per far rispettare la **Security Policy**.



1.3 CIA, Security Violation, and Mitigation

La triade CIA non è solo essenziale per rendere sicuri i dati ma aiuta anche a capire cosa è andato storto nel caso di una **security violation**

1.3.1 Attacco Ransomware

L'arttacco Nel caso di un attacco ransomware i dati vengono tenuti in ostaggio e la vittima viene ricattata con la pubblicazione di dati personali o il mancato accesso ad essi fino a che un riscatto non viene pagato.

Questo attacco cripta i dati rendendoli inaccessibili e per ripristinarli è necessaria una chiave di decodifica, i pagamenti solitamente vengono eseguiti in crypto valute poco tracciabili.

Questo genere di attacco viola la **confidentiality** e la **availability** dei dati.

Mitigation Per mitigare il rischio (che comunque non può essere eliminato) è utile procedere con un approccio **Zero Thrust** ovvero un "protocollo" che richiede che tutti gli utenti all'interno e all'esterno della rete aziendale siano autenticati e che la connessione è continuamente validata, prima di consentire accesso a applicazioni e dati.

1.4 Risk

Non esiste un sistema sicuro, esistono solo diversi livelli di insicurezza, si punta sempre ad ottenere il miglior livello di sicurezza possibile con il budget a disposizione.

1.4.1 Vulnerability

Definizione

Una **vulnerabilità** è una debolezza in un sistema informativo, un sistema di sicurezza, nei controlli interni o nell'implementazione di questi. Questa debolezza può essere sfruttata o compromessa da una minaccia alla sicurezza.

Examples

Un esempio di **vulnerabilità** è una backdoor nascosta nel software e/o hardware, oppure dei bug non conosciuti di un software, o anche delle password deboli.

1.4.2 Threat

Definizione

Qualunque circostanza o evento con il potenziale per causare perdite o danni a: operazioni dell'azienda, risorse o individui che possono accedere a tali risorse tramite un accesso non autorizzato, distruzione, pubblicazione, modifica e/o DoS. Inoltre un **threat** è un potenziale attacco che sfrutta una **vulnerabilità** del sistema.

Examples

Un esempio di **threat** sono:

- Hacker:
  - Trova e/o **sniff** una password.
  - Una il **social engineering** per ottenere informazioni sensibili e password.
  - Occupa le risorse di sistema con richieste inutili (attacchi DoS).
- Virus e worms:
  - I **virus** sono programmi auto replicanti che richiedono una azione dell'utente per essere attivati, esempio ne è una Email, un allegato o un link o un Floppy/CD/USB infetto.
  - I **worms** sono programmi auto replicanti che non richiedono una azione dell'utente per essere attivati, esempio ne è un worm che sfrutta una vulnerabilità di un sistema operativo.

1.4.3 Attack

Definizione

Un **attacco** è un qualsiasi attività dannosa che prova a raccogliere, degradare, negare o distruggere informazioni o servizi. Può anche manifestarsi sotto un tentativo di ottenere un accesso ad una risorsa alla quale non si ha diritto

1.4.4 Risk

Definizione

Il **rischio** è la **probabilità** che una particolare **minaccia sfrutti** una particolare **vulnerabilità**.Il **rischio** è inoltre una misura di quanto una particolare circostanza o evento possa essere potenzialmente minacciato ed è tipicamente in funzione di:

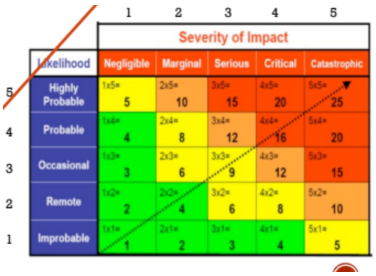
1. L'impatto che questa circostanza o evento potrebbe avere
2. La probabilità che questa circostanza o evento si verifichi

Riassumendo

Riassumendo le **minacce** sono una combinazione di **intento** e **capability** di un attacco con successo. Invece le **vulnerabilità** sono caratterizzate da quanto è facile **identificarle** e **sfruttarle**.

C'è da dire che l'impatto di un attacco deve essere valutato dal punto di vista di ogni soggetto coinvolto, ad esempio un attacco che compromette la privacy di un singolo individuo può avere un impatto molto diverso se lo si guarda dal punto di vista dell'individuo o dell'azienda.

Risk Matrix



La presente matrice è una rappresentazione grafica del rischio, in cui si valuta la probabilità di un attacco e l'impatto che questo avrebbe, solitamente un rischio per essere accettabile deve avere un valore risultante tra basso (1-4) e medio-basso (5-9), i valori quali medio-alto (10-14) e alto (15-25) sono considerati inaccettabili e quindi devono essere mitigati abbassando la probabilità di attacco e/o l'impatto che questo avrebbe.

1.5 Fine della diffusione degli attacchi

Generalmente Il rischio è causato dunque dalla somma di una minaccia e una vulnerabilità, la minaccia è la capacità di sfruttare una vulnerabilità per causare un danno.

Pianizifazione Quando si studiano i rischi sia prima che dopo un attacco bisogna considerare che se la **Minaccia - Threat** riesce a sfruttare la **Vulnerabilità - Vulnerability** allora si ha un **Rischio - Risk** se questo risulta inaccettabile allora si deve procedere con la **Mitigazione - Mitigation** del rischio, ciò riduce la **Minaccia - Threat** e il ciclo riprende fino a quando il **Rischio residuo - Residual Risk** è accettabile.

1.6 Security & Human Factor - Sicurezza & Fattore Umano

Il fattore umano è uno dei fattori più importanti nella sicurezza informatica, infatti la maggior parte degli attacchi sono causati da password poco sicure o da una mancata preparazione degli utenti.

1.6.1 Passwords

C'è chi dice che le password andrebbero trattate come la biancheria intima, ovvero:

- Cambiate regolarmente - magari senza usare un pattern

- Non condivise con nessuno - nemmeno con colleghi o familiari
- Non lasciate sulla scrivania - non scritte su post-it o in chiaro

1.6.2 Security

Ogni persona che ha accesso ad un sistema informatico è un potenziale punto di attacco, quindi è importante che ogni utente sia formato sulla sicurezza, ogni utente al suo livello a partire dall'utente base che necessita di una formazione base fino ad arrivare all amministratore di sistema che necessita di una formazione più avanzata.

Capitolo 2

Authentication I: Passwords & co

2.1 User Authentication & Digital Identity - Autenticazione e Identità Digitale

2.1.1 Introduzione e Definizioni

Identità è un insieme di attributi relazionati ad una entità

Attributo è una caratteristica o proprietà di una entità ch e può essere usata per descriverne lo stato, apparenza o ogni altro aspetto.

Identità digitale è ua identità i quali attributi sono conservati e trasmessi in forma digitale

Ma perché la identità digitale è importante? In pratica questa può essere una soluzione per diversi aspetti che i governi mondiali vorrebbero risolvere.

2.1.2 Digital Identity Lifecycle - Ciclo di Vita dell'Identità Digitale

Creazione L'identità viene creata, in genere da un ente di certificazione che assicura l'identità dell'utente e fornisce/chiede una forma di accesso per riconoscere in futuro l'utente. Viene creato un ID univoco per l'utente

Autenticazione L'identità viene autenticata tramite il metodo concordato in fase di creazione

Autorizzazione/ Accesso L'utente ha accesso a risorse e servizi in base ai permessi assegnati, i dati necessari dell'utente vengono trasmessi dall'ente di certificazione al servizio.

Cancellazione o Disattivazione L'identità viene cancellata o disattivata quando non è più necessaria e i dati devono essere cancellati dopo un certo periodo di tempo per garantire la privacy dell'utente.

Enrollment / On-Boarding - Creazione

La fase di creazione di una utenza porta un **Applicant** a diventare un **Subscriber** tramite una serie di passaggi:

Resolution Vengono raccolti gli **attributi** essenziali dell'utente (nome, indirizzo, data e luogo di nascita, ...) vengono ora raccolti anche due documenti di identità (patente, passaporto, C.I.,...) - ora l'utente viene distinto univocamente su un contesto

Validation Gli **attributi** raccolti vengono validati sulla base di **prove** tramite una fonte autorevole- gli attributi vengono ora associati ad una persona fisica

Verification Le **prove** vengono verificate, in questo punto si verifica la corrispondenza tra le diverse foto, viene mandato un codice ai contatti per verificare che siano i suoi, etc - l'utente viene ora confermato e la sua identità è certificata

Identity Assurance Levels - Livelli di Assicurazione dell'Identità

IAL1 GLi attributi, se presenti, vengono auto-dichiarati, o considerati come tali

IAL2 Una persona o di remoto o in presenza, verifica gli attributi

IAL3 Gli attributi vengono verificati da una persona autorizzata e i documenti fisici vengono esaminati

Authentication - Autenticazione

Definzione L'**autenticazione** è il processo di verifica dell'identità di un utente, processo o dispositivo.
Il **richiedente** deve dimostrare al **verificante** che è chi dice di essere.

Base Quando esegui il login solitamente inserisci un **username** e una **password** che vengono confrontati con quelli memorizzati nel sistema.
L'autenticazione tramite **password** è molto diffusa e semplice da implementare.

2.2 An Introduction to Passwords - Introduzione alle Password

2.2.1 User Authn - Autenticazione Utente

Quando esegui il login inserisci un **username** per annunciare chi sei e una **password** per dimostrare chi dici di essere, questo tipo di autenticazione si chiama **user authn** ovvero il processo di verifica dell'identità di un utente.

Entollment - Creazione

Le **password** dovrebbero essere conosciute solo dall'utente e dal sistema. Spesso però, soprattutto nelle grandi aziende, le password vengono mandata via mail o scritte dall'utente su una pagina web. In questi casi da chi potrebbe essere intercettata la password?

La teoria rispetto alla realtà

Le password hanno dominato il mondo dell'informatica sin dall'avvento dei computer, ma da allora la richiesta verso queste è che siano più sicure ma allo stesso tempo più "User Friendly", siamo in ricerca di alternative ma per ora delle proposte alternative come *criteri di complessità* ma studi hanno rilevato che spesso non vengono rispettati e/o che non sono efficaci.

2.2.2 Password Security - Sicurezza delle Password

All'inizio di internet le password venivano conservate in file appositi in chiaro, ma questo comportava un rischio molto alto, infatti se un attaccante riusciva ad accedere a quel file poteva facilmente accedere a tutte le password.

Hashing

Per risolvere questo problema si è pensato di "**hash-are**" le password, ovvero di applicare una funzione di hash alla password e conservare il risultato. In questo modo se un attaccante accede al file delle password non può risalire alla password originale.

Proprietà della funzione di hash Una funzione di hash ottimale dovrebbe avere le seguenti proprietà:

Deterministica dato un input la funzione restituisce sempre lo stesso output

Rapida la funzione deve essere veloce da calcolare

Difficile da invertire data l'output è difficile risalire all'input

Biettiva due input diversi devono avere output diversi

Lunghezza fissa l'output deve avere una lunghezza fissa indipendentemente dalla lunghezza dell'input

Molto spesso però capita che i bit disponibili per l'hash siano limitati rispetto a tutti i possibili input, per questo motivo si è cercato di creare funzioni di hash che generano due output uguali solo in rarissimi casi, inoltre si aggiunge a questo che per input simili l'output sia molto diverso.

Crack of Passwords

In caso di un "leak" di password che sono state opportunamente codificate con una funzione di hash, un attaccante può tentare di decifrarle ma questo è molto difficile. Questo diventa più semplice se vengono usate password poco sicure e/o comuni. infatti si stimi che per trovare una password da 8 caratteri composti da lettere minuscole e numeri ci vogliano "solo" 155€ su una istanza di AWS.

2.2.3 Hash & Salt

Per rendere più difficile il lavoro degli attaccanti si è pensato di aggiungere un **salt** alle password, ovvero un valore casuale che viene aggiunto alla password prima di calcolare l'hash. Questo rende più difficile per l'attaccante decifrare la password.

2.3 Multi Factor Authentication - Autenticazione a più fattori

2.3.1 Definizione

L'autenticazione a più fattori è un metodo di autenticazione che richiede l'uso di più metodi di autenticazione per verificare l'identità di un utente.

2.3.2 Fattori di Autenticazione

I fattori di autenticazione sono:

Qualcosa che sai (password, PIN, ...)

Qualcosa che hai (smartphone, token, ...)

Qualcosa che sei (impronte digitali, riconoscimento facciale, ...)

Vantaggi/Svantaggi quello che sai

Vantaggi

- Facile da implementare - non richiede hardware aggiuntivo
- Facile da usare - Basta ricordare la password
- Facile da resettare se dimenticato - basta fare il reset della password

Svantaggi

- Facile da rubare - se un attaccante riesce a scoprire la password può accedere al sistema
- Facile da dimenticare - se la password è complessa è facile dimenticarla
- Facile da indovinare - se la password è semplice ed è stata usata in altri contesti è facile indovinarla

Vantaggi/Svantaggi quello che hai

Vantaggi

- Difficile da rubare - se un attaccante non ha il dispositivo non può accedere al sistema
- Difficile da indovinare - se il dispositivo è protetto da PIN o password è difficile indovinare l'accesso
- Difficile da clonare - se il dispositivo è protetto da un token è difficile clonarlo inoltre la parte di autenticazione è fatta dal dispositivo stesso e non dal sistema

Svantaggi

- Facile da perdere - se il dispositivo viene perso l'utente non può accedere al sistema
- Difficile da resettare - se il dispositivo viene perso l'utente deve contattare l'amministratore per resettare l'accesso
- Costoso - i dispositivi possono essere costosi

Vantaggi/Svantaggi quello che sei

Vantaggi

- Difficile da rubare - le impronte digitali o il riconoscimento facciale sono unici e difficili da rubare
- Difficile da indovinare - è difficile indovinare le impronte digitali o il riconoscimento facciale
- Difficile da clonare - è difficile clonare le impronte digitali o il riconoscimento facciale

Svantaggi

- Se il fattore viene compromesso non può essere cambiato - se le impronte digitali vengono rubate non possono essere cambiate
- Costoso - i dispositivi possono essere costosi
- Non sempre preciso - il riconoscimento facciale può essere ingannato

PSD2 Compliance

Cos'è PSD2 La **PSD2** è una direttiva europea che regola i servizi di pagamento e che richiede l'autenticazione a più fattori per i pagamenti online. Questo per ridurre il rischio di frodi.

How to comply MFA in PSD2 Una idea sarebbe quella di includere nella challenge anche i dati della particolare transazione come:

- L'identificativo del destinatario della transazione
- L'importo della transazione
- L'instante nella quale la transazione è stata inizializzata
- ...

Purtroppo però ciò non è abbastanza in quanto i dati sopra indicati possono essere ricavati dal contesto il che rende la challenge prevedibile.

2.3.3 FIDO: Phishing Resistant Authentication

Cos'è FIDO

FIDO (Fast Identity Online) è un insieme di standard aperti per l'autenticazione a più fattori che mira a ridurre il rischio di phishing. Il suo scopo è quello di assicurare una **autenticazione forte** e di **ridurre l'uso di password**.

Come funziona

1. Viene chiesto all'utente di scegliere un ente FIDO
2. L'utenti sblocca il dispositivo FIDO usando un'impronta digitale, un pulsante su un dispositivo di secondo fattore, un PIN o un qualsiasi altro metodo di autenticazione supportato
3. Il dispositivo crea una chiave pubblica e una chiave privata univoca per il dispositivo, il servizio online e l'utente
4. La chiave pubblica è inviata al servizio online ed associata all'account dell'utente
5. Il servizio online chiede all'utente di autenticarsi con il dispositivo precedentemente registrato
6. L'utente sblocca il dispositivo FIDO usando lo stesso metodo di autenticazione usato in precedenza
7. Il dispositivo FIDO usa l'account dell'utente identificato per inviare la corretta chiave al servizio online
8. Infine il dispositivo invia la challenge ricevuta dal servizio online firmata con la chiave privata e il servizio online verifica la firma con la chiave pubblica, l'utente è autenticato.

2.4 Outsourcing Authentication

2.4.1 Definizione

L'outsourcing dell'autenticazione è il processo di affidare l'autenticazione degli utenti ad un servizio esterno, spesso chiamato **Identity Provider**. Questo servizio si occupa di verificare l'identità dell'utente e di inviare un token di autenticazione al servizio che richiede l'autenticazione. Esempio in italia per l'accesso ai servizi pubblici si usa SPID.

2.4.2 Che problema risolve

L'outsourcing dell'autenticazione risolve diversi problemi:

Sicurezza L'Identity Provider è specializzato in autenticazione e può offrire un livello di sicurezza più alto

User Experience L'Identity Provider può offrire un'esperienza utente migliore in quanto l'utente non deve ricordare le password per ogni servizio ma solo quella dell'Identity Provider - SSO

Compliance L'Identity Provider può aiutare a rispettare le normative sulla privacy e la sicurezza

2.4.3 La soluzione

L'outsourcing dell'autenticazione è delegato ad una terza parte, l'**Identity Provider**, che si occupa di verificare l'identità dell'utente e di inviare le informazioni di autenticazione al servizio che richiede l'autenticazione. A questo punto il servizio invia all'utente un token di autenticazione che può essere usato per accedere al servizio senza dover inserire nuovamente le credenziali del SSO.

2.4.4 Potenziali Problemi

L'outsourcing dell'autenticazione può comportare alcuni problemi:

Single Point of Failure Se l'Identity Provider è compromesso tutti i servizi che usano l'Identity Provider per l'autenticazione sono compromessi

Privacy L'Identity Provider può raccogliere informazioni sull'utente e sulle sue abitudini di navigazione

Capitolo 3

Cryptography Introduction

3.1 Cryptosystem

Definizione Un **cryptosystem** è una tupla di 5 elementi (E,D,M,K,C):

E è un *algoritmo di cifratura*

D è un *algoritmo di de-cifratura*

M è un insieme di *messaggi in chiaro*

K è un insieme di *chiavi*

C è un insieme di *messaggi cifrati*

In modo astratto i punti **E** e **D** possono essere espresse come funzioni:

$$\begin{aligned} E &: M \times K \to C \\ D &: C \times K \to M \end{aligned} \qquad D(E(m,k),k) = m$$

Nella criptografia base ogni chiave $k \in K$ può essere usata per cifrare e de-cifrare i messaggi, nella criptografia simmetrica la chiave è la stessa per entrambi i processi, mentre nella criptografia asimmetrica le chiavi sono diverse.

Quali componenti sono pubbliche? Solitamente non abbiamo necessità di mantenere segrete le funzioni `E` e `D` o il messaggio cifrato `C`, quello che davvero deve essere segreto è la chiave `K`. Il motivo per il quale le funzioni non devono essere segrete è che il sistema deve essere pubblico e quindi tramite processi di *reverse engineering* è possibile ricavare le funzioni (es: il sistema di cifratura di un DVD è stato ricavato tramite *reverse engineering* dopo solo due giorni).

3.1.1 Why Cryptography?

Sicurezza La crittografia è usata nei **meccanismi di sicurezza** per garantire **confidenzialità** nascondendo il contenuto dei messaggi, **integrità** provvedendo al controllo di integrità tramite funzioni di **hash** e **la verifica dell'origine** dei dati tramite firme digitali verificabili da una fonte autorevole.

3.1.2 Cryptography on rented servers

Problema Se si usa un server di terze parti per conservare dati, ad esempio su un database, è possibile che il proprietario del server possa accedere ai dati in chiaro. Per evitare ciò si può cifrare i dati prima di inviarli al server, in modo che il proprietario non possa leggerli, ciò comporta però un aumento del carico computazionale lato client in quanto tutti i dati prima di essere letti devono essere de-cifrati, per questo esistono meccanismi di ricerca su dati cifrati che permettono di effettuare ricerche su dati cifrati senza de-cifrarli, una volta trovati i dati desiderati si de-cifrano solo quelli.

3.1.3 Come definiamo "Computazionalmente sicuro" nelle comunicazioni

Definzizione Definiamo un sistema di comunicazione **Computazionalmente sicuro** quando la decifrazione di un messaggio cifrato senza conoscere la chiave è molto difficile, o addirittura impossibile, e richiede molto tempo e risorse computazionali.

$E(k, P) = C$ Calcolare C da P deve essere difficile senza k, inoltre calcolare C da P sapendo k deve essere facile.

$D(k, C) = P$ Calcolare P da C deve essere facile sapendo k, ma deve essere difficile senza k.

Trapdoor - funzione a senso unico Una **trapdoor** è una funzione a senso unico che richiede una ulteriore informazione. Ed una **funzione a senso unico** è descritta come una funzione che è facile da calcolare in una direzione, ma molto più difficile nell'altra.

3.1.4 Hash v/s Encryption

Quando l'una e quando l'altra Usiamo funzioni di hash quando non abbiamo bisogno di accedere all'informazione originale, ma solo di verificare l'integrità dei dati. Ricordiamo che le funzioni di **hash** per definizione sono **one-way** e **deterministiche**, quindi non possiamo de-cifrarle e non necessitiamo di una chiave per cifrare i dati.
D'altro canto usiamo la cifratura quando i dati che vogliamo proteggere devono poter essere letti e ne vogliamo preservare la **confidenzialità**, bisogna prima concordare una **chiave** in maniera sicura, poi calcolare il **messaggio cifrato** con suddetta chiave e infine inviare il messaggio cifrato, il destinatario potrà de-cifrare il messaggio con la chiave concordata. In questo modo proteggiamo la **confidenzialità** e la **riservatezza** dei dati ma non la loro **integrità**.

3.1.5 La Criptografia non è la soluzione a tutti i problemi

Perchè? La crittografia non è la soluzione a tutti i problemi di sicurezza, in quanto comunque è sensibile a delle chiavi conservate su supporti digitali, i quali devono proteggere questa informazione. Inoltre da sola la crittografia non è mai usata come soluzione a problemi di sicurezza, ma sempre in combinazione con altri meccanismi di sicurezza.

3.2 Types of cryptography - Tipologie di criptografia

3.2.1 Cifrari di sostituzione

In breve I **cifrari di sostituzione** sono cifrari che sostituiscono un simbolo del *dizionario* con un altro simbolo. In questo contesto la *chiave* è la *sostituzione* dei simboli.

In lungo I **cifrari di sostituzione** sono dei **metodi di criptazione** per i quali ogni simbolo del **messaggio in chiaro** è rimpiazzato nel **messaggio cifrato** rispetto ad un prefissato sistema. I "simboli" possono essere lettere, coppie di lettere, triple di lettere, o anche una combinazione di questi, e anche altro. Il ricevente decifra il testo svolgendo le sostituzioni inverse.

Cifrario di cesare

Il **cifrario di Cesare** è un cifrario di sostituzione in cui il **dizionario** del testo in chiaro è spostato di un numero fisso di posizioni nell'alfabeto. Per decifrare il testo cifrato si sposta indietro dello stesso numero di posizioni.
Questo tipo di sistema viene anche chiamato: **rotazione di k posizioni**, in quanto il dizionario viene ruotato di k posizioni.

Problemi e proprietà Il cifrario descritto è il più semplice tra tutti come decodifica in quanto esistono solo 25 chiavi e con un semplice attacco di **brute force**. Inoltre se non si vuole usare un attacco di **brute force** si può comunque cercare di trovare lettere comuni del messaggio cifrato e provare a ricondurle a lettere comuni dell'alfabeto.

Figura 3.1: Esempio di cifrario di Cesare spostato di 3 posizioni

Cifrario di Vigenère

Il **cifrario di Vigenère** è un cifrario che prevede l'uso di una "parola chiave", se necessario ripetuta per la lunghezza del messaggio, per poi calcolare la "somma" dei valori di ogni lettera del messaggio con la corrispondente lettera della parola chiave (prendendo poi il resto della divisione per 26). Quindi assegnando ad ogni lettera un numero da 0 a 25, si somma il numero della lettera del messaggio con il numero della lettera della parola chiave, si prende il resto della divisione per 26 e si assegna alla lettera corrispondente il numero ottenuto.

A	T	T	A	C	K	A	T	D	A	W	N	plaintext
L	E	M	O	N	L	E	M	O	N	L	E	keyword repeated to match plaintext length
L	X	F	O	P	V	E	F	R	N	H	R	ciphertext

Figura 3.2: Esempio di cifrario di Vigenère chiave "lemon"

Per decifrare il messaggio cifrato si sottrae il numero della lettera della parola chiave al numero della lettera del messaggio e si prende il resto della divisione per 26.

Vantaggi Questo cifrario è molto più sicuro rispetto a quello di cesare in quanto come mostrato nell'esempio la lettera "A" viene codificata in quattro lettere diverse e la lettera "T" che è presente tre volte viene cifrata in due lettere corrispondenti. In quanto la parola chiave ha lunghezza "l" allora dimensione della chiave è l^{26}. Inoltre se la chiave è della stessa lunghezza del messaggio allora il cifrario di Vigenère è equivalente a un cifrario di sostituzione casuale il che rende impossibile la decifrazione, però ciò comporta ad una ulteriore difficoltà per il mittente e il destinatario nel concordare una chiave.

3.2.2 Cifrari di trasposizione

In breve I **cifrari di trasposizione** sono cifrari che permutano i simboli del messaggio in chiaro. In questo contesto la *chiave* è la *permutazione* dei simboli.

In lungo I **cifrari di trasposizione** sono dei **metodi di criptazione** per i quali i simboli del **messaggio in chiaro** sono permutati in un certo modo per ottenere il **messaggio cifrato**. Il ricevente decifra il testo svolgendo le permutazioni inverse.

Cifrario di trasposizione per colonne

Il **cifrario di trasposizione per colonne** è un cifrario che permuta i simboli del messaggio in chiaro per colonne, in modo che il messaggio cifrato sia una matrice di colonne.

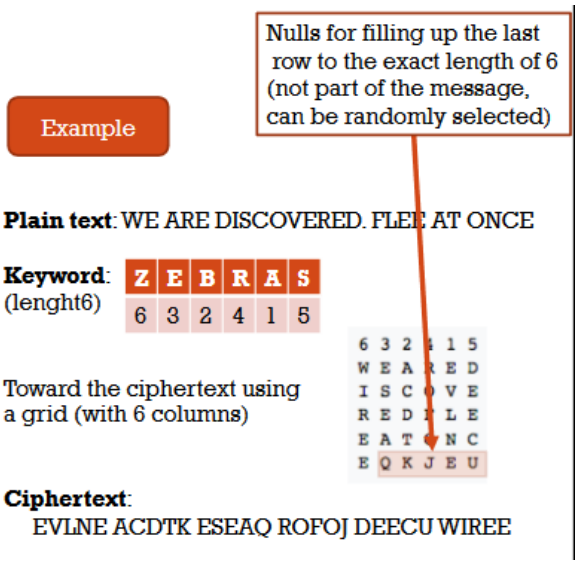


Figura 3.3: Esempio di cifrario di trasposizione per colonne

Per decifrare il messaggio cifrato si riordinano le colonne in base alla chiave.

3.2.3 Criptografia Simmetrica

Come detto in precedenza la **criptografia simmetrica** prevede la stessa **chiave** per cifrare e de-cifrare i messaggi. La gestione di chi possiede le chiavi determina chi può accedere ai dati.

Tipi Esistono nella *criptografia moderna* due tipi ci crittografia simmetrica:

Cifrari a flusso I **cifrari a flusso** cifrano una "breve" porzione di un blocco di dati alla volta con una chiave che varia nel tempo. Questa tipologia si base su un **generatore di chiavi**, la cifratura è relativamente semplice (es. `XOR`) e molto spesso viene usato un bit per blocco.

Cifrari a blocchi I **cifrari a blocchi** cifrano un blocco di dati "lungo" alla volta con una chiave fissa. Questa tipologia è più complessa e richiede uan funzione di cifratura e una di de-cifratura. Solitamente si usano blocchi di 64/128 bit.

Cifrari a flusso

I cifrari a flusso operano su un flusso di *testo in chiaro* e producono un flusso di *testo cifrato*. Lo stesso testo se ripetuto può essere cifrato in modo diverso in base al tempo nel quale è stato cifrato. I circuiti sono molto semplici e progettati per essere molto veloci.

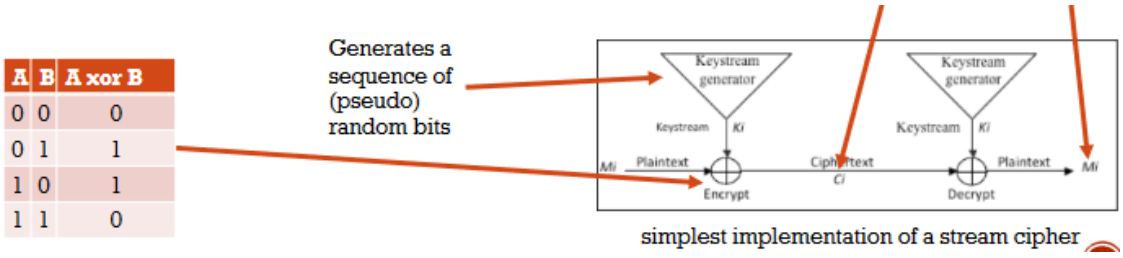


Figura 3.4: Esempio di circuito di cifratura a flusso

Dove si usano I cifrari a flusso sono usati in applicazioni in cui la velocità è fondamentale, come ad esempio nelle comunicazioni via radio o nella comunicazione in tempo reale.

Cifrari a blocchi

Nei **cifrari a blocchi** vene trattato una sezione di *testo in chiaro* come una cosa unica per produrre un *testo cifrato* di egual lunghezza. La dimensione tipica di questa sezione è solitamente tra 64 e 128 bit. Un messaggio M più lungo di n bit viene diviso in blocchi di n bit e ogni blocco $(M_1, M_2, \ldots, M_i)$ viene cifrato separatamente con la stessa chiave k.

Esempi Alcuni esempi di cifrari a blocchi sono:

DES Data Encryption Standard è un cifrario a blocchi che opera su blocchi di 64 bit.

AES Advanced Encryption Standard è un cifrario a blocchi che opera su blocchi di 128 bit e utilizza chiavi di 128, 192 o 256 bit.

DES Il **Data Encryption Standard** è un cifrario a blocchi che opera su blocchi di 64 bit che venne progettato da `IBM` negli anni '70. Il cifrario è basato su una rete di sostituzione e trasposizione, e utilizza una chiave di 56 bit. Il DES è stato il cifrario più usato fino al 2000 (anche da ANS[1] e in tre F.I.P.S.[2]), fino al 2005 quando è stato definitivamente dichiarato insicuro.

Feistel (1973) Il cifrario DES è basato su una struttura di **Feistel**, la quale prevede che il testo in chiaro venga diviso in due parti uguali, la prima passa attraverso una funzione di cifratura insieme ad una chiave, il risultato viene messo in **XOR** con la seconda parte del testo in chiaro, il risultato viene poi scambiato con la prima parte e il processo viene ripetuto per 16 volte con 16 chiavi diverse (o una chiave divisa in 16 parti). La decodifica avviene tramite lo stesso processo ma con le chiavi e le parti invertite.

Vantaggi e svantaggi Primo vantaggio di questo algoritmo è lo stesso meccanismo di cifratura e de-cifratura oltre alla semplicità, questo rende le componenti hardware molto più semplici e meno costose in quanto non necessitano di componenti diverse per cifrare e de-cifrare.

AES L'**Advanced Encryption Standard** è un cifrario a blocchi sviluppato come successore del `DES` dal dicembre 2001. L'`AES` usa una chiave simmetrica in uno schema chiamato **Rijndael**.
Il processo di cifratura è basato su una serie di tabelle che contengono le operazioni da eseguire e di operazioni `XOR` che sono molto veloci e semplici conoscendo la chiave, l'operazione di de-cifratura è la stessa ma con le tabelle invertite. Tuttavia anche se il processo è lo stesso bisogna costruire hardware e software diversi per cifrare e de-cifrare.

3.2.4 Criptografia Asimmetrica

La **criptografia asimmetrica** anche chiamata ***Crittografia a chiave pubblica*** `PKC` prevede l'uso di due chiavi diverse, una per cifrare e una per de-cifrare. Questo sistema viene usato per avere una comunicazione sicura tra due parti in un contesto non sicuro.

One way function Le **funzioni one way** sono funzioni matematiche che costituiscono la base della crittografia asimmetrica, queste funzioni sono facili da calcolare in una direzione ma molto difficili nell'altra. Nella crittografia asimmetrica tuttavia usiamo funzioni *one-way* che contengono una così detta *trapdoor* o *backdoor* che permette di invertire la funzione in modo semplice se si conosce un certo parametro.

Caso dell'`RSA` Nell'`RSA` viene usata una funzione matematica che è facile da calcolare in una direzione ma molto difficile nell'altra, ovvero la **fattorizzazione di numeri primi**.

[1] American National Standard X3.92
[2] Federal Information Processing Standards

Le chiavi Le due chiavi usate nella crittografia asimmetrica sono due, una per cifrare e una per de-cifrare, la chiave pubblica è usata per cifrare i messaggi e la chiave privata è usata per de-cifrarli. La chiave pubblica è distribuita a tutti. Queste chiavi sono diverse ma corelate matematicamente anche se la conoscenza di una non permette la conoscenza dell'altra.

`RSA` - *(Rivest, Shamir, Adleman)*

L'`RSA` è un algoritmo di crittografia asimmetrica più usato al mondo, è basato sulla difficoltà di fattorizzare numeri primi molto grandi. L'algoritmo è stato inventato nel 1977 da `Ron Rivest`, `Adi Shamir` e `Leonard Adleman`. Oggi è il sistema di crittografia più usato al mondo e viene usato per scambio di chiavi, firme digitali e per la crittografia di dati.

Funzionamento Si prendono due numeri p e q entrambi primi molto grandi [3], poi si calcola il prodotto $N = p \cdot q$ questo sarà il **modulo**. Ora si prende un qualsiasi numero e tale che questo sia primo rispetto a $(p-1)(q-1)$, con questo troviamo d definito come l'inverso del modulo e come $d \cdot e = 1 \mod (p-1)(q-1)$. La chiave pubblica sarà (N, e) e la chiave privata sarà (N, d). Assumendo quindi che M con $0 < M < N$ sia il messaggio da cifrare, il messaggio cifrato sarà $C = M^e \mod N$ e il messaggio de-cifrato sarà $M = C^d \mod N$. Notare come $C^d \mod N = (M^e \mod N)^d \mod N = M^{ed} \mod N = M$.

Attacchi In quanto N e e sono pubblici se un attaccante riesce a fattorizzare N allora può usarlo per calcolarsi d e quindi de-cifrare i messaggi in quanto $e \cdot d = 1 \mod (p-1)(q-1)$. Per questo motivo è importante che N sia molto grande e che p e q siano molto grandi.

Un problema recente con `RSA` Recentemente è stato scoperto un problema relativo alla generazioni di chiavi `RSA` infatti una libreria software usata nelle *smartcard*, *security tokens* e *trusted platform modules* per generare chiavi `RSA` presentava una vulnerabilità: la libreria non generava in modo casuale le chiavi ma usava un generatore di numeri pseudo-casuali che permetteva di conoscere la chiave privata conoscendo la chiave pubblica. Questo problema è stato riscontrato in molti dispositivi che sono stati prodotti con questa libreria, questi dispositivi sono stati ritirati dal mercato e sostituiti.

Integrità con `RSA` Per verificare l'integrità di un messaggio si può usare la crittografia `RSA` a proprio vantaggio. Prendiamo un messaggio, ce ne calcoliamo l'*hash* tramite una funzione di *hashing*, ora grazie alla chiave privata il mittente cifra il *digest* ottenuto e invia messaggio e *digest* cifrato al destinatario. Il destinatario de-cifra il *digest* grazie alla chiave pubblica del mittente e confronta e ri-calcola l'*hash* del messaggio, se i due *digest* coincidono allora il messaggio è integro, altrimenti se non coincidono il messaggio è stato alterato (o il messaggio o il *digest*).

Problemi dell'`RSA` In generale l'unico reale problema dell'`RSA` è la lentezza, infatti la generazione delle chiavi è molto lenta e la cifratura e de-cifratura sono molto lente, per questo motivo l'`RSA` viene usato solo per scambiare chiavi simmetriche che verranno poi usate per cifrare i messaggi. In questo modo si combina la sicurezza dell'`RSA` con la velocità dei cifrari simmetrici.

store now decrypt later Peccato che in questo modo si è esposti ad attacchi del genere *store now decrypt later* ovvero se un attaccante riesce a intercettare tutti i messaggi compreso lo scambio di una chiave allora quando sarà riuscito a de-cifrare il messaggio con la chiave simmetrica potrà de-cifrare tutti i messaggi scambiati.

Pubblicazione della chiave privata Un altro modo per attaccare un sistema crittografico basato su `RSA` è quello che per un qualunque motivo la chiave privata venga pubblicata, in questo caso l'attaccante può de-cifrare tutti i messaggi cifrati con la chiave pubblica, e se la chiave pubblica è usata per firmare i messaggi allora l'attaccante può falsificare i messaggi. In questa situazione la coppia di chiavi deve essere revocata e sostituita con una nuova coppia.

[3] generatore di numeri casuali e controllo che questi siano primi

`DH` - ***Diffie-Hellman***

La cifratura `DH`è un algoritmo crittografico che permette a due parti di scambiarsi una chiave senza che questa venga trasmessa in alcun modo tramite un canale (sicuro o meno). L'algoritmo si basa sulla complessità dell'inversione di esponenti modulari. L'algoritmo è stato inventato da `Whitfield Diffie` e `Martin Hellman` nel 1976.

Fondamenti matematici L'algoritmo si basa su un campo finito $F = GF(p)$ che sono tutti i numeri interi modulo p con p primo. Inoltre sfrutta le proprietà degli esponenti, in quanto $g^x(\mod p)$ è una funzione irreversibile ma semplice da calcolare. Infine viene usato il **Problema del logaritmo discreto** o *DLP* che descrive come dati p, g, X trovare x tale che $g^x \mod p = X$, il che è computazionalmente difficile se p è primo e g è un generatore per $0 < X < p$ esiste un x tale che $g^x \mod p = X$.

Funzionamento nel dettaglio Consideriamo `A` e `B` i nostri interlocutori allora il procedimento è il seguente:

1. `A` sceglie un numero primo p e un generatore g per $GF(p)$ e un numero segreto a.
2. `A` sceglie un numero a grande (chiave privata) e calcola $A = g^a \mod p$ (chiave pubblica).
3. Ora `A` invia p numero primo scelto, g generatore scelto e A chiave pubblica di `A` a `B`.
4. `B` ricevuto p, g e A sceglie un numero segreto b molto grande (chiave privata) e calcola $B = g^b \mod p$ (chiave pubblica).
5. `B` invia a `A` B chiave pubblica di `B`.
6. Entrambi ora si calcolano la chiave risultante, `A` calcolerà $K_a = B^a \mod p$ e `B` calcolerà $K_b = A^b \mod p$. Si può dimostrare che $K_a = K_b$.[4]

Sicurezza del protocollo La sicurezza di `DH` dipende dalla difficoltà del `DLP`[5] e dall'abilità di un attaccante di risolvere il `DLP`, altrimenti non è possibile ricavarsi le chiavi private da quelle pubbliche.

men-in-the-middle D'altra parte il protocollo `DH` non garantisce l'autenticità del messaggio, dunque quando dell'esempio precedente `B` riceve la chiave pubblica, il numero primo e il generatore, non sà se effettivamente lo ha ricevuto da `A` allora una terza parte si può mettere nel mezzo e inviare diverse informazioni a `A` e `B` e quindi la comunicazione tra `A` e `B` intercettata da `C` usa due chiavi diverse tra `A-C` e tra `C-B` il tutto mantenendo sicure le corrispondenti chiavi private. Per risolvere questo problema si può usare la crittografia asimmetrica per autenticare le chiavi pubbliche.

3.3 Riassumendo

- Bisogna evitare di crearsi il proprio algoritmo di crittografia, in quanto è molto difficile creare un algoritmo sicuro e quelli fatti in casa sono molto più vulnerabili.
- *DES* non deve essere più usato in quanto è stato dichiarato insicuro.
- Consigliati blocchi di chiavi da 80/90 bit per la crittografia simmetrica usando `AES`.
- La sicurezza di `DES` e `AES` non è provata, è solo resistente ad attacchi conosciuti.
- Non dimenticarti di gestire bene le chiavi:

  come condividerla? vedi dopo

  come proteggerla? Usa un sistema di controllo degli accessi

  Con n partecipanti servono n^2 chiavi

[4] $K = A^b \mod p = (g^a \mod p)^b \mod p = g^{ab} \mod p = (g^b \mod p)^a \mod p = B^a \mod p$
[5] *Discrete Logarithm Problem*

- PKC≠RSA in quanto alcune proprietà di RSA non usano nessun aspetto di PKC.
- La sicurezza effettiva dipende dallo stato dell'arte nel risolvere problemi matematici. (avanzamento tecnologico, disponibilità di risorse, ecc. . .)
- DH usato solo per scambio di chiavi ma persistono problemi di autenticità. (Attenzione agli attacchi MITM)