Riassunto Pugliese

Tommaso Puccetti
Studente presso Universita degli studi di Firenze

October 25, 2018

Contents

List of Tables

List of Figures

1 Overview

1.1 Computer Security Concepts

Computer security : *La protezione fornita ad un sistema informatico automatizzato con lo scopo di raggiungere gli obiettivi applicabili di preservare* ***integrity***, ***availability***, ***confidentiality*** , *delle risorse del sistema (include hardware, software, firmware, informazioni/dati, telecomunicazioni).* (NIST Computer Security Handbook 1995)

1.1.1 CIA triad: tre obiettivi per la sicurezza

- **Confidentiality** questo termine copre due concetti a esso correlati
    1. **Data confidecntiality**: Assicura che informazioni confidenziali non siano rilasciate ad individui non autorizzati.
    2. **Privacy**: Assicura all'individuo la possibilita di controllare od influenzare quali informazioni a loro relative possono essere raccolte e salvate, a chi e permesso raccoglierle e a chi potrebbero essere rilasciate.
- **Integrity**:
    1. **Data integrity**: assicurare che le informazioni e i programmi sono modificati solo in modo specificato ed autorizzato
    2. **System integrity**: assicurare che il sistema esegue le sue operazioni in modo non corrotto, senza che vi siano manipolazioni non autorizzate.
- **Availability**: Assicura che il sistema fornisce i suoi servizi prontamente, senza che il loro accesso sia negato ad utenti non autorizzati. (In QAS percentuale di tempo in cui il sistema e funzionante rispetto alla vita operazionale totale del sistema.)

Si hanno ulteriori concetti complementari a quelli sopra elencati:

- **Authenticity**: assicura che un'entita o un oggetto genuino e fidato. Deve esserci la possibili di essere verificato. *Supporta la confidence riguardo alla validit di una trasmissione, di un messaggio o del suo creatore*

- **Accountability**: Assicura che le azioni di una particolare entit siano attribuibili unicamente a quell'entit.*Supporta nonrepudiation, intrusion detection, prevention, fault isolation ecc.*

1.1.2 Livelli di impatto riguardo a breccie nella sicurezza

- **Low**: la perdita di sicurezza informazioni ha un impatto limitato (una degradazione rispetto alla mission del sistema, danno minimo, minima perdita finanziaria, minaccia)

- **Moderate**: la perdita a effetti seri (significativa degrado rispetto alla missione del sistema, minaccia significativa verso individui senza che sia a rischio la vita o siano considerabili infortunei che mettono a rischio la vita di un individuo.)

- **High**: la perdita di sicurezza ha severi o catastrofici effetti collaterali riguardo le operazioni, asset organizzativi o individui (perdita della vita).

Esempio Confidentiality:
La confidentiality dei voti ottenuti da uno studente considerata essere **molto importante**. Per quanto riguarda, invece, le informazioni riguardo la sua immatricolazione, possiamo dire che hanno un tasso di confidentiality piu basso. La lista degli studenti ha un tasso ancora minore.
Esempio Integrity:
Informazioni riguardo le allergie di un paziente, richiedono requisiti alti di integrit (un dottore deve poter ritenere affidabili le informazioni che riceve up to date. Se qualcuno falsifica deliberatamente i dati il database dobrebbe essere ripristinato ad una base fidata e si dovrebbe poter associare le informazioni falsificate a chi ha commesso tale falsificazione.) Se consideriamo le informazioni di una newsgroup queste hanno dei requisiti di integrit pi bassi. Ancora meno ne hanno i sondaggi anonimi online.
Esempio Availability: Un sistema che fornisce un servizio di autenticazione a sistemi critici, applicazioni o dispositivi. In questo caso abbiamo possibili perdite finanziarie e dunque la necessit di avere alti requisiti di availability. Se consideriamo il sito web di una universit, l'interruzione del servizio causa imbarazzo, ma nessun danno critico. Dunque si hanno requisiti pi bassi di availability. Infine un elenco telefonico online che non consultabile al mas-

simo crea disagio, ma ci sono comunque fonti alternative per questo tipo di informazioni, bassi requisiti di availability.

1.1.3 Sfide riguardo la sicurezza dei computer

1. La sicurezza dei computer non semplice;
2. Si devono considerare potenziali (inattesi) attacchi;
3. Le procedure utilizzate sono spesso controintuitive;
4. Dobbiamo decidere dove inserire (deploy) meccanismi di difesa;
5. Include algoritmi e informazioni segrete (keys);
6. COMPLETARE L?ELENCO

1.1.4 Un modello per la sicurezza dei computer

Gli utenti ed i proprietari vogliono proteggere gli **asset**, o risorse del sistema: Hardware, software (OS, apps), data (users, system, database), communication facilities and networks (LAN, bridges, routers, ...).
Le nostre preoccupazioni riguardano le **vulnerabilit** di queste risorse (sottratte, danneggiate, non disponibili). Le minacce evidenziano vulnerabilit e rappresentano potenziali danni nei confronti di un **asset**.
Un **attacco** una minaccia che portata avanti da un **attaccante** (or **threat agent**).

- **Attacco attivo:** un tentativo di alterare le risorse del sistema o di influenzare le sue operazioni.
- **Attacco passivo:** un tentativo di imparare o fare uso di informazioni provenienti dal sistema, senza influenzarne le risorse.
- **Attacco dall'interno:** iniziato da un'entit all'interno del perimetro di sicurezza.
- **Attacco dall'esterno:** iniziato da un'entit al di fuori del perimetro di sicurezza, da parte di un utente non autorizzato o illegittimo.

Per **contromisure** intendiamo tutte le azioni intraprese per prevenire, rilevare, ripristinare o minimizzare un rischio che riguardi un asset del sistema.
Segue figure:1

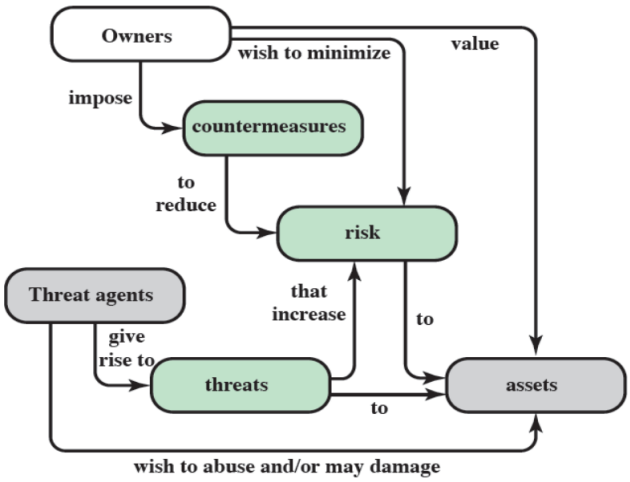

Figure 1: Security concepts and relationships

1.2 Minacce, attacchi e asset

1.2.1 Minacce, conseguenze ed attacchi

- **Rilascio non autorizzato (unauthorized disclosure):** minaccia la confidentiality.
  - Una circostanza o un evento nel quale un'entit non autorizzata guadagna l'accesso a dati per i quali non sarebbe autorizzato.
  - Ad esempio rilascio, intercettazione o inferenza di dati sensibili.
- **Inganno (deception):** minaccia l'integrit.
  - Circostanza o evento nel quale un'entit autorizzata riceve dati falsificati credendo che siano corretti.
  - Pu essere il caso di un utente non autorizzato che si finge utente autorizzato (**masquerade**). Si includono venti relativi alla falsificazione (**alter data**) o al falso diniego della responsabilit nei confronti di uno specifico atto (**repudation**).
- **Rottura (disruption):** minacce ad integrit e availability.
  - Circostanza o evento che interrompe il funzionamento o i servizi del sistema.

  - Ad esempio disabilitare una componente del sistema (**incapacitation**), modificare in modo negativo le funzioni del sistema o i suoi dati (**corruption**), o intasare (overload) una linea di comunicazione **obstruction**

- **Usurpazione**: minaccia all'integrit.
  - Circostanza nella quale un'entit non autorizzata prende il controllo di una funzione o servizio del sistema.
  - Furto di un servizio (**misappropriation**), hacker che guadagnano un accesso non autorizzato (**misuse**.)

Segue figure:2,3,4

Threat Consequence	Threat Action (Attack)
Unauthorized Disclosure A circumstance or event whereby an entity gains access to data for which the entity is not authorized.	**Exposure:** Sensitive data are directly released to an unauthorized entity. **Interception:** An unauthorized entity directly accesses sensitive data traveling between authorized sources and destinations. **Inference:** A threat action whereby an unauthorized entity indirectly accesses sensitive data (but not necessarily the data contained in the communication) by reasoning from characteristics or byproducts of communications. **Intrusion:** An unauthorized entity gains access to sensitive data by circumventing a system's security protections.
Deception A circumstance or event that may result in an authorized entity receiving false data and believing it to be true.	**Masquerade:** An unauthorized entity gains access to a system or performs a malicious act by posing as an authorized entity. **Falsification:** False data deceive an authorized entity. **Repudiation:** An entity deceives another by falsely denying responsibility for an act.
Disruption A circumstance or event that interrupts or prevents the correct operation of system services and functions.	**Incapacitation:** Prevents or interrupts system operation by disabling a system component. **Corruption:** Undesirably alters system operation by adversely modifying system functions or data. **Obstruction:** A threat action that interrupts delivery of system services by hindering system operation.
Usurpation A circumstance or event that results in control of system services or functions by an unauthorized entity.	**Misappropriation:** An entity assumes unauthorized logical or physical control of a system resource. **Misuse:** Causes a system component to perform a function or service that is detrimental to system security.

Figure 2: Threat consequences and attack

	Availability	Confidentiality	Integrity
Hardware	Equipment is stolen or disabled, thus denying service.	An unencrypted CD-ROM or DVD is stolen.	
Software	Programs are deleted, denying access to users.	An unauthorized copy of software is made.	A working program is modified, either to cause it to fail during execution or to cause it to do some unintended task.
Data	Files are deleted, denying access to users.	An unauthorized read of data is performed. An analysis of statistical data reveals underlying data.	Existing files are modified or new files are fabricated.
Communication Lines and Networks	Messages are destroyed or deleted. Communication lines or networks are rendered unavailable.	Messages are read. The traffic pattern of messages is observed.	Messages are modified, delayed, reordered, or duplicated. False messages are fabricated.

Figure 3: Examples of threats to assets

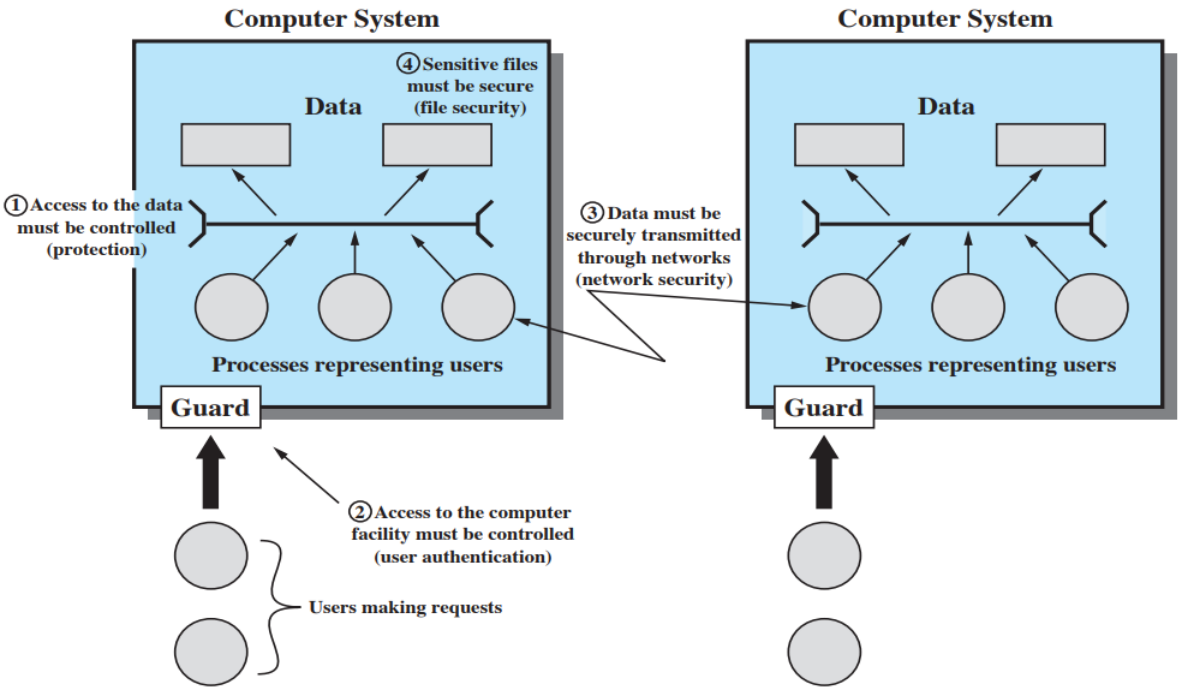

Figure 4: The scope of computer security

1.3 Functional requirement per la sicurezza

Le contromisure possono essere classificate in termini di **requisiti funzionali**, prendiamo ad esempio lo standard FIPS200 che definisce 17 aree che coinvolgono misure tecniche per la sicurezza dei sistemi piuttosto che controlli e procedure di gestione.
Segue figure:5 Ne elenchiamo alcune di rilievo:

Threat Consequence	Threat Action (Attack)
Unauthorized Disclosure A circumstance or event whereby an entity gains access to data for which the entity is not authorized.	**Exposure:** Sensitive data are directly released to an unauthorized entity. **Interception:** An unauthorized entity directly accesses sensitive data traveling between authorized sources and destinations. **Inference:** A threat action whereby an unauthorized entity indirectly accesses sensitive data (but not necessarily the data contained in the communication) by reasoning from characteristics or byproducts of communications. **Intrusion:** An unauthorized entity gains access to sensitive data by circumventing a system's security protections.
Deception A circumstance or event that may result in an authorized entity receiving false data and believing it to be true.	**Masquerade:** An unauthorized entity gains access to a system or performs a malicious act by posing as an authorized entity. **Falsification:** False data deceive an authorized entity. **Repudiation:** An entity deceives another by falsely denying responsibility for an act.
Disruption A circumstance or event that interrupts or prevents the correct operation of system services and functions.	**Incapacitation:** Prevents or interrupts system operation by disabling a system component. **Corruption:** Undesirably alters system operation by adversely modifying system functions or data. **Obstruction:** A threat action that interrupts delivery of system services by hindering system operation.
Usurpation A circumstance or event that results in control of system services or functions by an unauthorized entity.	**Misappropriation:** An entity assumes unauthorized logical or physical control of a system resource. **Misuse:** Causes a system component to perform a function or service that is detrimental to system security.

Figure 5: Threat consequences and attack

- **Access Control (controllo accessi)**: limitare l'accesso alle informazioni del sistema ad utenti autorizzati, limitare i processi che agiscono per contro di utenti autorizzati, limitare il tipo di transazioni e funzioni che un utente autorizzato pu utilizzare.

- **Awarness and Training (consapevolezza e addestramento)** : Assicurarsi che i manager e gli utenti del sistema siano informati riguardo ai rischi per la sicurezza associati alla propria attivit, oltre alle leggi, regolamenti e politiche che riguardano la sicurezza di un sistema informativo. Inoltre ci dobbiamo assicurare che il personale sia ben addestrato per poter svolgere i propri doveri e responsabilit relativi alla sicurezza delle informazioni.

- **Incident response**: stabilire delle capacit operazionali che permettano la gestione di incidenti dovuti alla sicurezza dei dati. Include: preparazione, analisi, contenimento e ripristino. Risulta importante la capacit di tracciare, documentare e gli incidenti di sicurezza, con lo scopo di informare le autorit competenti.

Il concetto fondamentale riportato dal framework quello di **combinare approcci tecnici e manageriali**.

If you think technology can solve your security problems, then you dont understand the problems and you dont technology

1.4 Principi fondamentali di design per la sicurezza

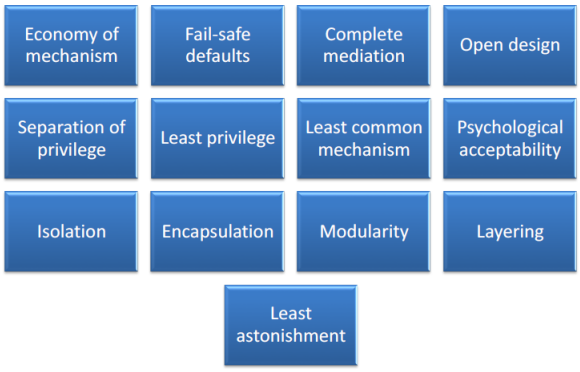

Figure 6: Fundamental security design principles

A dispetto di anni di ricerca, ancora molto difficile progettare sistemi per i quali si possono sistematicamente escludere difetti e che siano in grado di prevenire azioni non autorizzate. Tuttavia sono state documentate delle

good practice per una corretta progettazione di questi sistemi. Principi di design per la sicurezza che siano largamente accettati possono guidare l'implementazione dei meccanismi di sicurezza, inoltre possono fornire un **secondo punto di vista** riguardo le misure da adottare nei confronti di specifiche minacce al sistema.
Segue figure:6
Il National Centers of Academic Excellence in Information Assurance/Cyber Defense elenca i relativi principi fondamentali:

- **Economy of mechanism (economia dei meccanismi):** il design delle misure di sicurezza devono essere pi semplici possibile. Infatti pi sono facili da implementare e testabili meno falle di sicurezza sono esposte.
- **Fail safe as default**: La gestione degli accessi deve essere basata sui permessi e la situazione di deafault deve essere quella di **negare l'acesso**. Infatti un meccanismo che fornisce permessi espliciti tende a fallire rifiutando gli accessi, una situazione sicura che pu essere facilmente individuata.
- **Complete mediation**: Tutte le richieste di accesso devono passare per il meccanismo di controllo degli accessi. Nello specifico **non dobbiamo basare tali decisioni su eventuali cache degli accessi.**
- **Open Design:** Il design dei meccanismi di sicurezza deve essere open. Un esempio dei vantaggi ottenuti riguarda gli algoritmi per la criptografia: in questo modo possono essere analizzati e studiati da esperti, con lo scopo di migliorarne le prestazioni e la sicurezza.
- **Separation of privilege**: Dovrebbero essere necessari pi privilegi per eseguire l'accesso o l'esecuzione di un'operazione. Esempio: **Autenticazione degli utenti a pi fattori** (chiave odt + PIN).
- **Least privilege**: Gli utenti devono avere privilegi minimi in relazione ai task che devono svolgere. In questo caso si evidenzia anche un aspetto temporale: un amministratore che ha privilegi speciali deve avere quei privilegi solo per il tempo necessario, quando eseguono attivit ordinarie quei privilegi devono essere declassati.

- **Least common mechanism**: il design deve minimizzare le funzioni condivise da utenti diversi e quindi ridurre il numero di path di comunicazione indesiderati.

- **Psycological acceptability**: I meccanismi di sicurezza dovrebbero riflettere il modello mentale di protezione degli utenti. Il concetto che i meccanismi di sicurezza non devono interferire impropriamente con il lavoro degli utenti.

- **Isolation:** legata a tre aspetti:

  1. **Gli accessi pubblici dovrebbero essere isolati dalle risorse critiche in modo tale da prevenire disclosure o tampering (manomissioni).** *A lezione stato fatto l'esempio di* ***zona demilitarizzata***.
  2. I processi ed i file di uno specifico utente devono essere isolati l'uno rispetto all'altro.
  3. I meccanismi di sicurezza dovrebbero essere isolati.

- **Incapsulamento:** Una collezione di procedure e data ojects sono incapsulati in un dominio a se stante, cos facendo la struttura interna di un oggetto solo dalle procedure del sottosistema protetto e le procedure possono essere chiamate solo agli entrypoint del dominio.

- **Modularity:** Abbiamo due significati :

  1. Risulta opportuno che le funzioni di sicurezza siano sviluppate separatamente, come moduli protetti. Ad esempio, nel caso di un insieme di protocolli o applicazioni che utilizzano funzioni crittografiche, opportuno che tali funzioni siano sviluppate in un comune modulo di crittografia che pu essere invocato da tali funzioni, anzich implementarle per ogni protocollo o applicazione.
  2. Il secondo riguarda la possibilit di sviluppare il design e l'implementazione dei meccanismi di sicurezza in senso modulare, in questo modo si possono aggiornare o modificare tali moduli senza dover modificare tutto il sistema.

- **Layering:** Utilizzare approcci sovrapposti alla sicurezza (pi di uno) riguardo persone, tecnologia, e aspetti operazionali di un sistema informativo. In questo modo il fallimento di uno tra questi non lascer

il sistema privo di protezione (**defense in depth**: approccio militare per il quale si cerca di ritardare l'avanzata di un attaccante invece che prevenirla).

- **Leas astonishment:** Un programma o un'interfaccia deve rispondere in un modo per il quale meno probabile che l'utente sia disorientato (stupito).

1.5 Superfici di attacco e alberi di attacco

Ci fornisce un terzo punto di vista riguardo le misure che possono essere prese per gestire le minacce alla sicurezza di un sistema. *Segue figure:7*

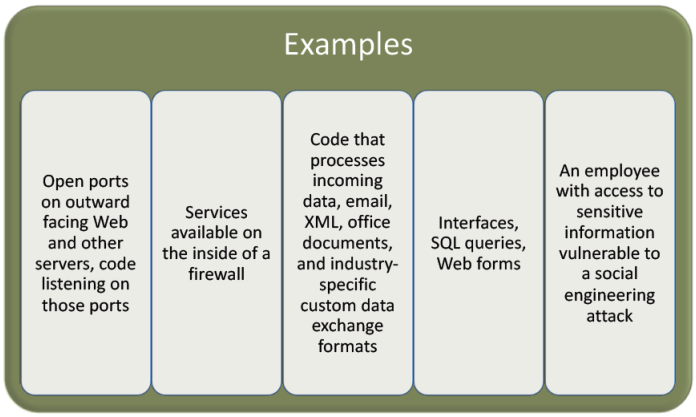

Figure 7: Some reachable and exploitable vulnerabilities in a system

1.5.1 Categorie di superfici di attacco

- **Network attack surface**: Vulnerabilit che riguardano una rete aziendale, una rete su larga scala (wide-area network) oppure l'internet. Si includono vulnerabilit nei protocolli di rete utilizzate per attacchi denial-of-service e altri.

- **Software attack surface**: Vulnerabilit che riguardano in programmi e sistemi operativi (con un riguardo particolare nei confronti del software dei web server).

- **Human attack surface**: Vulnerabilit create dal personale o da esterni (ingegneria sociale, errori umani e insider fidati).

Un'**analisi della superficie di attacco** utile nell'ottica di stabilire la scala e la severit delle minacce a cui un sistema sottoposto. Una volta che la superficie di attacco stata definita, **si pu cercare di ridurla.**

1.5.2 Alberi di attacco

Un **albero di attacco** una struttura dati ramificata (branching) e gerarchica che rappresenta un insieme di potenziali vulnerabilit. **L'obiettivo** quello utilizzare in modo efficace (sfruttare/exploit) tutte le informazioni disponibili sui **pattern di attacco**. Infatti gli analisti della sicurezza possono utilizzare gli attack trees per guidare sia il design del sistema e delle applicazioni, oltre che la scelta e la forza delle contromisure. La conoscenza riguardo i pattern di attacco stata raccolta nel tempo e rappresenta uno strumento importante alla sicurezza. Ad esempio il CERT (Computer Emergency Response Team) uno degli organismi che ha svolto, nel tempo, questo genere di attivit.

- **L'obiettivo (goal)** di un attacco (incidente di sicurezza) rappresentato come radice dell'albero
- **I modi (ways)** attraverso il quale l'attaccante pu raggiungere un goal sono rappresentati iterativamente ed in modo incrementale come rami dell'albero.
- Ogni **sotto-nodo (subnode)** definisce un **sotto-obiettivo (subgoal)**. Questo tipo di scomposizione di goal ricorsiva.
- Le **foglie (leaf nodes)** dell'albero rappresentano i diversi modi di **iniziare** un attacco
- Tutti i nodi escluse le foglie pu essere un **And-node** oppure un **Or-node**.
    1. Per realizzare un goal relativo ad un And-node si devono realizzare **tutti** i suoi subgoal.
    2. Per realizzare un goal relativo ad un Or-node si deve realizzare **almeno uno** dei suoi subgoal (Or non esclusivo).

- I rami possono essere **etichettati** con valori rappresentati: difficolt, costo o altri attributi in questo modo possiamo confrontare attacchi alternativi

1.5.3 Albero di attacco relativo a una applicazione di autenticazione per servizio bancario

- La radice rappresenta l'obiettivo dell'attaccante: **compromettere l'account di un utente**
- I **box verdi** sono gli eventi che costituiscono un attacco.
- I **box bianchi** sono categorie relative agli attacchi.
- Tutti i nod, tranne i nodi foglia, sono or nodes.
- L'analisi utilizzata per generare questo albero di attacco considera 3 componenti coinvolte nell'autenticazione.

  1. **Terminali degli utenti e utenti (UT/U)**: questo genere di attacco mira agli strumenti utilizzati dagli utenti, inclusi i token coinvolti (smart card, password generator), oppure le azioni degli utenti.
  2. **Comunication channel (CC)**: Questi attacchi si concentrano sui canali di comunicazione.
  3. **Internet server bancari (IBS)**: questi sono attacchi offline nei confronti del server che ospita le applicazioni internet della banca. INSERIRE immagine pagina 63.

  In questo scenario possiamo identificare 5 strategie di attacco che utilizzano una o pi componenti delle tre identificate.

  – **Credenziali utente compromesse**: consiste nel monitorare per osservare pin o credenziali dell'utente, rubare un token o una nota scritta a mano contenente un token, utilizzare programmi software malevoli per compromettere il login di un utente, hackerare una smart card, utilizzare un approccio a forza bruta per indovinare il pin di un utente, fare uno sniffing delle credenziali durante il loro passaggio in un canale di comunicazione.

- **Iniezione di comandi**: intercettare le comunicazioni tra un utente e il server della banca, impersonando quindi tale utente valido con lo scopo di ottenere l'accesso ai server della banca.
- **Indovinare le credenziali di un utente**: utilizzare algoritmi di forza bruta inviando login e password casuali.
- **Violazione delle politiche di sicurezza:** un dipendente pu causare un incidente interno di sicurezza ed esporre l'account di un utente, se egli viola le politiche di sicurezza della banca in combinazione con un meccanismo di gestione degli accessi debole.
- **Utilizzare una sessione conosciuta gi autenticata:** forzare un utente a connettersi al server della banca co un ID preimpostato, utilizzato dall'attaccante per rubare l'identit dell'utente.

1.6 Strategia per la computer security

Una strategia globale per provvedere alla sicurezza di un sistema:

- **Specifiche/politiche** : cosa dovrebbe fare lo schema di sicurezza ?
- **Implementazione/meccanismi:** come applicarli ?
- **Correttezza/assurance**: funziona davvero ?

1.6.1 Specifiche/politiche

Dichiarazione formale di regole e pratiche che specificano e regolano come il sistema o un'organizzazione fornisce servizi di sicurezza, per proteggere le risorse critiche del sistema. I **fattori** da considerare nello sviluppare queste politiche sono:

- **Il valore delle risorse da proteggere**;
- **Le vulnerabilit del sistema**;
- **Potenziali minacce e la probabilit di subire un attacco**.

Si devono considerare anche i seguenti **trade-off**:

- **Facilit di utilizzo vs benefici di sicurezza**;

- **Costo della sicurezza vs costi di fallimento e ripristino**

Inoltre queste politiche sono influenzate da **decisioni di business** e **requisiti legali**.

1.6.2 Implementazione/meccanismi

L'implementazione della sicurezza coinvolge quattro processi (courses of action) complementari:

1. **Prevenzione (Prevention)**: l'idea che uno schema ideale di sicurezza tale se nessun attacco ha successo. In qualche caso la prevenzione un traguardo ragionevole. Es: attacchi alla confidentiality prevenuti utilizzando canali criptati.

2. **Rilevazione (Detection)**: In alcuni casi, una protezione assoluta non realizzabile, ma molto pi pratico rilevare gli attacchi alla sicurezza.

3. **Reazione (Response)**: Il sistema pu essere in grado di reagire in modo tale da bloccare un attacco alla sicurezza che stato rilevato e prevenire cos ulteriori danni.

4. **Ripristino (Recovery)**: Se l'integrit dei file risulta compromessa utile averne una copia corretta da poter ricaricare.

1.6.3 Correttezza/assurance

2 CryptoTools

Gli algoritmi crittografici sono una componente importante per la computer security. In questo capitolo si presenta una panoramica dei diversi tipi di algoritmo, ne discuteremo il contesto di applicazione e introdurremo gli algoritmi di uso comune pi importanti.

2.1 Symmetric Encryption

La criptografia simmetrica risale agli anni 70 e per questo anche indicata come **criptografia convenzionale** o criptografia a singola chiave. Nella pratica una tecnica universale per fornire la **confidentiality** dei dati (trasmessi o salvati). Questo tipo di criptografia ad oggi ancora la pi utilizzata.

2.1.1 Model of Symmetric Encryption

Le componenti del modello sono le seguenti:

1. **Encryption algorithm**: esegue le sostituzioni e le trasformazioni sul plaintext.
2. **Input**:
   - **Plaintext**: il messaggio originale da criptare.
   - **Secret key**: le sostituzioni e le trasformazioni eseguite dall'algoritmo dipendono da questa chiave. Due chiavi segrete diverse produrranno due.
3. **Ciphertext**: testo cifrato prodotto come **output** dall'algoritmo di encryption.
4. **Decryption algorithm**: lavora al contrario dell'algoritmo di encryption (testo cifrato — plain text).

INSERIRE IMMAGINE PAGINA 7 I **requisiti** per un uso sicuro sono i seguenti:

1. necessario un algoritmo di encryption che sia **forte**. Ci significa che un attaccante **non deve essere in grado** di scoprire la chiave, nemmeno se in possesso di un insieme di coppie cyphertext, plaintext.
2. Il sender e il receiver devono aver ottenuto la chiave in modo sicuro, si deve inoltre mantenerla sicura.

I pi diffusi algoritmi di symmetric encryption si basano su **block ciphers**. Block ciphers processa il plaintext in input in **blocchi di grandezza fissata** e produce in output blocchi di cyphertext della stessa grandezza. I principali algoritmi simmetrici (block ciphers) sono:

- DES
- Triple DES
- AES

INSERIRE IMMAGINE SLIDE 11

2.1.2 Approaches to Attacking

Gli approcci all'attacco riguardo la criptografia simmetrica sono due:

1. **Cryptanalitic Attacks**: si basa sulle caratteristiche dell'algoritmo. Si tenta di dedurre uno specifico plaintext o la chiave utilizzata (in questo ultimo caso non solo tutti i messaggi gi inviati sono compromessi, ma anche quelli futuri). Si cerca di dedurre informazioni da coppie di plain e cyphertext.

2. **Brute-Force Attacks**: Si prova a decifrare un cyphertex tentando tutte le chiavi possibili, fino a che non si ottiene un plain text che sia sensato. In media per aver successo si devono provare almeno la met di tutte le chiavi possibili. Inoltre dobbiamo conoscere almeno una parte del plaintext per decidere se quello che otteniamo non semplicemente unt testo senza senso (garble).

2.1.3 Data Encryption Standard (DES)

Adottato nel 1977 dal NIST, fino a poco tempo fa era l'algoritmo pi utilizzato.

1. Prende in input **blocchi di plaintext di 64 bit** e una **chiave da 56 bit** per produrre un **cyphertext da 64 bit.**

2. La chiave anch'essa da 64 bit, ma 8 bit sono utilizzati per il **parity check**.

3. L'algoritmo originariamente aveva chiavi da 128 bit che per sono state ridotte a 64 dal NIST.

Analizziamo le **debolezze** dell'algoritmo dal punto di vista degli approcci di attacco visti in precedenza.

- Dal punto di vista delle debolezze dell'algoritmo (**cryptanalysis**), il DEA uno dei pi studiati tra gli algoritmi di encryption. Non sono mai state evidenziate debolezze critiche.

- La lunghezza delle chiavi **56 bit** pertanto si ha un numero di 2^{56} chiavi possibili. La lunghezza delle chiavi rappresenta la debolezza maggiore, infatti l'algoritmo stato violato nel 1998 dalla Electronic Frontier Foundation.

Se l'unico modo di attaccare un algoritmo di enryption quello di utilizzare brute force, allora la metodologia di difesa ovvia: utilizzare chiavi pi lunghe. INSERIRE IMMAGINE PAGINA 14.

2.1.4 Triple DES

1. Ripete tre volte l'algoritmo DES
2. Si utilizzano 1,2 o 3 chiavi.
3. La dimensione delle chiavi di 168 bit.

INSERIRE IMMAGINe PAG 15 Vantaggi e svantaggi rispetto a **DES**:

- **Vantaggi:**
  1. le chiavi a 168 bit permettono di superare le vulnerabilit rispetto a brute force (anche se a sicurezza effettiva corrisponde a 112 bit).
  2. L'algoritmo sottostante lo stesso di DES per il quale, come gi detto non sono state individuate vulnerabilit critiche.
- **Svantaggi:**
  1. Il codice che implementa l'algoritmo non efficiente (computazionalmente oneroso).
  2. utilizza blocchi di **64 bit**, mentre sarebbero desiderabili blocchi di dimensioni maggiori, dal punto di vista della sicurezza e dell'efficienza.

2.1.5 Advanced Encryption Standards (AES)

Rimpiazza il 3DES. Requisiti definiti dal NIST:

- Sicurezza pari a 3DES.
- Efficienza significativamente migliorata.
- Symmetric block cipher.
- Blocchi di 128 bit.
- Chiavi da 128/192/256 bit.

2.1.6 Block Cipher

Un plaintext ha una lunghezza di $n * b$ diviso in n blocchi da b-bit. Nel caso la dimensione del plaintext non sia un multiplodi b si "imbottisce" il file aggiungendo bit. Ogni blocco criptato separatamente. Produce una sequenza di n b-bit.
Electronic CodeBook l'approccio pi semplice per l'encryption con blocchi multipli. Si utilizza l'algoritmo con una **stessa chiave per ogni blocco**. Questo approccio rende possibile applicare la cryptanalysis per individuare delle regolarit nel plaintext, pertanto potrebbe non essere indicato per messaggi di grossa lunghezza.
INSERIRE IMMAGINE PAGINA 20

2.1.7 Stream Cipher Encryption

Uno stream cipher processa gli input in modo continuo producendo un output un elemento alla volta. Nello specifico **l'algoritmo cripta il plaintext un byte alla volta** (la lunghezza pu essere configurata: ad esempio un bit alla volta). L'algoritmo effettua i seguenti passi:

1. Si utilizza uno pseudorandom bit generator, che prende in input la chiave di cifratura, che produce uno stream di numeri a 8 bit che sono apparentemente casuali.

2. L'output di questo generatore, chiamato **keystream**, combinato un byte alla volta con lo stream del plaintext utilizzando un l'operatore **XOR**.

INSERIRE IMMAGINE PAGINA 22

2.1.8 Block Cipher vs Stream Cipher

- Il vantaggio di block cipher che si possono riutilizzare le chiavi.

- Il vantaggio di stream cipher essere pi veloce rispetto alla controparte, inoltre necessario meno codice per l'implementazione.

- Se il generatore di numeri casuali realizzato propriamente lo stream cipher pu essere sicuro al pari di block cipher (a parit di lunghezza delle chiavi).
- block cipher pi appropriato per applicazioni che devono gestire blocchi di dati (trasferimento file, email, database). Al contrario utilizzato nel caso di stream di dati (canali di comunicazione, browser).

In generale entrambi possono essere utilizzati in qualsiasi applicazione.

2.2 Message Authentication and Hash Function

2.2.1 Message (or Data) Authentication

La message authentication fornisce protezione nei confronti di attacchi **attivi** (falsificazione dei dati), mentre la criptografia ci protegge da attacchi **passivi** (eavesdropping).
La message authentication **una procedura che permette alle parti coinvolte in una comunicazione di verificare che i messaggi ricevuti o salvati siano autentici** ovvero:

- **genuini**: i contenuti non sono stati alterati.
- provengono dalla fonte attesa.
- **timeliness**: i messaggi non sono stati volontariamente ritardati o inviati una seconda volta (replay).
- **sequenza relativa** dei messaggi che vengono scambiati dalle due parti.

Tutti questi concerns riguardano **l'integrit dei dati**

2.2.2 Authentication with and without Symm. Encryption

Nel caso di utilizzo della criptografia simmetrica, se solo il sender e il receiver condividono la chiave allora:

- Solo il sender originale sar in grado di fare l'encription di un messaggio ricevuto e depcriptato dal receiver.

- Se il messaggio include un codice per la **error detection** e un **sequence number**, il receiver assicurato del fatto che nessuno modifica stata apportata al file e che l'ordine relativo giusto.

- Se il messaggio ha un timestamp il receiver certo che il messaggio non sia stato ritardato di proposito.

Tuttavia la criptografia simmetrica non un tool adatto per la data authentication. Infatti se un attaccante riordina i blocchi di un singolo ciphertext, i blocchi vengono comunque decriptati con successo. Tuttavia l'ordinamento posticcio dell'attaccante fa si che il testo decriptato perda il suo significato.
Nel caso in cui non si utilizzi criptografia simmetrica l'approccio quello di utilizzare un **authentication tag** che viene allegato in ogni messaggio. In questo caso **non si garantisce la confidentiality** dei dati ma solo **l'autenticazione**. Ovviamente possibile combinare i due approcci in un solo algoritmo.
Tuttavia le funzioni che realizzano la message authentication e l'encryption sono tenute separate poich ci sono situazioni in cui, tra le due, l'encryption superflua:

- Applicazioni nelle quali un messaggio deve essere inviato tramite **broadcast** a pi destinatari. In questo caso molto meno oneroso avere un solo destinatario responsabile di monitorare l'autenticit. Si invia tramite broadcast un plaintext con un authentication tag, dunque il sistema responsabile di eseguire l'autenticazione verifica che ci siano violazioni ed in caso positivo avvisa tutti gli altri destinatari.

- Scenario nel quale due entit si scambiano dati e uno dei due ha un **grosso carico di lavoro** a causa del quale non ha tempo per decriptare tutti i messaggi in entrata.

- Un altro scenario quello dell'**esecuzione di un programma**. Il programma pu essere eseguito senza che il suo codice debba essere decriptato ogni volta (oneroso per il processore). Avere un authentication tag ci permette di controllare comunque **l'integrit** del programma (se richiesto).

2.2.3 Message Authentication Code (MAC)

Un'altra tecnica di autenticazione quella di utilizzare un blocco di dati di piccole dimensioni (MAC) da allegare al messaggio e generato utilizzando una chiave segreta condivisa tra mittente e destinatario. Nello specifico:

1. Quando A ha un messaggio per B, calcola il MAC come una funzione complessa del messaggio e della chiave $MAC_M = F(K_{AB}, M)$.
2. Si trasmette il messaggio pi il MAC a B.
3. B esegue lo stesso calcolo utilizzando il messaggio e la chiave e producendo quindi un altro MAC.
4. Il MAC spedito da A viene confrontato da B con quello da lui generato.

Le propriet che garantisce questo sistema di sicurezza, se la chiave conosciuta solo da A e B e i MAC corrispondono, sono le seguenti:

- Il destinatario ha la garanzia che il messaggio **non sia stato alterato**
- Il destinatario ha la garanzia che il messaggio **proviene dal mittente atteso**.
- Se il messaggio contiene un timestamp si pu garantire **ordine relativo e timeliness**.

INSERIRE IMMAGINE PAGINA 31
Alcune **considerazioni**: un codice MAC da 16 - 32 bit attualmente non garantisce la **collision resistance**. Una funzione H collision resistance se difficile trovare due input che sono mappati sullo stesso output. Ovviamente una funzione che ha pi input che output (dominio> codominio) avr delle collisioni, la collision resistance significa che le collisioni sono **difficili da trovare**.
Un'altra considerazione che, a differenza dell'encryption, l'algoritmo di autenticazione non deve essere **reversibile**.

2.2.4 Cryptographic Hash Function

Si possono utilizzare **one-way hash function** per l'authentication. Una funzione hash accetta un messaggio M di lunghezza variabile e produce in

output un **digest H(M)** di lunghezza fissata. Al messaggio abbinato un multiplo di una lunghezza fissata a cui a sua volta associato un parametro L che indica la lunghezza del messaggio originale. Il parametro L una difficolt in pi per un attaccante nell'ottica di produrre un messaggio alternativo con lo stesso valore hash. **A differenza del MAC non utilizza una chiave segreta come input**.
Non solo il digest garantisce l'authentication, ma anche la data integrity (indfatti se un qualche bit risulta alterato il digest dar un errore).
INSERIRE IMMAGINE PAGINA 34
INSERIRE IMMAGINE PAGINA 36
INSERIRE IMMAGINE PAGINA 37
INSERIRE IMMAGINE PAGINA 38
FINIRE HASH FUNCTION

2.3 Public Key Encryption

Proposta da Diffie e Hellman nel 1976 rappresenta un avanzamento rivoluzionario della tecnologia criptografica.
Gli **ingredienti** di questo tipo di criptografia sono:

- Come al solito un **plaintext** ed il relativo **ciphertext**, ottenuto dal primo tramite un **algoritmo di encryption**. Ovviamente sar necessario anche un algoritmo di **decriptazione**.

- Una coppia di chiavi una **pubblica** e una **privata**, una sar utilizzarta per criptare il plaintext l'altra per decriptarlo (dipende dallo scopo e da quale si sceglie.)

Ci sono diverse **false idee** riguardo la criptografia con chiave pubblica rispetto a quella simmetrica:

- **pi sicura:** non c' alcuna differenza in relazione alla restistenza alla cryptanalysis, tra criptografia simmetrica e a chiave pubblica

- **rende criptografia simmetrica obsolete**: La criptografia a chiave pubblica ha un overhead computazionale considerevole rispetto alla controparte simmetrica, pertanto non c' motivo per il quale la prima dovrebbe sostituire l'altra.

Possiamo criptare un messaggio utilizzando la chiave pubblica o quella privata, analizziamo questi due scenari.

2.3.1 Encryption with public key

Per l'**encryption con chiave pubblica** il procedimento il seguente:

1. Ogni utente genera una coppia di chiavi, una pubblica l'altra privata. La chiave pubblica salvata in un registro accessibile agli altri, mentre la chiave privata tenuta segreta.

2. Se Bob vuole inviare un messaggio ad Alice, **utilizza la chiave pubblica di Alice per criptare il messaggio.**

3. Quando Alice riceve un messaggio lo **decripta utilizzando la sua chiave privata**. Nessun altro potr decriptare il messaggio poich la chiave pubblica di alice, utilizzata per l'encryption, si abbina solo a quella privata di quest'ultima (che solo lei conosce).

4. La distribuzione delle chiavi pi semplice rispetto all'handshaking della controparte simmetrica: le procedure non sono pi semplici.

2.3.2 Encrytpion with private key

Per questo motivo tale schema di encryption **utilizzato con lo scopo di garantire la confidentiality** di un messaggio. Ovviamente la confidentiality che garantita dipende da diversi fattori, tra i quali la sicurezza dell'algoritmo, se la chiave privata tenuta al sicuro e la sicurezza del protocollo di cui la funzione di encryption fa parte.
INSERIRE IMMAGINE PAGINA 47
Alternativamente possiamo effettuare l'**encryption del messaggio con la chiave privata**, attraverso il seguente procedimento:

1. Bob cripta il messaggio con la propria chiave privata

2. chiunque conosca la sua chiave pubblica pu decriptare il messaggio.

Questo schema di encryption utilizzato per **garantire l'authentication e/o la data integrity**:

- Se qualcuno in grado di decriptare un messaggio utilizzando la chiave pubblica di Bob sicuro che quel messaggio provenga da bob, poich solo la chiave privata di Bob avrebbe potuto criptare quel messaggio.
- Solo Bob pu modificare il testo poich le modifiche si possono fare esclusivamente con la chiave privata di bob.
- L'integrit e l'authentication dipendono dagli stessi fattori elencati per lo schema precedente.

2.3.3 Requirement for Public-Key Cryptosystems and Algorithms

I requirements che vogliamo per questo tipo di criptografia sono i seguenti:

- facile a livello computazionale creare paia di chiavi
- facile reperire le chiavi pubbliche per criptare i messaggi
- facile per il receiver conoscere la chiave privata del sender
- impossibile per l'attaccante risalire alla chiave privata essendo a conoscenza della chiave pubblica.
- Infattibile a livello computazionale risalire al plaintext senza essere a conoscenza dalla chiave privata.
- Utile anche se non necessario per tutte le applicazioni poter utilizzare entrambe le chiavi sia per la decriptazione sia per la criptazione.

Di seguito un elenco di algoritmi di encryption a chiave pubblica (dettagli pagina 51 slide):

- **RSA**
- **Diffie-Hellman key exchange algorithm**
- **Digital Signature Standards (DSS)**
- **Elliptic curve cryptography**

Gli utilizzi per questi algoritmi sono i seguenti : INSERIRE IMMAGINE PAGINA 56.

2.4 Digital Signatures and Key Management

Gli algoritmi a chiave pubblica sono utilizzati per due categorie di applicazioni, le **Digital Signatures** e **Key management and distribution**. Gli aspetti che dobbiamo considerare riguardo queste due categorie sono:

- La distribuzione sicura delle chiavi pubbliche (**Public-Key Certificates**)
- La distribuzione delle chiavi simmetriche
- La protezione delle chiavi simmetriche one-time per l'encryption (Digital Envelopes).

2.4.1 Digital Signature

Definita come **il risultato di una trasformazione criptografica di dati, che se propriamente implementata, mette a disposizione un meccanismo per verificare la sua autenticazione, l'integrit dei dati e il non ripudio da parte del firmatario. Pertanto consiste nel criptare un messaggio ottenuto dal plaintext con una chiave privata utilizzata come firma**.Supponiamo che Bob voglia inviare un messaggio ad Alice in modo che sia certo che il messaggio proviene da lui. Il funzionamento il seguente:

1. Bob utilizza una hash function sicura per generare un **hash value per il messaggio**.
2. L'**hash value insieme alla chiave privata** di Bob sono gli input per l'algoritmo che genera la firma digitale. L'output un **blocchetto** che utilizzato come firma digitale.

Quando Alice riceve il messaggio :

1. Calcola un hash value per il messaggio
2. utilizza questo hash value e la chiave pubblica di Bob per verificare che il messaggio provenga effettivamente da lui.

In questo modo Alice sicura che il messaggio arriva da Bob perch:

- Un messaggio che verificato tramite la chiave pubblica di Bob e l'hash del messaggio, ha una firma digitale creata necessariamente con la chiave private di Bob.
- Nessuno pu modificare il messaggio senza conoscere la chiave privata di Bob, pertanto si garantisce anche la **data integrity**.
- Non si garantisce tuttavia la **confidentiality**. Infatti anche se il messaggio fosse completamente criptato (non solo l'hash) chiunque potrebbe decriptarlo utilizzando la chiave pubblica.

INERIRE IMMAGINE PAGINA 58

2.4.2 Public Key Certificates

Una delle principali **debolezze** della **public key encryption** riguarda il fatto che le **chiavi pubbliche possono essere falsificate**. Un qualche utente potrebbe infatti fingere di essere Bob e mandare mandare una chiave pubblica ad un altro partecipante o effettuare addirittura un broadcast di questa chiave. Prima che Bob si accorga della falsificazione e che informi gli altri partecipanti, il falsificatore in grado di di leggere tutti i messaggi che gli altri utenti pensano di inviare a Bob (infatti la chiave pubblica di un utente utilizzata criptare messaggi a lui destinati).
Una possibile soluzione rappresentata dai **certificati a chiave pubblica**: Un certificato consiste in una **chiave pubblica e un user ID del proprietario della chiave**, inoltre il blocco formato dai due deve essere **firmato da una terza parte fidata**:

- Il certificato contiene informazioni riguardo la terza parte piu un'indicazione **del suo periodo di validit**.
- Comunemente la terza parte **un'autorit certificata** come un'agenzia governativa o un'istituzione finanziaria.
- In pratica un utente pu presentare in modo sicuro la sua chiave pubblica ed ottenere il relativo certificato dalla terza parte.
- A questo punto l'utente pu pubblicare il proprio certificato.

- Chiunque voglia utilizzare la chiave pubblica pu ottenere il certificato e verificarne la validit attraverso il garante di terze parti.

L'**X.509 standard** uno schema universalmente accettato per formattare i certificati a chiave pubblica, utilizzata nella maggior parte delle applicazioni di rete (ip security,transport layer security (TLS),Secure shell (SSH) ecc). INSERIRE IMMAGINE A PAGINA 62

2.4.3 Symmetric Key Exchange Using Public-Key Encryption

FUFFA (Guardare 64 per dettagli)

2.4.4 Digital Envelopes

Questa tecnica utilizzata per proteggere la **confidenzialit** dei messaggi, **senza che i partecipanti alla comunicazione si debbano mettere d'accordo in precedenza sull'adozione di una chiave segreta condivisa.** Rappresenta l'equivalente di una raccomandata. Questa tecnica si sviluppa nei seguenti passaggi (supponendo che Bob voglia inviare un messaggio ad Alice):

1. Preparare il messaggio da inviare
2. Generare una chiave simmetrica casuale che sar utilizzata solo una volta.
3. Utilizzare quella chiave per criptare il messaggio
4. Utilizzare la chiave pubblica di Alice per criptare la chiave simmetrica
5. Allegare la chiave simmetrica criptata al messaggio criptato.
6. Alice esegue i passaggi inversi, cominciando col dividere la chiave criptata dal messaggio criptato
7. Decripta la chiave simmetrica utilizzando la propria chiave privata
8. Decripta il messaggio utilizzando la chiave simmetrica ottenuta.

In questo modo solo Alice in grado di decriptare la chiave simmetrica e dunque decriptare anche il messaggio (confidentiality). Se Bob ha ottenuto la chiave pubblica di Alice attraverso il certificato a chiave pubblica, sa che la chiave pubblica valida e che dunque invier il messaggio effettivamente ad Alice (authentication).