Introduction

1. Which are the three main Information Security properties? Please list and briefly describe them.

(Quali sono le tre principali proprietà per la sicurezza dell'informazione? Elencarle e fornirne una breve descrizione.)

Abbiamo definito l'Information Security con un acronimo, CIA.

C sta per **Confidentiality:** le informazioni non devono essere impropriamente divulgate, quindi nessun accesso/letture di dati non autorizzati.

I sta per **Integrity:** le informazioni non devono essere impropriamente modificate, quindi nessuna modifica inautorizzata

Infine, A sta per **Availability:** l'accesso alle informazioni non è impedito ad utenti legittimati.

Abbiamo infine introdotto altre proprietà di sicurezza, quali Authentication *(identificare)* ed Accountability *(responsabilità, non-repudiation)* come casi speciali di CIA.

2. Which are the capabilities of an active attacker?

(Quali sono le capacità di un attaccante attivo?)

L'attaccante attivo è colui che è capace di ritrovare anche una singola vulnerabilità e di sfruttarla per impedire una delle 3 caratteristiche fondamentali della Information Security, per esempio leakando dati privati *(confidentiality)*, corrompendoli *(integrity)* o impedendone l'accesso *(availability)*. Per questo in ogni sistema è importante trovare "tutte" le vulnerabilità possibili, in quanto ad un attaccante ne basta anche solo una. Quest'ultimo è modellizzato nel peggiore degli ambienti possibili, ovvero ha accesso alla rete *(essendo di default un ambiente untrusted)*, può leggere tutti i messaggi *(per questo introduciamo la crittografia)*, etc.

3. What is the meaning of brute force attack?

(Che cosa si intende per attacco brute force?)

Un attacco brute force è un attacco che prova tutti i possibili valori, senza scartarne alcuno. È un attacco esaustivo, quindi arriverà al risultato finale, ma estremamente lento e costoso. A livello teorico, può essere utilizzato quando non si hanno sufficienti informazioni per inferire attacchi basati su social engineering, su vulnerabilità, etc, ma a livello pratico esso è utilizzato quando lo spazio da testare (es. lo spazio delle chiavi) è relativamente piccolo.

Language-based Security and Runtime Verification

1. What does a Safe Programming Language provide?

(Cosa offre un linguaggio di programmazione safe?)

Ne abbiamo discusso durante l'argomento della Language Based Security: in breve, è sempre possibile fare programmi sicuri e non sicuri a prescindere dal linguaggio, ma avere un linguaggio safe rende il tutto estremamente più semplice. Le features garantite da un linguaggio di programmazione safe sono varie, come:

- **memory safety:** se si garantisce che i programmi non accedano mai a parti non/de-allocate di memoria, impedendo questioni come il segmentation fault.
- **type safety:** conseguenza del memory, i tipi garantiscono proprietà invarianti degli elementi del programma.
- **Arithmetic Safety:** cosa fare in casi speciali come l'overflow: avere un comportamento undefined è una delle cause maggiori di problemi *(vedasi C per l'overflow di i=i+1).*
- **Thread Safety:** i data race problems, con aliasing etc è possibile creari incongruenze.

Più formalmente, vi è la necessità di una semantica ben definita in tutti questi ambiti in modo da ottenere un comportamento ben definito.

2. What is type safety?

(Che cosa si intende per type safety?)

Come precedentemente citato, i tipi assicurano proprietà invarianti degli elementi del programma: una variabile dichiarata const sarà sempre quella, un array di lunghezza 10 avrà sempre quella lunghezza, etc. In questo modo si evita il problema di un comportamento indefinito, poiché su tipi diversi sono possibili operazioni diverse *(abbiamo usato come esempio l'impossibilità di moltiplicare un booleano per un altro, garantendo che solo tra numeri [int, double, ..] questa operazione fosse valida)*.

3. Which is the difference between safety and security?

(Qual è la differenza tra safety and security?)

In italiano tendiamo a tradurre entrambi i termini con "sicurezza", ma c'è una grossa differenza tra i due termini. In caso si stia parlando di safety, ci si sta riferendo ad una protezione da incidenti, anche casuali, come potrebbe essere il caso di un evento atmosferico che si abbatte su un data center. Invece, quando si usa il termine security, ci si riferisce ad una protezione da agenti intelligenti, entità con uno scopo ben preciso che riguarda il danneggiare, impedire o sottrarre informazioni da un sistema.

4. Which are the advantages and the limitations of Run Time Verification?

(Quali sono i vantaggi e i limiti della Run Time Verification?)

Nella Runtime Verification si esegue il programma da analizzare e si osserva l'execution trace, dal quale poi verrà costruito un modello da analizzare. Le tracce sono una sequenza finita di eventi, quindi una vista formale di un'esecuzione discreta. La parte che controlla è chiamata monitor che eseguirà dei verdetti su

delle proprietà, ed esso può essere offline *(in caso di file di log, ..)*, online esterno *(thread parallelo [a]sincrono)* od online interno *(codice all'interno dell'applicazione stessa)*.

Le limitazioni sono che si ha meno copertura del codice, in quanto sarà necessario fare una grande quantità di testing *(basti pensare a tutti i tipi di input possibili)* e la necessità che il codice sia eseguibile *(esecuzioni simboliche, …)*. Inoltre è più costoso di un'analisi statica e solitamente più complesso, senza contare che potrebbe non terminare *(problema della fermata)*.

I vantaggi invece sono che, essendo a runtime, è più preciso di una astrazione statica.

Cryptography

1. DES is block cipher: it takes ...-bit block of plaintext as input, uses a ...-bit key and has ... rounds.

(DES è un cifrario a blocchi, i cui blocchi sono di ... bit e la cui chiave è di ... bit e in cui i round sono ...)

Blocchi da 64 bits, usando una chiave da 56 bits ed ha 16 rounds

2. Which is the difference between substitution ciphers and transposition ciphers?

(Quale differenza c'è tra i cifrari a sostituzione e i cifrari a trasposizione?)

I cifrari a sostituzione sostituiscono il testo secondo uno schema ben preciso ed il destinatario esegue l'operazione inversa per decriptare il crittogramma. Abbiamo visto per esempio le sostituzioni alfabetiche (e poi l'evoluzione in polialfabetiche) per esempio nel famoso Cifrario di Cesare.

I cifrari a trasposizione sono simili ai precedenti, ma più complessi: l'ordine delle lettere è alterato secondo uno schema che sia il mittente che il destinatario devono conoscere. Per esempio, potrebbe essere il mettere le lettere in una griglia e leggendo verticalmente invece che orizzontalmente.

Entrambi questi tipi di cifrario sono facilmente crackabili persino da un brute force, in quanto lo spazio delle chiavi in caso di Cesare è 25 (26 lettere, -1 per l'input). In caso il brute force sia complesso, esistono altre metodologie come l'analisi statistica *(controllare la frequenza delle lettere a seconda del linguaggio)*, etc.

3. What is a message authentication code?

(Che cosa è un message authentication code?)

Una famiglia di funzioni hash parametriche rispetto ad un parametro segreto condiviso tra due parti. È utilizzato nella Message Authentication dove si conferma che qualcuno è la sorgente reale di un dato.

4. Which are the three fields of application of the public-key systems?

(Quali sono le tre categorie di applicazione dei sistemi a chiave pubblica?)

La distribuzione di chiavi di crittografia, il bind di un'identità ad una chiave e la generazione/mantenimento/revoca delle chiavi.

Key Management

1. What are the Certification Authorities?

(Che cosa sono le Autorità di Certificazione?)

Nella gestione e distribuzione di chiavi a certificati pubblici, le Certification Authorities sono quelle da contattare per ottenere un certificato: esso associa criptograficamente un'identità ad una chiave.

2. What is cross-certification?

(Che cosa si intende con cross-certification?)

Ci sono due modi di certificare, il primo è detto gerarchico dove i certificati vengono scambiati fino al primo antenato comune ed il secondo è detto web of trust, dove la Certification Authority è assente e ci si basa sul giudizio degli utenti *(usato in Pretty Good Privacy)*.

Protocols

1. Which is the potential problem with the following protocol? (2: S -> A, 3: A -> B)

(Qual è il problema che si può creare con il seguente protocollo?)

$$
\begin{array}{lll}
1 & A \rightarrow S & A, B \\
2 & A \rightarrow S & \{K_{AB}, B\}_{K_{AS}}, \{K_{AB}, A\}_{K_{BS}} \\
3 & B \rightarrow A: & \{K_{AB}, A\}_{K_{BS}}
\end{array}
$$

Abbiamo introdotto il nonce *(number once, numero da usare una volta)* per via della possibilità che un attaccante possa usare una chiave di una vecchia sessione (replay attack). Basta aggiungere N_A come nonce di A e N_B come nonce di B.

Ciò ha come conseguenza l'impossibilità di avere key confirmation, ovvero né A né B possono sapere se l'altro ha ricevuto K_{AB}.

2. Given the following protocol, describe a possible type flaw attack.

(Dato il seguente protocollo, descrivere un possibile attacco di tipo type flaw.)

$$
\begin{array}{lll}
1 & A \rightarrow B: & A, \{N_A\}_{K_{AB}} \\
2 & B \rightarrow A: & \{N_A + 1, N_B\}_{K_{AB}} \\
3 & A \rightarrow B: & \{N_B + 1\}_{K_{AB}} \\
4 & B \rightarrow A: & \{K_{AB}, N'_B\}_{K_{AB}}
\end{array}
$$

Un utente malevolo C può far accettare ad A "N_A + 1" come chiave di sessione per via del nonce che è predicibile.

3. Given the asymmetric-key Needham-Schroeder protocol, describe the replay attack found by Gavin Lowe.

(Dato il protocollo di Needham-Schroeder a chiave asimmetrica, descrivere il replay attack trovato da Gavin Lowe.)

1 $A \to B: \{N_A, A\}_{K_B}$
2 $B \to A: \{N_A, N_B\}_{K_A}$
3 $A \to B: \{N_B\}_{K_B}$

Siccome A non ha modo di collegare N_B a B, B manderà anche il suo identificatore nel messaggio 2, quindi:

B -> A : $\{N_A, N_B, B\}_{KA}$

Trattandoli però come coppie (N_B, B) da mandare ad A come inizio di una nuova esecuzione è nuovamente un attacco.

BAN logic

1. What is BAN logic and which are its aims?

È una logica basata sui belief con regole di deduzione. Il suo scopo è sia quello di formalizzare protocolli sia per autenticazione. L'abbiamo introdotta in quanto spesso si hanno descrizioni ambigue *(quindi non formali)* e necessitiamo di un modo per chiarire le assunzioni.

Ci sono 4 costrutti principali: *believes, sees, once said, controls*.

2. Write the following formula in BAN logic terms.

(Scrivere nella BAN logic questa formula.)

A believes that B believes they share a secret key.

A believes B believes A <- k -> B

3. How the protocol messages are transformed in idealized messages?

(Come si trasformano i messaggi dei protocolli e quella dei protocolli idealizzati?)

Ogni messaggio viene trasformato in una serie di assunzioni di BAN.

Paulson Inductive Method

1. Which are the events that occur in traces?

(Che tipo di eventi compongono le tracce?)

Le tracce sono una lista di eventi send *(Says A B X – "A manda X a B")* e store *(Notes A X – "A salva X per usi futuri").*

2. Is the accidental loss of information modelled in this approach?

(Viene modellata la perdita accidentale di informazione in questo approccio?)

Sì *(Oops rule)*.

3. What are the operators parts, analz and synth supposed to do?

(Che cosa fanno gli operatori parts, analz e synth?)

L'insieme H contiene tutti i messaggi mandati all'interno di una traccia. Gli operatori aggiungono a questo insieme altri elementi derivabili da H stesso, dove:

- **Parts:** aggiunge le componenti singole di ogni messaggio + corpo dei messaggi criptati
- **Analz:** come il precedente, ma le chiavi con cui si cripta sono conosciute $Analz \subseteq Partz$
- **Synth:** riguarda la falsificazione dei messaggi, ovvero i messaggi che possono essere ricostruiti da H

4. How the attacker is taken into account?

(Come si tiene conto dell'attaccante?)

Si osserva la conoscenza iniziale *(initState)* e si controlla l'insieme dei messaggi che una spia può vedere in una traccia *(spies)*. Infine, un ulteriore insieme *used* è utilizzato per formalizzare la freshness *(nonces, etc)*.

Process algebras and security

1. Which are the main primitives in the Pi Calculus?

(Quali sono le primitive principali del Pi Calcolo?)

Nel Pi Calculus non abbiamo le classiche primitive che ci aspettiamo, ma si dà grande importanza a nomi e co-nomi *(N)* che insieme formano le labels (L) che quando unite all'azione "null" tau *(silent action)* formano le azioni *(Act)*.

2. Which are the main differences among Spi-calculus, Pi calculus and LySa?

(Quali sono le principali differenze tra Spi-calcolo, Pi calcolo e LySa?)

Spi-calculus è un'estensione del Pi Calculus con le primitive criptografiche, dove le chiavi possono essere dinamicamente create e comunicate.

In Lysa, ispirato allo Spi-calculus, si ha invece che i processi comunicano su una rete globale ed usano crittografia a chiavi simmetriche. Qui si introducono le annotazioni che possono essere considerate per bloccare l'esecuzione di un processo quando una di queste è violata, tramite l'error component.

CFA and its application to Security

1. What is static analysis and which are its advantages?

(Cosa si intende per analisi statica e quali sono i suoi vantaggi?)

L'analisi statica è la predizione di approssimazioni del comportamento dinamico di un programma. Ciò che si ottiene è l'analisi di proprietà che varranno ad ogni esecuzione del programma stesso. Abbiamo introdotto come tecniche l'abstract interpretation, i type systems ed infine la control flow analysis.

2. How must the over-approximation be interpreted concerning the prediction of events or violations?

(Come deve essere interpretata la sovrapprossimazione per quanto riguarda la predizione degli eventi o delle violazioni?)

Essendo un'approssimazione, causerà ovviamente delle incoerenze all'atto pratico. Un modello astratto non è perfetto e può produrre in caso di violazioni o altro sia falsi positivi che falsi negativi, e questo è un fattore chiave di cui tenere conto. Per questo, l'assenza di errori riscontrati dall'analisi non implica in alcun modo l'assenza assoluta di errori e nemmeno che la presenza di errori sia indicante di qualità negative: una volta riscontrati problemi, vanno controllati.

3. What kind of information the components ρ e κ in the CFA applied to the pi calculus collect?

(Che cosa raccolgono le componenti ρ e κ nella CFA vista del Pi calcolo?)

ρ è una funzione che prende un nome e restituisce un insieme di nomi a cui può essere associato

κ prende un nome e restituisce un insieme di nomi che possono essere mandati su quell'input

Quindi (ρ, k) è una stima valida quando (ρ, k) |= P, ovvero soddisfa un insieme di clausole logiche.

4.

$$P \mid Q \mid R = (\underbrace{\overline{c}d.P' | c(y).y(z)P''}_{P} \mid R \mid \underbrace{c(w).\overline{w}c.Q'}_{Q})$$

Which could be the result (in terms of the ρ e κ components) of the analysis in the above example?

(Quale potrebbe essere il risultato dell'analisi nell'esempio sopra nelle componenti ρ e κ?)

ρ: c può solo essere associato a c, mentre tutti gli altri valori saranno associati a d

k qualsiasi valore che non sia d e z

5. Which is the aim of the component ψ in the CFA for LySa?

(A cosa serve la componente ψ nella CFA per LySa?)

Ψ, chiamato error component, colleziona punti dove le asserzioni nelle notazioni possono essere violate. È interessante in quanto se è = 0, il reference monitor non può abortire l'esecuzione di P.