Introduzione

Qual è il nesso tra la sicurezza e la relazione di fiducia che ogni giorno instauriamo con i sistemi con cui veniamo in contatto?
Ritroviamo un esempio di fiducia nell'impiego dell'algoritmo RSA, la cui robustezza è fondata sull'ipotesi che i numeri primi scelti siano abbastanza grandi.
Ma cosa significa dunque "abbastanza grandi"? Quando possiamo definire un sistema sicuro e fino a che punto siamo disposti a fidarci di esso?

Con queste domande ci apriamo alla consapevolezza che la sicurezza è un processo complesso e che, forse troppo spesso, assumiamo per comodità che un sistema sia sicuro perché verificare la sicurezza richiede tempo, energia o conoscenze di cui non disponiamo.

Terminologia

Presentiamo alcuni termini che verranno utilizzati a seguire:

- Repository di vulnerabilità
- Vulnerabilità
- Fix
- Bug
- Patch
- Retrocompatibilità
- Valutazione del rischio
- Policy

📝 **Possibile domanda d'esame**: cos'è un fix e quali sono le sue implicazioni di sicurezza?
Esempio di risposta: un fix è una modifica, più o meno grande, al codice, per risolvere un bug, cioè un comportamento inatteso.
Gli aspetti rilevanti riguardano la continuità del servizio, i costi di sviluppo, se l'analisi costi-benefici è positiva o meno, se continuerà a funzionare dopo sull'infrastruttura nel suo complesso e se il bug è davvero sistemabile (quindi se non è legato all'hardware).

Se da un'analisi costi-benefici scopriamo che la **gold measure** (la soluzione ottima) per il problema è inapplicabile dobbiamo trovare un modo di ammortizzare, ad esempio attraverso campagne di sensibilizzazione o downgrade della funzionalità che ha la vulnerabilità (ove possibile).

L'aggiornamento del software è la prima misura di sicurezza perché diminuisce le possibilità di avere software vulnerabile.

Esempio di alcuni ambiti di interesse della sicurezza

- Programmazione in generale (null pointer dereference, buffer overflow)
- Sistemi operativi (protezione di memoria, controllo di accesso)
- Servizi di uso quotidiano: moneta elettronica, autenticazione
- Database (linking attack, k-anonymity, gestione dei privilegi, leak di dati, cifratura dei record, injection, problemi di progettazione delle tabelle)
- Reti

La sicurezza ha molto a che vedere con le **policy**, un set di regole scelte da chi produce il software, e non c'è una serie di regole universalmente giuste o sbagliate. Ad esempio non tutti i sistemi operativi hanno le stesse policy sui dati degli utenti.

La sicurezza è la coerenza del sistema con le sue regole di funzionamento (policy).
Un attacco è fare in modo che il sistema non funzioni in maniera coerente rispetto a come era stato progettato.
Un problema di sicurezza può quindi essere anche una policy troppo permissiva, sebbene ciò non costituisca propriamente un attacco.
Questa, che sembra una sottigliezza, è molto importante anche per la sua valenza legale.

Approcci moderni alla sicurezza

Security-by-design è l'approccio moderno alla sicurezza. Un'applicazione nuova va pensata sicura, non messa in sicurezza a posteriori, come avveniva in passato.
È anche importante che la sicurezza non si faccia per **obscurity**. Spesso gli algoritmi rinomatamente sicuri sono pubblici, così come il resto del software.

Elenco di lettura e approfondimenti:

- Security-by-design: strumenti e metodologie per lo sviluppo sicuro del software
- Heartbleed
- TDE - Transparent data encryption
- Pseudo-nimizzazione delle tuple di un database
- Esfiltrazione di dati
- Attributes based authentication - autenticazione basata sugli attributi