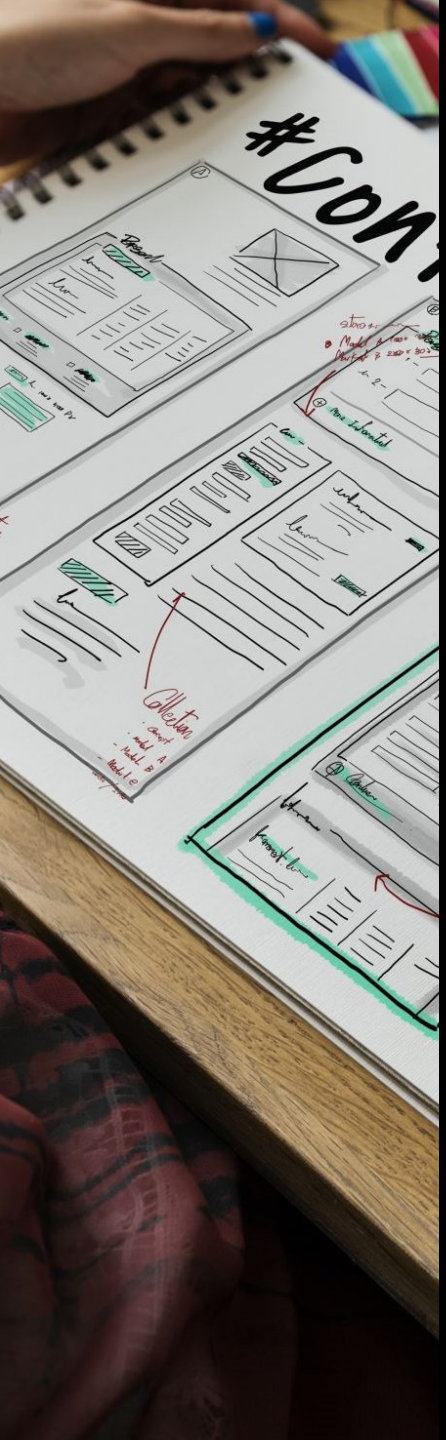


# Ciclo di Vita del Software Sicuro

Vincenzo Calabrò

# La sicurezza nel ciclo di vita del SW e ciclo di vulnerabilità del SW

## Sicurezza vs funzionalità

- Progettare un sistema SW che garantisca sia le proprietà di sicurezza che le qualità del software è un' attività complessa

- L' analisi di sicurezza (detta anche *analisi e gestione dei rischi*) è importante per riuscire a bilanciare tra sicurezza e funzionalità
  - È indispensabile per decidere quali obbietti realizzare e quali proprietà e qualità garantire

- L' analisi di sicurezza, è un processo **ortogonale** al processo di sviluppo del SW

## Modello ottimale per SW sicuro

- Il modello a spirale è il modello di sviluppo ottimale per poter effettuare un' efficace analisi di sicurezza
  - Prevede al suo interno la fase di analisi dei rischi
  - E' un modello di processo flessibile
  - Consente il raffinamento successivo
  - Consente la validazione già nelle fasi iniziali con possibilità di integrare soluzioni trovate a posteriori

## Ingegnere della sicurezza

- Nuova figura professionale
- Si occupa di analisi di sicurezza ortogonalmente al processo di sviluppo
- Si richiede:
  - profonda conoscenza di sviluppo del SW
  - competenza di sicurezza

# Artifatti e stakeholders

- L' analisi di sicurezza comporta l'impiego di tutti gli **artifatti** già esistenti e degli **stakeholder** convolti come risorse per l'identificazione dei rischi
- **Artifatti** esistenti:
  - documenti dei requisiti, documenti dell'architettura e del design, modelli, codice esistente ed anche casi di test
- **Stakeholders**:
  - utenti
  - clienti
  - ingegneri del SW/della sicurezza
  - esperti di dominio

# Politiche e Meccanismi di Sicerezza

**Politica dice cosa è e cosa non è consentito**

- Una politica definisce la "security" di un sistema
- Spesso definita in linguaggio naturale ed in modo ambiguo

**Meccanismo di sicurezza è un metodo, uno strumento, o una procedura per applicare e garantire una politica di sicurezza**

**Composizione di politiche**

- In caso di composizione di sistemi sicuri
- Se le politiche sono in conflitto, le discrepanze possono causare vulnerabilità per la sicurezza dell'intero sistema

# Sicurezza e requisiti (1)

- Nella definizione dei requisiti connessi alla sicurezza occorre identificare:
  - **ciò** che va protetto, **da chi** va protetto e per **quanto tempo** va protetto
  - **quanto** vale mantenere certi dati protetti
- Solitamente i **requisiti di sicurezza** fanno parte dei requisiti non funzionali
- I requisiti devono essere chiari e completi
  - Es. "Questa applicazione deve usare la crittografia quando necessario"
    - Requisito mal posto perché descrive la soluzione senza diagnosticare il problema

## Sicurezza e requisiti (2)

- Il documento dei requisiti deve specificare *cosa* un sistema deve e non deve fare, ma anche *perchè* deve farlo
  - Es. "I numeri delle carte di credito vanno protette contro potenziali furti perchè sono informazioni delicate"
    - Requisito ben posto; la scelta di come proteggere queste informazioni può essere rimandata alla fase di specifica

# Analisi dei rischi (1/2)

- Ha l'obiettivo di identificare i rischi possibili e di valutare strategie per prevenirli o affrontarli
  - L'analisi dei rischi influisce sul processo di auditing
- I rischi vanno classificati in base alla loro severità
  - Non è una classificazione assoluta ma context-sensitive
  - La classificazione è importante per allocare risorse per il testing e l'analisi del siste,a

# Analisi dei rischi (2/2)

- È difficile comparare sistemi in termini di sicurezza
  - Non esiste una metrica "assoluta" ma linee guida molto generali per la valutazione e la comparazione di sistemi sicuri
    - Come eseguire l'analisi di sicurezza
    - Che tipo di rischi considerare
- Il **livello di protezione** è funzione della probabilità che un attacco si verifichi e degli effetti dell'attacco qualora succeda
  - Se un attacco è improbabile, il livello di protezione è basso
  - Se un attacco improbabile causerebbe lunghi ritardi nella produzione di una società, ed uno probabile sarebbe solo un fastidio, sforzo maggiore va messo nel prevenire l' attacco improbabile

# Sicurezza e specifica

- La specifica deve integrare eventuali possibili soluzioni a rischi individuati

- L'analisi di sicurezza continua in fase di specifica
  - Aiuta avere una specifica:
    - **dettagliata** che descriva la reazione del sistema a circostanze critiche
    - **formale**, completa e non ambigua, ma anche chiara e comprensibile
    - **eseguibile** per avere un feed-back immediato

- Se nuovi rischi sono individuati, vanno riflessi nei requisiti

# Sicurezza e design

- Nella fase di design occorre identificare:
  - come i dati passano da una componente all'altra e garantirne l'integrità
  - utenti, ruoli e diritti che sono esplicitamente dichiarati o implicitamente supposti dal design (sulla base di politica per confidenzialità e privacy)
  - ogni soluzione potenzialmente applicabile a problemi individuati
  - la relazione di *trust* di ciascuna componente
- Il *security plan* va attuato in fase di design per essere effettivo
  - Esempio di sistemi progettati senza alcun principio di sicurezza: window9X e Unix

# Sicurezza ed implementazione

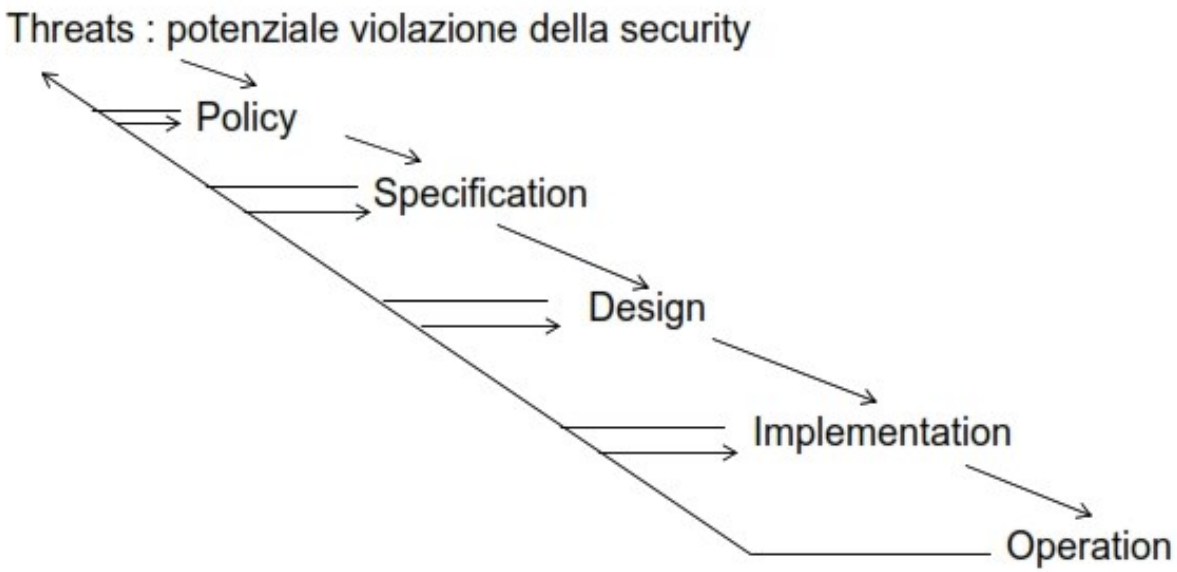
- L' implementazione è una fase critica
  - Molti attacchi sono possibili a causa di errori introdotti in fase di stesura di codice

- Compito dell' ingegnere della sicurezza è quello di:
  - identificare le linee guida per la stesura del codice
  - applicare (se possibile) metodi formali per la prova della correttezza del SW
  - predisporre tecniche per il controllo del codice (*code audit*)

# Sicurezza e testing

- Per sistemi sicuri si distingue tra:
  - **testing funzionale**: comporta il mettere alla prova un sistema per determinare se il sistema fa ciò che si suppone debba fare in circostanze normali o critiche
  - **testing di sicurezza**: comporta il mettere alla prova un sistema allo stesso modo in cui può provarlo un utente malizioso, ossia sfruttando i punti di debolezza del sistema
- Code coverage rimane ancora un metodo importante per il testing di sicurezza
  - Permette di scoprire se esistono parti di codice che non vengono mai eseguite
    - Codice non eseguito può avere seri bachi sfruttabili per attacchi di successo

# Visione d'insieme e ciclicità del processo



# Ciclo di vulnerabilità

- E' una sequenza di eventi che si susseguono molto comunemente nel mondo della sicurezza
  - Va dalla determinazione alla eliminazione di una vulnerabilità del SW

- E' un ciclo **ortogonale** al ciclo di vita del SW
  - Solitamente  ha luogo dopo la distribuzione del SW

# Ciclo di vulnerabilità: sequenza di eventi

1. Viene scoperta una nuova vulnerabilità in una componente SW
2. Si cerca una soluzione al problema:
   - Si **analizza** la vulnerabilità, **sviluppa** una soluzione e la si **testa** in un ambiente controllato, si **distribuisce** la soluzione
   - Analisi interna o affidata ad ente esterno
3. La patch distribuita (spesso rilasciata come parte di update automatico), viene installata dagli addetti
4. Tecnici della sicurezza analizzano frammenti di codice per scoprire vulnerabilità simili

**A questo punto il ciclo si ripete !!!**

# Ciclo di vulnerabilità: conseguenze

- Sistemi e reti vulnerabili raramente vengono riparate durante il ciclo di vita della vulnerabilità
  - Le patch sono ricevute come parte di un versione aggiornata del SW
- Se la vulnerabilità è seria, o gli attacchi sono gravi, anche i media ne danno pubblica informazione, spesso con catastrofiche conseguenze
- Se non si agisce in modo tempestivo, utenti *malevoli* possono riuscire a sfruttare la vulnerabilità causando e propagando danni senza misura

# Ciclo di vulnerabilità: quanto frequente?

- Per farsi un'idea delle dimensioni del problema:
  - su 100 applicazioni Internet che forniscono servizi critici, esistono circa 10 mila bug di sicurezza;
  - sistemi operativi come Linux o Window siano soggetti a centinaia di migliaia di bugs di sicurezza.

# Criteri di valutazione della sicurezza

- La valutazione di sistemi critici sicuri richiede la definizione di uno standard per l'analisi

- Nel 1985 viene pubblicato l' **Orange Book**
  - *Department of Defence Trusted Computer Systems Evaluation Criteria*
  - In collaborazione tra il NSA (National Security Agency) ed il DoD (Department of Defence)
- Successivamente sono stati introdotti
  - Information Technology Security Evaluation Criteria (ITSEC)
  - Federal Criteria
  - Common Criteria

# Criteri di valutazione della sicurezza

**Obiettivo di questi standard:**

- Definire criteri e scale di valori per ***misurare*** l'affidabilità di prodotti e di sistemi
- Definire complesso di ***documenti*** ed artifatti in base ai quali fare la valutazione
- Definire una ***metodologia*** di valutazione da seguire

**Finalità degli standard:**

- Sensibilizzare sull'aspetto security oltre che safety
- Poter confrontare prodotti diversi
- Stabilire regole di valore internazionale

# Valutazione di prodotti e di sistemi

- La valutazione di **prodotti** (entità utilizzabili in diverse situazioni e diversi ambienti e con diversi gradi di sicurezza)
  - Devono soddisfare requisiti delle *security class* dell' orange book e *protection file* dei Common Criteria
- La valutazione di **sistemi** (insieme di vari prodotti assemblati)
  - Vengono valutate le singole componenti come specificato dall' ITSEC
- La valutazione è affidata ad un attore indipendente dal costruttore
  - Stati Uniti: agenzia governativa
  - Europa: agenzie certificate ad assolvere il compito

## Classi di valutazione

**Parametri di valutazione:**

- ***Politiche di sicurezza*** (mandatorie o discrezionali)
- ***Etichettatura degli oggetti*** (in base al livello di criticità)
- ***Identificazione dei soggetti*** (garantita mediante meccanismi di autenticazione)
- ***Auditing*** (accessi continuamente monitorati)
- ***Documentazione*** (importante per uso e testing)
- ***Affidabilità*** (meccanismi di sicurezza attivi per evitare il blocco dei sistemi)

## Classi di valutazione

**4 classi di ripartizione (gerarchiche):**

- D (*minimal protection*): no classi, no meccanismi di protezione
- C (*discretionary protection*): controllo di sicurezza discrezionale (eventualmente autenticazione e auditing) e politica di riuso
- B (*mandatory protection*): controllo obbligatorio
  - B1 (labelled security protection): implementano politiche multilivello secondo il modello di Bell-La Padula
  - B2 (structured protected): ulteriori vincoli in fase di progettazione per la protezione della memoria ed esecuzione di processi in spazi limitati
  - B3 (security domains): meccanismi di prevenzione di attacchi estrni e sistemi di riprostino dopo anomalie

# Classi di valutazione

**4 classi di ripartizione (gerarchiche):**

- .....
- A (*verified protection*): uso di metodi formali per la verifica di sicurezza
  - Bisogna verificare formalmente che TCB siano corrette rispetto al modello formale delle politiche e che implementano le politiche prefissate
    - TCB (trusted computing base): componenti HW e Sw *fidati*

## Common Criteria ed Evaluation Methodology

- **Common Criteria** vers. 2 (~ 600 pagine), standard ISO
  - Definiscono un insieme di classi e di componenti progettate per essere opportunamente combinate per definire profili di protezione per ogni tipo di prodotto IT, incluso hardware, firmware, software
- **Common Evaluation Methodology** (circa 400 pagine) definisce le modalità di valutazione di un sistema in base ai criteri comuni
  - Ad es., l'industria di smart card, sotto la guida dello Smart Card Security User's Group ha largamente utilizzato i common criteria per standardizzare approcci per la sicurezza nel loro mercato mondiale

# Processo di valutazione (1)

1. Wall Street produce un profilo di protezione per il firewall usato per proteggere le macchine che contengono informazioni finanziarie dei clienti.
2. Il profilo viene valutato in base alle Common Evaluation Methodology per garantire che esso sia consistente e completo.
3. Ottenuta la valutazione, il profilo viene pubblicato (target di sicurezza).
4. Un vendor realizza una sua versione (target di valutazione) di firewall dotato del profilo di protezione definito da Wall Steet
5. Il *target di valutazione* viene inviato ad un istituto accreditato per la valutazione rispetto al *target di sicurezza.*

## Processo di valutazione (2)

6. L'istituto applica la Common Evaluation Methodology per determinare in base ai common criteria se il target di valutazione soddisfa il target di sicurezza.
7. Se la valutazione ha esito positivo, la documentazione di testing dell'istituto viene inviata al NVLAP (National Voluntary Lavoratory Accreditation Program) per controllarne la correttezza.
8. Se gli atti vengono approvati, il prodotto ottiene la **certificazione** di prodotto valutato in base ai common criteria.

## Limiti dei *common criteria*

- La valutazione della sicurezza è un'attività molto più complessa dell' applicare un target di sicurezza ad un target di valutazione

- I problemi di sicurezza sono *context-sensitive* ed è difficile valutarli in base a criteri comuni

- Come molti standard, definiscono il "cosa" e non il "come" mentre nel campo della sicurezza è fondamentale sapere *come* affrontare problemi di sicurezza e gestire i rischi

## Valutazione delle smart cards

- La maggior parte degli enti che lavorano sulla sicurezza delle smart cards non sono interessati ai common criteria.

- Dal 2001 lo Smart Card Security User's Group sta cercando di definire dei Common Criteria per la valutazione delle smart cards.
  - Esistono due profili per la sicurezza
    a) Quello europeo che mira maggiormente alla protezione fisica dei dati;
    b) Quello americano che mira maggiormente alla protezione SW e alla prevenzione di attacchi noti;

## Cosa è un attacco

**Un attacco ad un sistema è ogni atto malevolo contro un sistema o un complesso di sistemi**

**Nell'ambito di un attacco vanno distinti:**

- *goal*
- *sottogoal*
- *attività*
- *eventi*
- *conseguenze*
- *impatto*

# Fasi di un attacco (1)

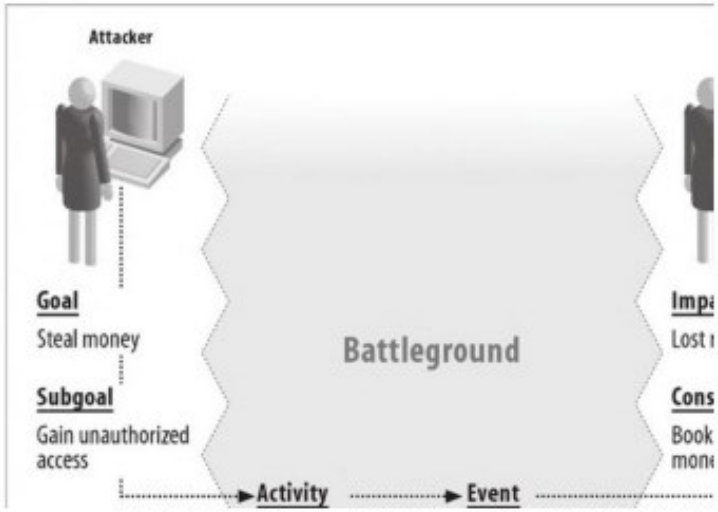
- *Goal*
  - Il danno causato al sistema
    - Es. rubare denaro, servizi, ecc.
- *Sottogoal*
  - Gli obbiettivi intermedi necessari a raggiungere il goal
    - ES. ottenere privilegi o autorizzazioni dal sistema
- *Attività*
  - Le attività che un attaccante deve svolgere per raggiungere uno o più sottogoal
    - Es. usare credenzialità (es. username e password) rubate per logarsi al sistema; spacciarsi per un diverso computer, o user, o device; invadere la rete con pacchetti malformati; ecc.

# Fasi di un attacco (2)

- *Eventi*
  - L'occorrenza di attacchi causati dalle attività
    - Es. ottenere un accesso improprio; sospendere il processamento di una richiesta; esaurire lo spazio di memoria; fermare un sistema o un programma
- *Conseguenze*
  - Le conseguenze del verificarsi di un evento
    - Es. incorrettezza dei estatti conto bancari; non disponibilità di un sistema a processare richieste di business; ecc.
- *Impatto*
  - Effetti di business
    - Es. offuscare la reputazione di una compagnia; perdita di guadagno; ecc.

# Esempio di attacco

**Attacker**




**Target**

- *Goal:*
  - –rubare denaro
- *Sottogoal:*
  - –ottenere accesso non autorizzato
- *Attività:*
  - –usare password rubata
- *Eventi:*
  - –controlli d'accesso violati
- *Conseguenze:*
  - –mancanza di bilancio
- *Impatto:*
  - –perdita di guadagno

# Modalità di attacco

**Come un attacker attacca un sistema?**

Il *come* dipende spesso dal *perché* e dalla *competenza* in materia di sicurezza

Per **rafforzare** la sicurezza di un'applicazione è importante domandarsi

Cosa può accadere e

Come può accadere

## Tipi di attacco

**In fase di**

- *Progettazione*

  Mentre si sta progettando l' applicazione

- *Implementazione*

  Mentre si sta scrivendo l' applicazione

- *Operazione*

  Dopo che l' applicazione è in produzione