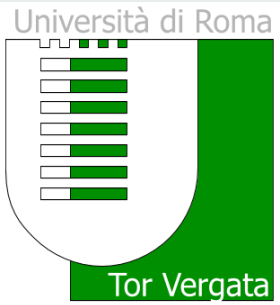


La sicurezza dell'hardware: problematiche e contromisure

Prof. Marco Ottavi
Dipartimento di Ingegneria Elettronica
Università degli Studi di Roma “Tor Vergata”

Panoramica

- Problema e definizioni
- Attendibilità dell’hardware (Hardware dependability)
- Fiducia nell’hardware (Hardware Trust)
- Vulnerabilità dell’hardware (Hardware Vulnerability)
- Esempi di contromisure

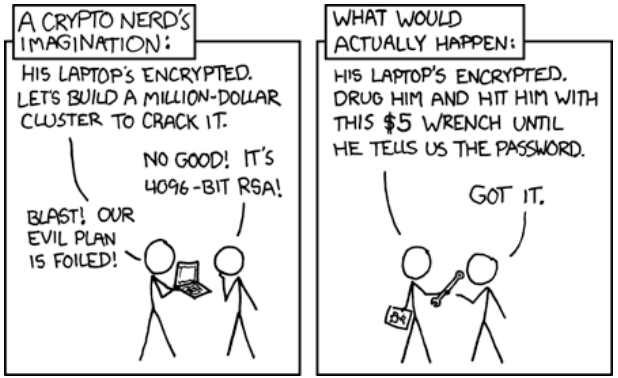
Attendibilità e sicurezza dell'hardware

- L' **attendibilità (dependability)** di un sistema rappresenta il livello di fiducia che funzioni come previsto e non fallirà per qualche motivo non intenzionale
  Questa definizione generale include qualità, affidabilità e sicurezza
  - **La qualità (quality)** rappresenta il livello di fiducia che un sistema funzioni correttamente di fronte al verificarsi di eventi casuali durante la produzione
  - **L'affidabilità (reliability)** rappresenta il livello di fiducia che un sistema funzionerà correttamente di fronte al verificarsi di eventi casuali durante il suo tempo di missione.
  - **La sicurezza (security)** rappresenta il livello di fiducia che un sistema funzionerà correttamente di fronte al verificarsi di eventi intenzionali sia in fase di produzione che di missione.
- Molte tecniche utilizzate per ottenere qualità e affidabilità sono applicabili anche alla sicurezza

Problema di sicurezza hardware

Gli esperti di sicurezza informatica hanno tradizionalmente presunto che l'hardware alla base dei sistemi informatici sia attendibile. Tuttavia tale presupposto non è più vero.

L'hardware non può essere considerato la radice della fiducia (root of trust)

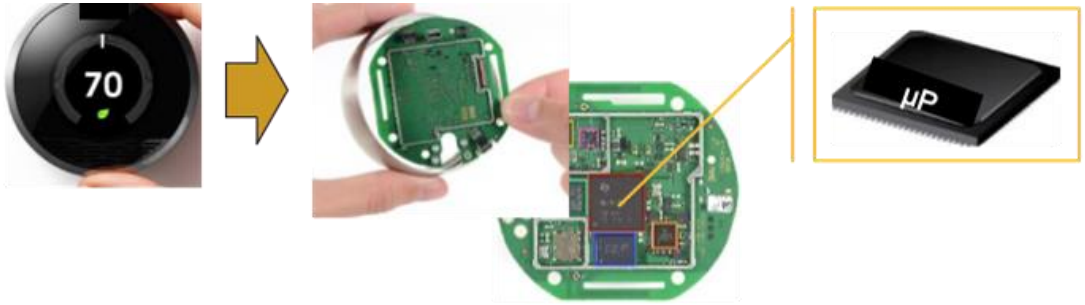


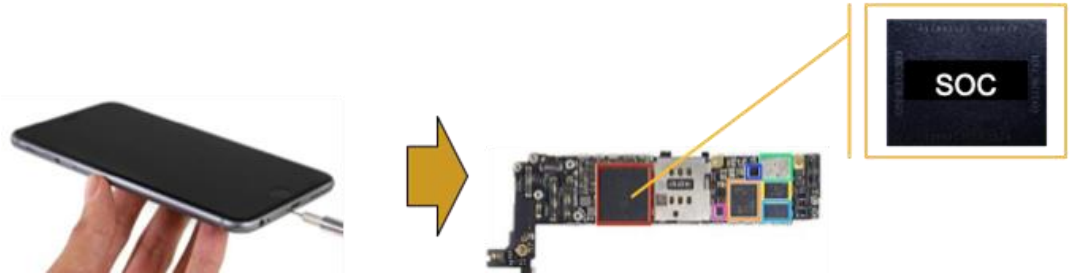
Vulnerabilità hardware e credito hardware

Due problemi principali con la assunzione root of trust

1) **Fiducia nell'hardware (Hardware Trust)**:
   a) Siamo sicuri che l'hardware sia stato realizzato da fonti attendibili?
   b) Sono presenti modifiche dannose indesiderate (Trojan)?
   c) Quali sono gli effetti dei trojan sulla funzionalità dell'hardware?
2) **Vulnerabilità dell'hardware (Hardware Vulnerability)**
   a) Esistono vulnerabilità che potrebbero essere sfruttate da un utente malintenzionato?
   b) Vulnerabilità Side Channel Attack
   c) Attacchi di esecuzione transitori (ad esempio Spectre, Meltdown)

Cos'è l'Hardware?





Sistema elettronico

Hardware di sistema: funge da **"root-of-trust"** : PCB → IC (SoC | µP)

Definizioni

- **Aspetti di sicurezza informatica**
  - **Riservatezza** : i relativi asset sono accessibili solo da soggetti autorizzati
  - **Integrità** : il bene viene modificato solo da soggetti autorizzati
  - **Disponibilità** : il bene è accessibile ai soggetti autorizzati in tempi opportuni
- **Minaccia** : insieme di circostanze che possono potenzialmente causare perdite o danni
- **Vulnerabilità** : debolezza nel sistema sicuro
- **Attacco** : l'atto di un essere umano che sfrutta la vulnerabilità nel sistema

Problemi di hardware security

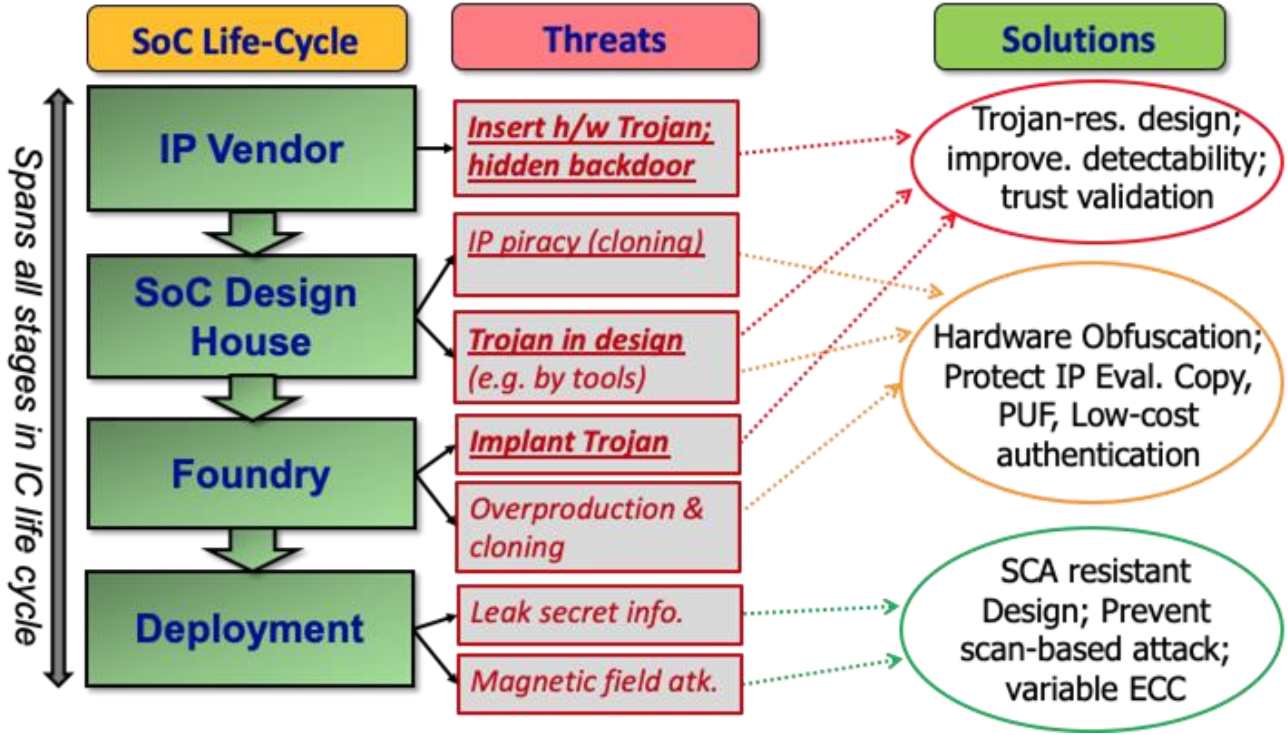
L'hardware security può essere influenzata da diverse minacce

- **Trojan Horse** (implementati a diversi livelli di progettazione) → **fiducia**

- **Attacchi fisici** (ad es. attacchi di canali laterali; vulnerabilità microarchitetturali) →**vulnerabilità**

- **Pirateria IP** (clonazione di IP)
- **Pirateria e contraffazione IC** (clonazione, sovrapproduzione)
- **Manomissione** (ad es. modifiche del flusso di bit FPGA)
- **Reverse Engineering**

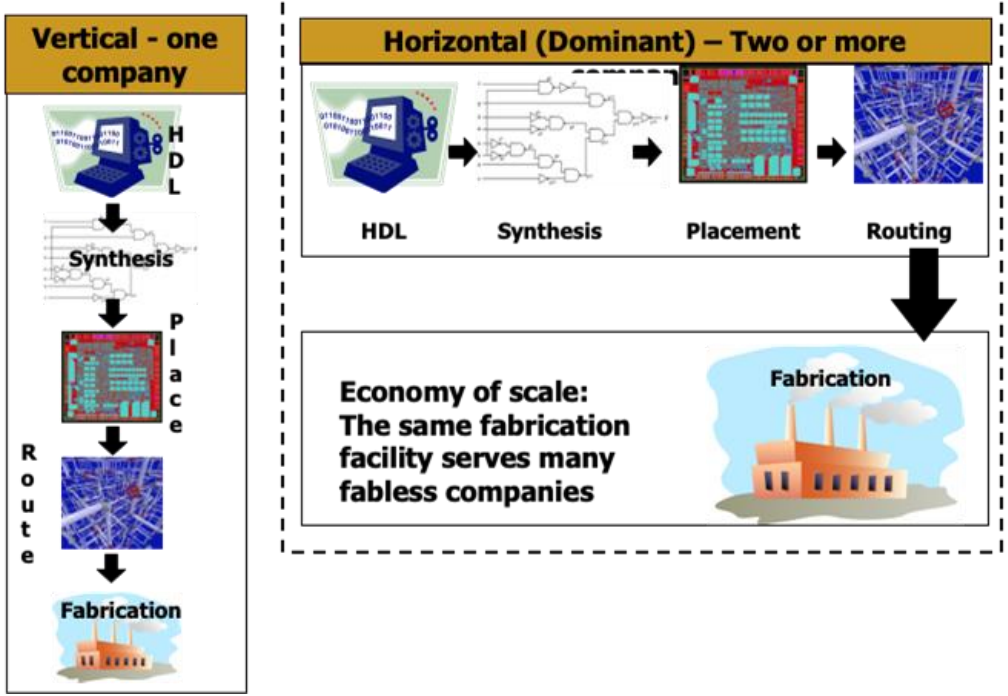
Minacce hardware



Le minacce possono presentarsi a diversi livelli di sviluppo e dell'uso di un SoC.

Le contromisure devono essere prese in considerazione durante tutto il processo

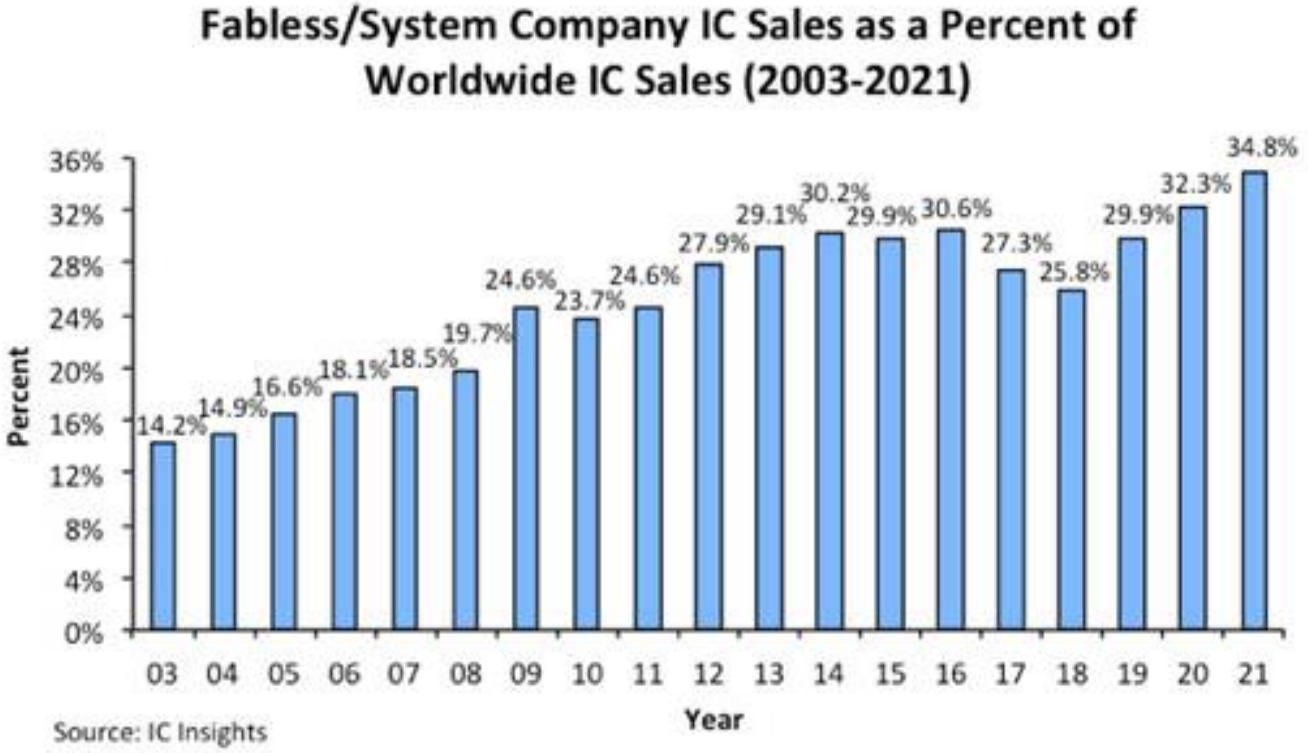
Settore VLSI: tendenze del modello di business



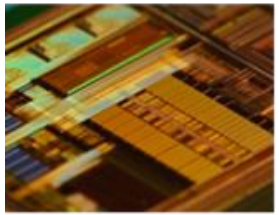
Modello Verticale: tutto sviluppo interno → costi elevati, basse economie di scala

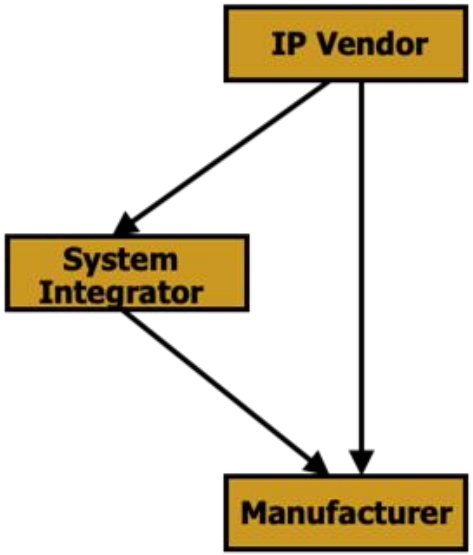
Modello orizzontale : più aziende coinvolte → minori costi, economia di scala

Senza fabbrica (fabless) 34.8% produzione IC nel mondo.



Possiamo fidarci dell'hardware?





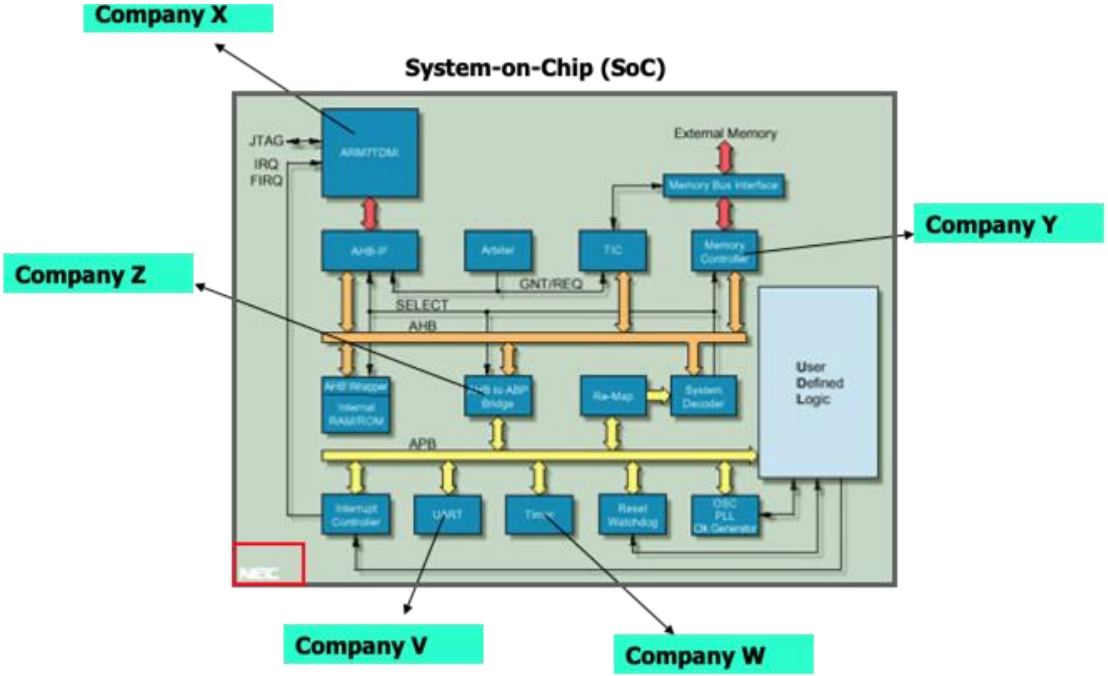
Any of these steps can be untrusted

IP: proprietà intellettuali a volte **fornite da fornitori di terze parti**

System Integrator **combina diversi IP** in un progetto di chip

Il produttore fabbrica i chip **in base al progetto ricevuto**

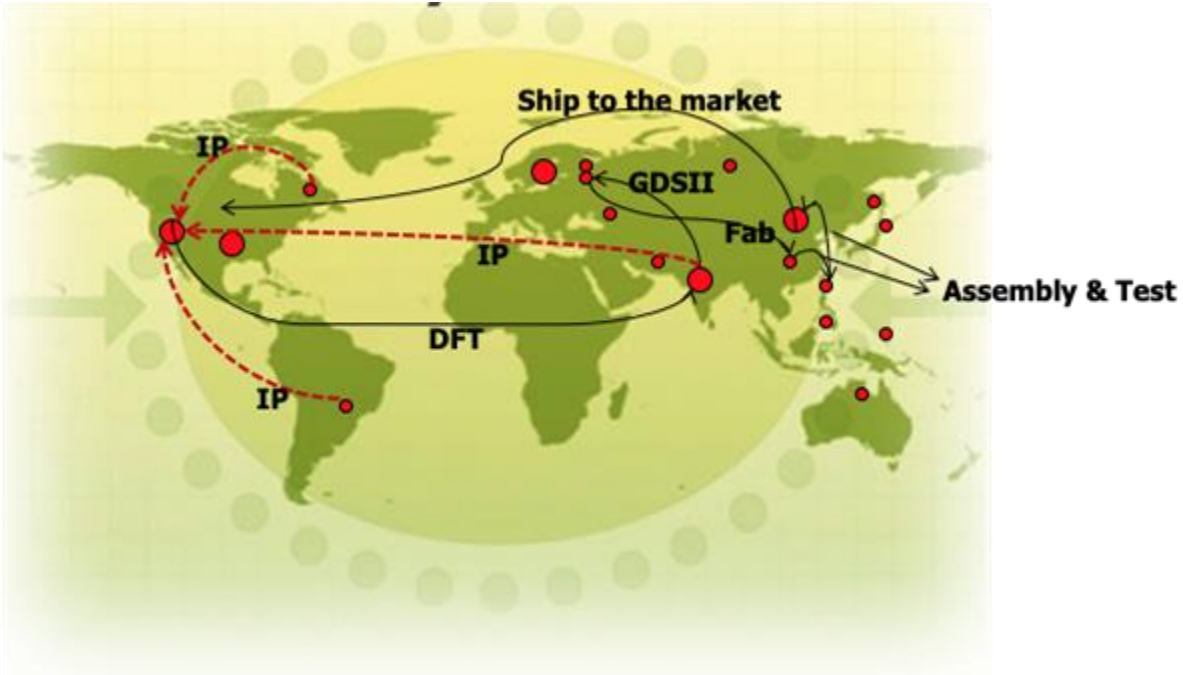
Problemi con l'utilizzo di IP di terze parti



Un moderno **System on Chip (SoC)** potrebbe essere composto da IP forniti da diversi fornitori

Queste aziende si trovano in tutto il mondo Non c'è alcun controllo sul processo di progettazione

Dove vengono sviluppati i chip moderni?



In tutto il mondo

Le fasi di progettazione, produzione, collaudo, packaging sono un'attività veramente globale

Ciò solleva potenziali problemi di fiducia

Avversari

- Singolo, gruppo o governi stranieri
  - **Implementazione di cavalli di Troia**
  - **Spiare sfruttando le vulnerabilità IC**
  - Pirateria delle IP – uso illegale di IP
  - Inserimento di backdoor o circuiti dannosi
  - Reverse engineeting dei circuiti integrati
- Integratori di sistema
  - Pirateria delle IP
- Strutture di fabbricazione
  - Pirateria delle IP
  - Pirateria dei circuiti integrati
- Parti contraffatte
  - Riciclo, clonato, ecc.

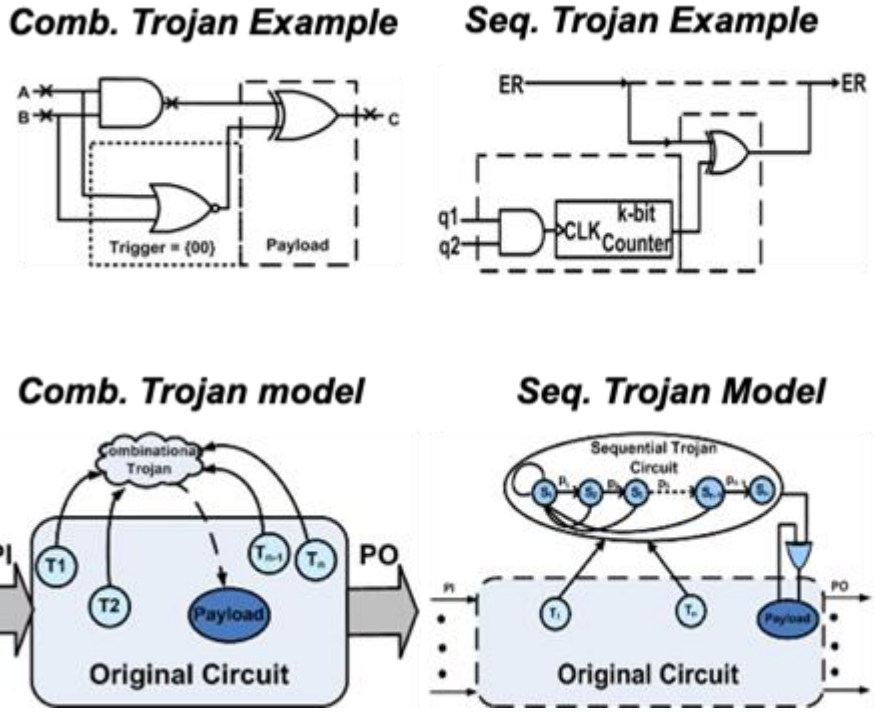
Quindi, su cosa ci concentriamo?

- **Problemi di fiducia nell'hardware**
  - Trojan hardware
  - Approcci di classificazione e rilevamento HWT
- **Problemi di vulnerabilità hardware**
  - Attacchi di canale laterale
  - Attacchi che sfruttano le vulnerabilità dell'implementazione hardware
  - Contromisure

Fiducia nell'hardware: cos'è un trojan hardware?

- **Trojan hardware:**
  - Un'aggiunta o una modifica dannosa agli elementi del circuito esistente.
- **Cosa possono fare i trojan hardware?**
  - Cambia la funzionalità
  - Ridurre l'affidabilità
  - Perdi informazioni preziose
- **Applicazioni che potrebbero essere bersagli di aggressori**
  - Applicazioni militari
  - Applicazioni aerospaziali
  - Applicazioni critiche per la sicurezza civile
  - Applicazioni finanziarie
  - Sicurezza dei trasporti
  - Dispositivi IoT
  - Dispositivi commerciali
  - Ecc...

Esempi e modelli di Trojan HW

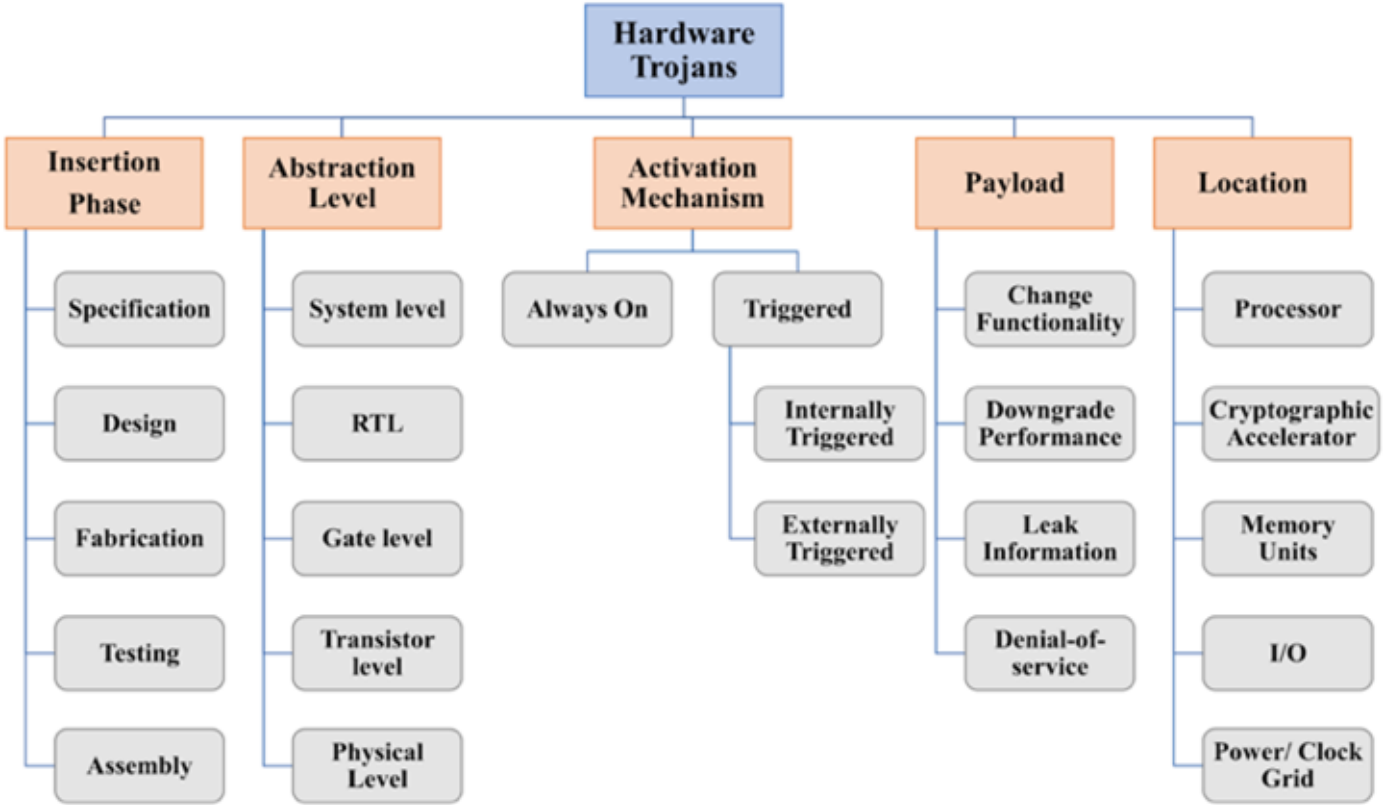


Due semplici esempi di HWT inserito in un circuito digitale

Combinatorio: il Trojan capovolge l'uscita solo se gli ingressi sono in 00

Sequenziale: il Trojan è un contatore cronometrato dai valori di q1 e q2 → capovolge l'uscita a un certo valore

Tassonomia Trojan



I trojan possono essere classificati in base a diverse metriche

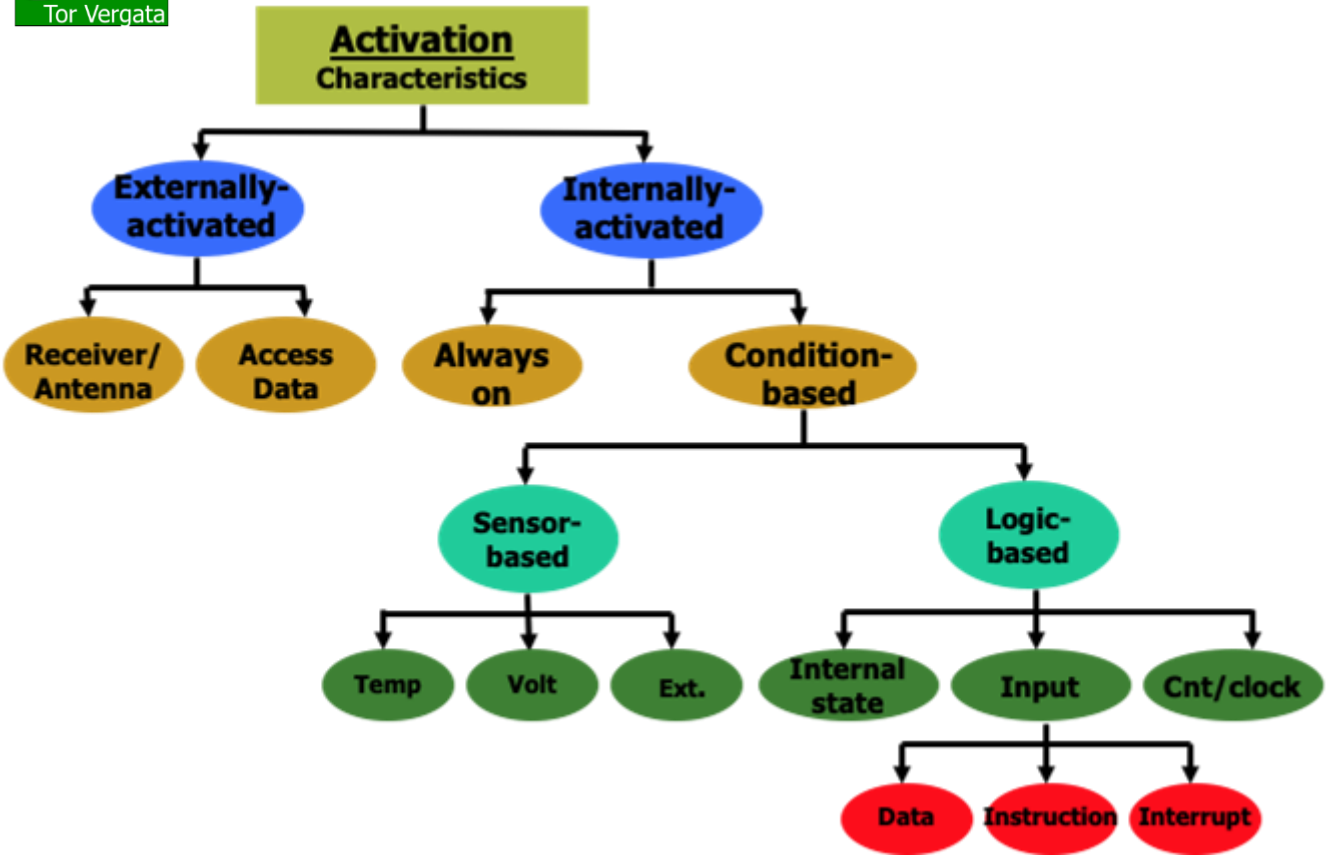
Fase di inserimento

Livello di astrazione

Meccanismo di attivazione

Carico utile

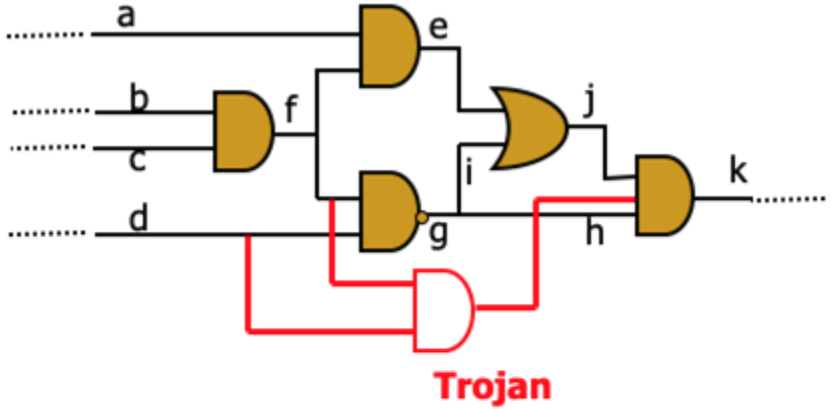
Classificazione di attivazione dei trojan



L'attivazione dei trojan può basarsi su diverse metriche

Le contromisure devono tener conto della presenza di diversi approcci di attivazione!

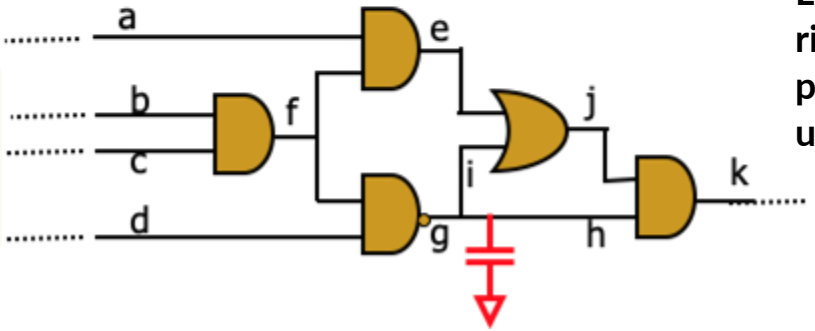
Esempio di due payload



Modify Function

Modify Specification:

Noise, Delay and Temperature



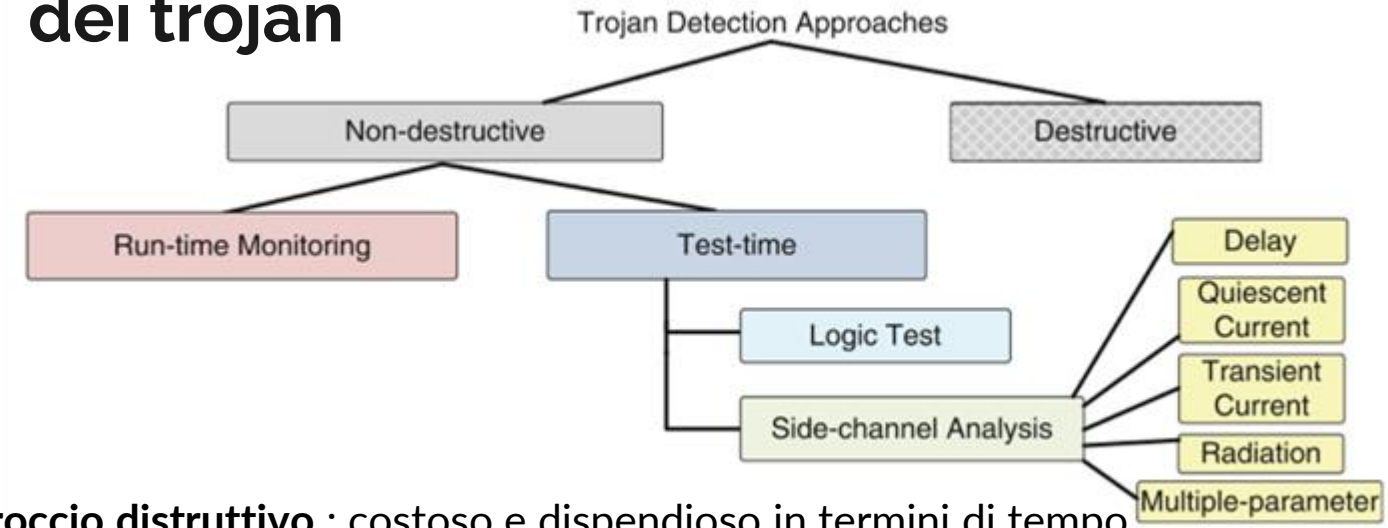
Il payload del trojan definisce la sua capacità di modificare il comportamento normale

Due esempi

Trojan modifica l'output controllando l'output quando d o f sono =0

Le modifiche al trojan ritardano l'aumento del percorso critico aggiungendo un carico capacitivo sul nodo g

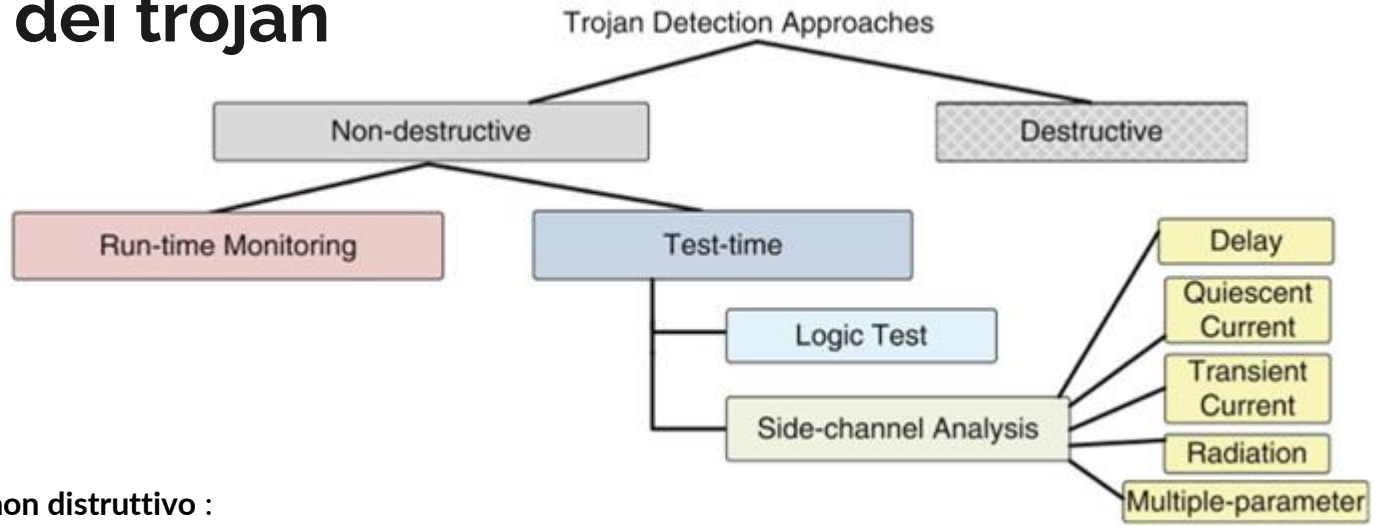
Classificazione degli approcci di rilevamento dei trojan



Approccio distruttivo : costoso e dispendioso in termini di tempo

- Reverse engineering per estrarre immagini strato per strato utilizzando il delayering e il microscopio elettronico a scansione
- Identifica transistor, gate ed elementi di instradamento utilizzando un approccio basato sulla corrispondenza dei modelli: **richiede IC/layout dorati**

Classificazione degli approcci di rilevamento dei trojan



Approccio non distruttivo :

- **Monitoraggio in fase di esecuzione** : monitora il comportamento anomalo durante la fase di esecuzione
  - Sfrutta la ridondanza preesistente nel circuito
  - Confronta i risultati e seleziona una parte attendibile per evitare una parte infetta del circuito.
- **Autenticazione durante il test** : rileva i trojan per tutta la durata del test.
  - Approcci basati sui test logici
  - Approcci basati sull'analisi del canale laterale

Benchmark dei trojan hardware

- È necessario
  - Fornire una linea di base per l'esame dei diversi metodi sviluppati
  - Stabilire una solida base per la resistenza di ogni istanza di benchmark
  - Contribuire ad aumentare la riproducibilità dei risultati da parte di altri che intendono utilizzare determinate metodologie nel loro flusso di progettazione
- Vedere il sito Web **Trust-Hub supportato da NSF** (www.trust-hub.org)
  - Tassonomia completa dei Trojan
  - Sono disponibili più di 120 benchmark di fiducia progettati a diversi livelli di astrazione, attivati in diversi modi e con diversi meccanismi di effetto
  - Più di 300 pubblicazioni hanno utilizzato questi benchmark

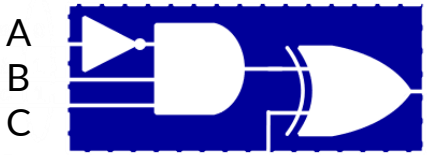
Monitoraggio durante l'esecuzione

Possiamo aggiungere ridondanze specifiche al progetto per rilevare comportamenti dannosi?

- In fase di progettazione includere monitor (più avanti)
- Può essere utilizzato per il rilevamento di errori casuali e intezionali
  - Rilevamento di trojan IP soft/firm in FPGA
  - Rilevamento hard IP Trojan su SoC
- Sulle architetture a microprocessore opportunità sfruttando la ISA RISC-V open standard

Approccio con test logico

- **L'approccio di test logico** si concentra sulla generazione di vettori di test per
  - Attivazione di un circuito di Troia
  - Osservando il suo effetto dannoso sul carico utile agli output primari
  - Sono applicabili sia vettori di test funzionali che strutturali.
- **Pro e contro:**
  - **Pro** :
    - Semplice e facile da differenziare
  - **Contro** :
    - La difficoltà nell'eccitare o nell'osservare nodi a bassa controllabilità oa bassa osservabilità.
    - I trojan inseriti intenzionalmente vengono attivati in rare condizioni. (ad esempio, Trojan sequenziali)
    - Non può attivare trojan attivati esternamente e può solo osservare trojan funzionali.

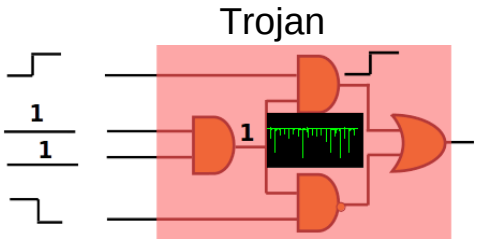


Esempio: trova il vettore per attivare la condizione di attivazione del trojan
(a=0, b=1, c=1)

Approccio con analisi del canale laterale

- Tutte le analisi del canale laterale si basano sull'osservazione dell'effetto di un Trojan inserito su un parametro fisico come
  - **IDDQ** : I cancelli aggiuntivi consumeranno corrente di dispersione.
  - **IDDT:** le attività extra di commutazione consumeranno più energia dinamica.
  - **Ritardo del percorso** : porte e capacità aggiuntive aumenteranno il ritardo del percorso.
  - **EM:** Radiazione elettromagnetica dovuta all'attività di commutazione

- **Pro e contro**
  - **Pro:** è efficace per i Trojan che non causano malfunzionamenti osservabili nei circuiti.
  - **Svantaggi** : grandi variazioni di processo nelle moderne tecnologie nanometriche e il rumore di misurazione possono mascherare l'effetto dei circuiti Tojan, specialmente per i piccoli Trojan.



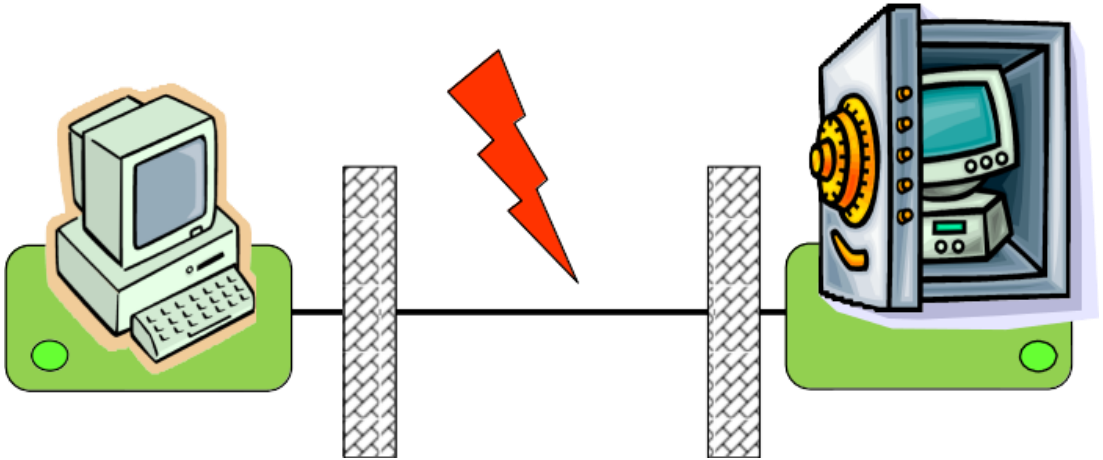
Esempio: rilevare la potenza, il ritardo, le impronte digitali di emissione EM del Trojan inserito

HA BISOGNO DI GOLDEN CHIP

E la Vulnerabilità dell'Hardware?

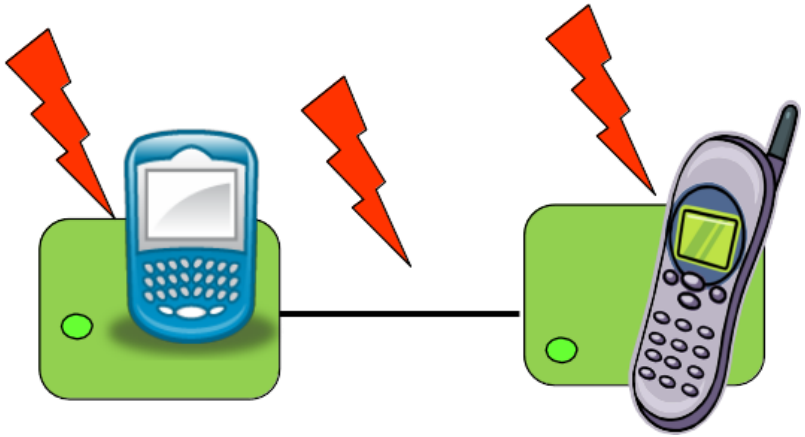
- La crittografia classica vede i problemi sicuri con **astrazioni matematiche**
- La crittoanalisi classica ha avuto un grande successo e promesse
  - Analizzare e quantificare la resilienza degli algoritmi crittografici contro gli attacchi
- Recentemente, molti dei protocolli di sicurezza sono stati attaccati tramite **attacchi fisici (noti anche come Side Channel Attacks)**
  - Sfruttare i punti deboli nell'implementazione hardware del sistema crittografico mirato a recuperare i parametri segreti

Modello tradizionale (vista semplificata)



- Attacco al canale **tra parti comunicanti**
- Crittografia e operazioni crittografiche in **scatole nere**
- Protezione mediante potenti algoritmi e protocolli matematici
- Computazionalmente sicuro

Cosa sta succedendo adesso (IoT e Embedded)



- Nuovo modello (vista semplificata):
  - Attacchi al **canale e agli endpoint**
  - Crittografia e operazioni crittografiche in **scatole grigie**
  - Protezione mediante potenti algoritmi e protocolli matematici
  - **Protezione mediante implementazione sicura**

- **implementazioni sicure non solo algoritmi**

Dove sono i punti deboli?

Un sistema è sicuro quanto il suo anello più debole

Cos'è l'attacco del canale laterale?

- Gli attacchi side-channel mirano a **input e output del canale laterale,** aggirando la forza teorica degli algoritmi crittografici → utilizzano informazioni aggiuntive per violare la protezione crittografica

- Cinque emissioni del canale laterale comunemente sfruttate:
  - **Consumo di energia**
  - Elettromagnetico
  - Ottico
  - **Tempi e ritardi**
  - Acustico

Perdita di informazioni sul canale laterale

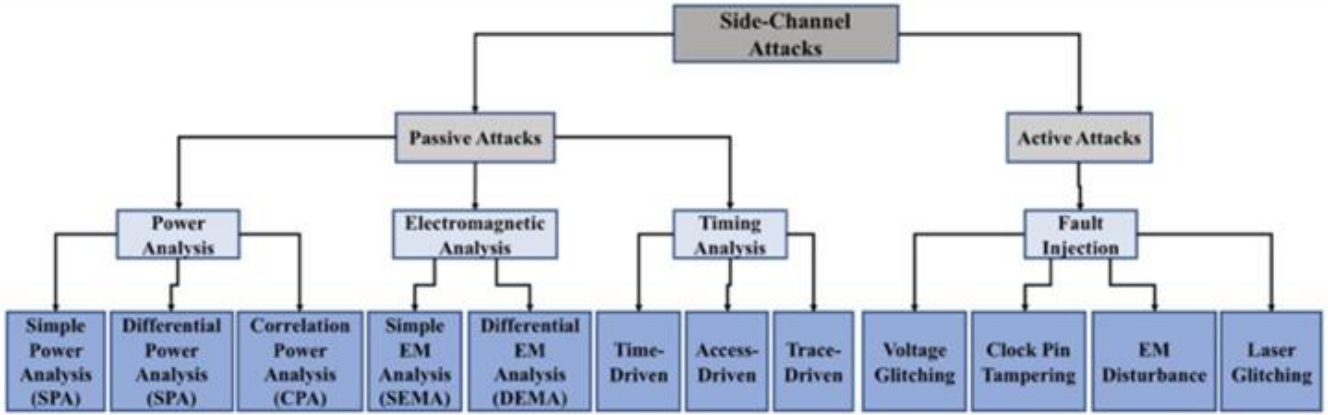
Physical attacks ≠ Cryptanalysis

(gray box, physics) (black box, maths)

- Does not tackle the algorithm's math

Osserva **le quantità fisiche nelle vicinanze del dispositivo** e utilizza informazioni aggiuntive durante la crittoanalisi

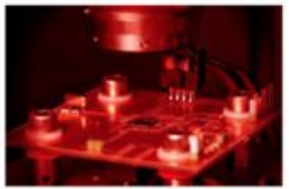
Tassonomia degli attacchi Side Channel



Attacchi passivi : osservare il comportamento del dispositivo per dedurre informazioni sul segreto
Attacchi attivi: operare fisicamente sul dispositivo per raccogliere informazioni sul segreto (ad es. Fault injection o microprobing)
Attacchi attivi vs. passivi:
gli attacchi attivi sfruttano gli input del canale laterale Attacchi passivi sfruttare le uscite del canale laterale

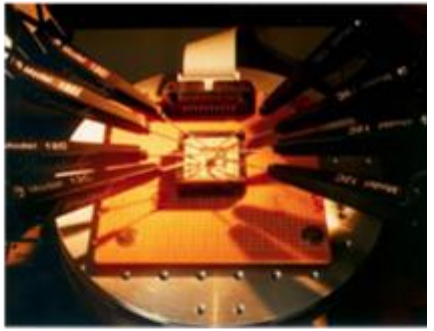
Quali sono i canali laterali utilizzati?

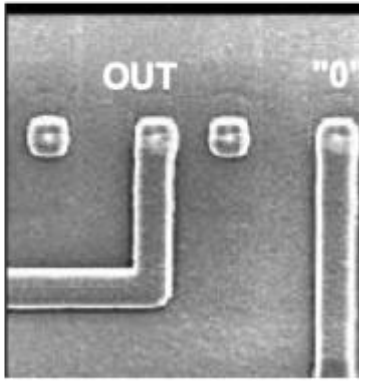
- **Passivo** :
  - Tempistica (fusione dello spettro)
    - Tempo di esecuzione complessivo o “locale”.
  - Potenza, radiazione elettromagnetica (EM).
    - Tecnologia CMOS predominante
    - Consumo energetico dinamico
    - La corrente elettrica induce un campo EM
  - Più esotico ma dimostrato di essere pratico
    - Suono, temperatura, …
- **Invasivo** : emissioni fotoniche



Attacchi invasivi

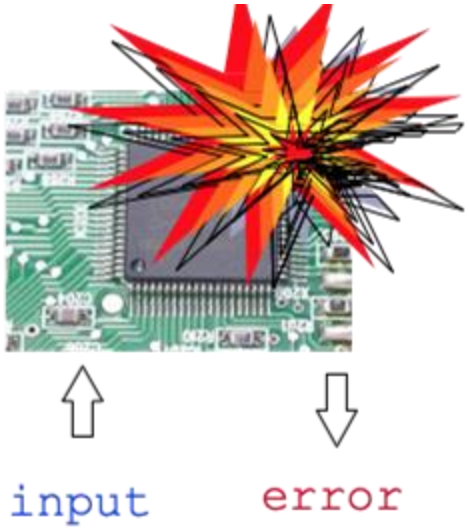
- **Passivo: micro-probing**
  - Sonda i bus interni con un ago molto sottile
  - Leggere direttamente i dati dal bus o dalle singole celle
  - Diversi aghi contemporaneamente
- **Attivo : modifica del circuito**
  - Connettere o disconnettere il meccanismo di sicurezza
    - Scollega i sensori di sicurezza
    - RNG bloccato a un valore fisso
    - Ricostruzione di fusibili bruciati
  - Taglia o incolla tracce con laser o raggio ionico focalizzato





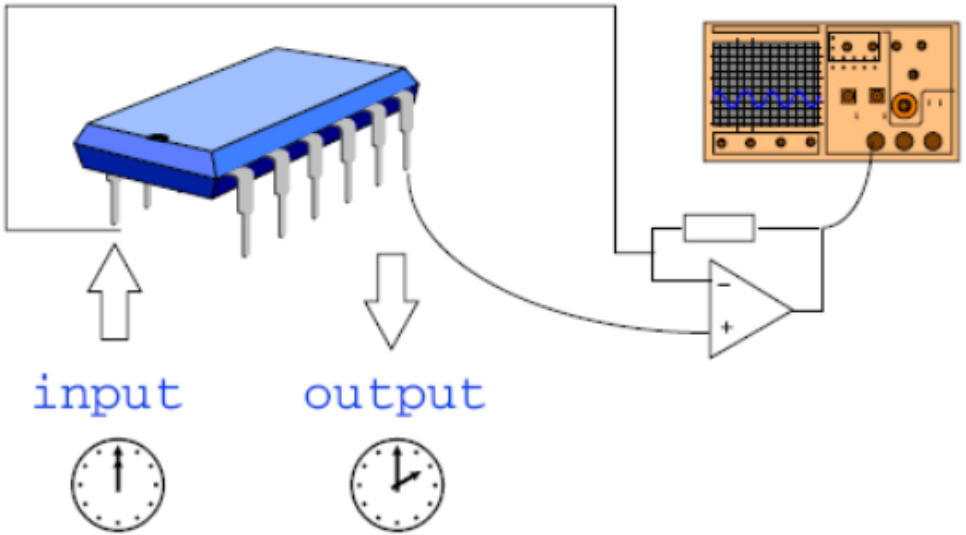
Attacchi di iniezione di guasti

- Non (semi)invasivo: applica una combinazione di condizioni ambientali non considerate
  - Vcc
  - Problema
  - Orologio
  - Temperatura
  - UV
  - Leggero
  - Raggi X
- E bypassare i meccanismi di sicurezza o dedurre i segreti

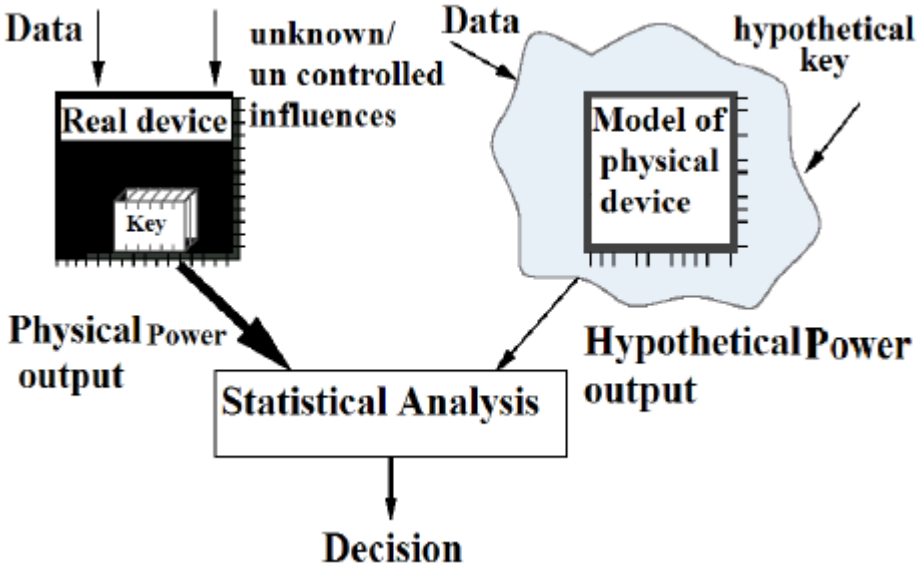


Esempio: Power Attack

- Measure the circuit's processing time and current consumption to infer what is going on inside it.



Esempio: Power Attack (continua)



Correlare il consumo energetico effettivo con un modello per dedurre la chiave

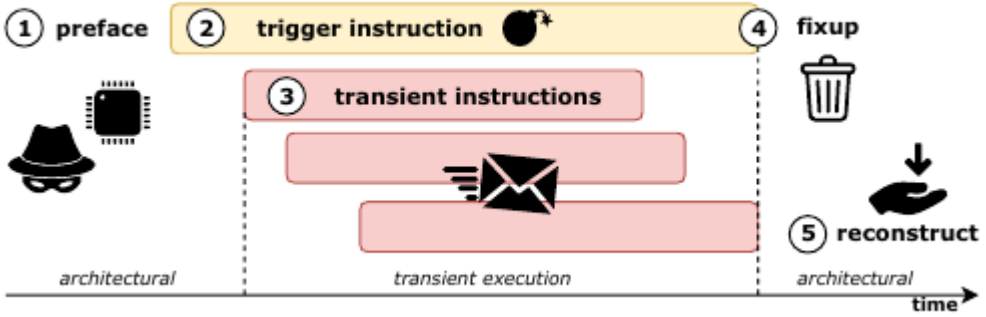
Attacchi su tempi di esecuzione

- Sfrutta le vulnerabilità architetturali dei processori moderni
- Esempi tipici: **Spectre e Meltdown**
  - **Spettro:** consente di leggere posizioni arbitrarie nella memoria allocata di un programma.
  - **Meltdown** consente a un processo di leggere tutta la memoria in un dato sistema.
- **Spectre e Meltdown** rappresentano esempi di **attacchi di "esecuzione transitoria"** , che si basano su difetti di progettazione hardware nell'implementazione di esecuzione speculativa , pipeline di istruzioni ed esecuzione fuori ordine nelle moderne CPU.
- Le implementazioni di questi variano tra i produttori di CPU e le microarchitetture; di conseguenza, non tutte le varianti di Spectre e Meltdown sono sfruttabili su tutte le microarchitetture.

Attacco di esecuzione transitorio

Un tipico Transient Execution Attack è composto da 5 fasi

1. Preparare la microarchitettura,
2. Eseguire un'istruzione di trigger,
3. Le istruzioni transitorie codificano i dati non autorizzati attraverso un canale segreto della microarchitettura,
4. La CPU ritira l'istruzione trigger e cancella le istruzioni transitorie,
5. Ricostruisci il segreto dallo stato della microarchitettura.

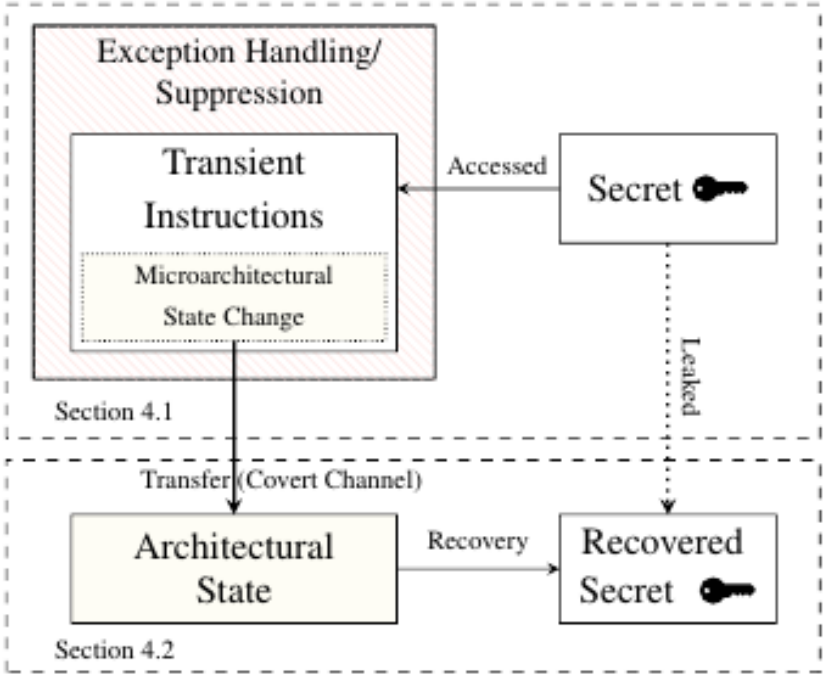


Meltdown sfrutta out of order execution

Spectre sfrutta la branch preditcion

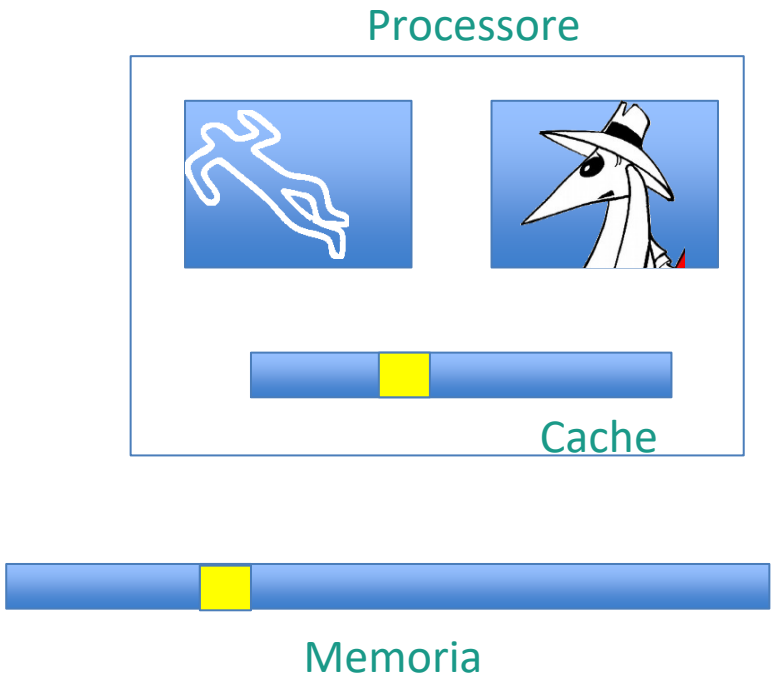
Attacco di esecuzione transitorio (continua)

- Esempio Meltdown utilizza Flush + Reload per recuperare i dati trapelati
  - Configurazione: svuota la cache
  - Transmit: l'esecuzione Out of Order forza la speculazione che dipende dal segreto → carica nei dati della cache un indirizzo che dipende dal segreto
  - Ricevi: misura i tempi della cache



FLUSH+RELOAD

- FLUSH linea di cache
- Aspettare un po’
- Misurare il tempo per RICARICARE la linea
  - lento-> nessun accesso
  - veloce -> accesso
- Ripetere



Yarom, Yuval, and Katrina Falkner. "FLUSH+ RELOAD: a high resolution, low noise, L3 cache side-channel attack." *23rd {USENIX} Security Symposium ({USENIX} Security 14)*. 2014.

Contromisure contro l'SCA

- **Hiding;** Ridurre l'SNR aumentando il rumore o riducendo l'ampiezza del segnale
  - Generatori di rumore, montanti logici bilanciati (dual rail), logica asincrona, design a bassa potenza, schermatura
- **Mascheramento:** rimuovere la correlazione funzionale tra i dati di input e le emissioni del canale laterale dai blocchi funzionali intermedi
  - modificare le esecuzioni interne
- **Design Partitioning:** regione separata del chip che opera su testo in chiaro dalle regioni che operano su testo cifrato

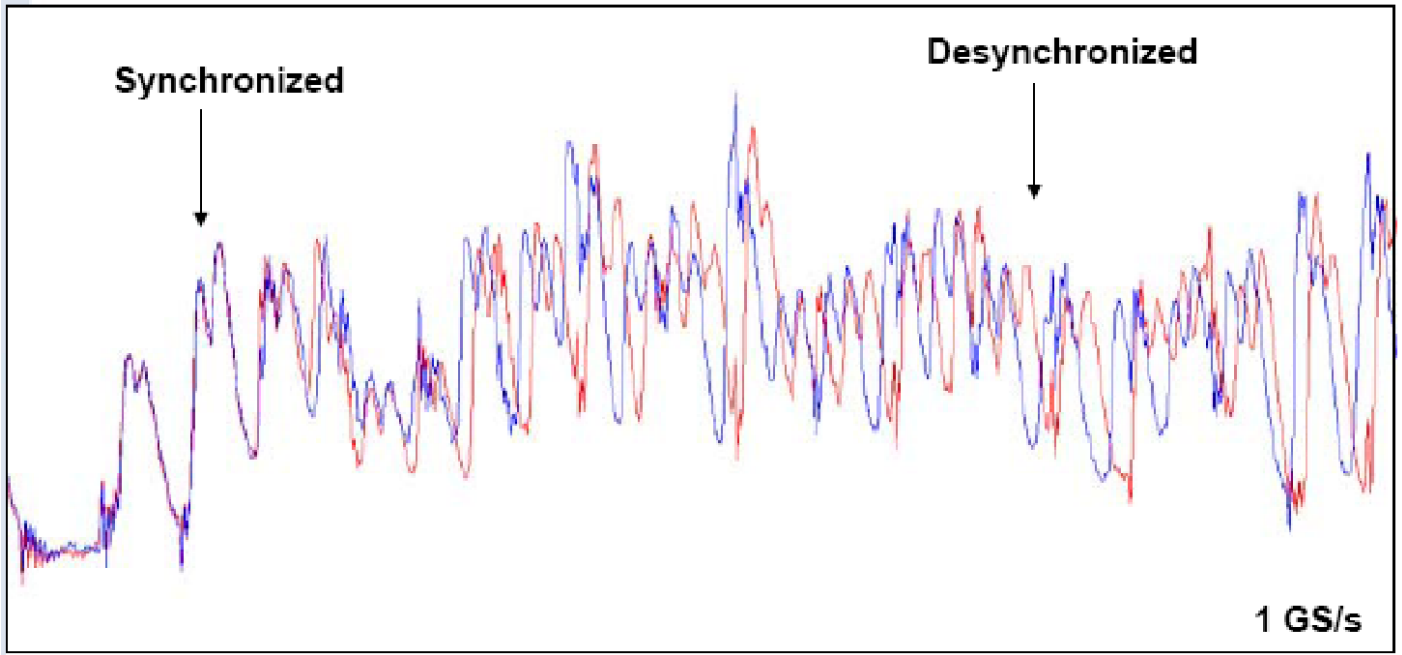
Contromisure contro Power Attack

L'idea di base è evitare che la funzione di perdita rifletta veramente la chiave segreta.

- Curve di potenza decorrelate dai dati:
  - **in hardware:** current scramblers (rumore additivo)
  - **in software:** data whitening
- Desincronizzare le N tracce (disallineamento curve)
  - ritardi casuali del software
  - ordine casuale del software (es. SBOX in ordine casuale)
  - stati di attesa hardware (cicli fittizi aggiunti dalla CPU)
  - hardware clock interno instabile (spostamento di fase)
- DPA è potente, generico (per molti algoritmi e resistente agli errori di modellazione)
  - Ma esistono contromisure

Anti-DPA

spostamento di fase del clock



Contromisure architettoniche

- E gli attacchi sul tempo di esecuzione?
  - si basano su vulnerabilità della microarchitettura che consentono all'attaccante di dedurre il segreto monitorando i ritardi di esecuzione
  - attacchi basati sull'esecuzione → basati sulla ripetizione, **ad esempio Flush+Reload,** mostrano un'impronta digitale
- Aggiungere un controllo online per analizzare e rilevare potenziali attività dannose
  - sfruttare ISA aperti come RISC-V (più nei prossimi lucidi)

Contromisure basate su strutture dati probabilistiche

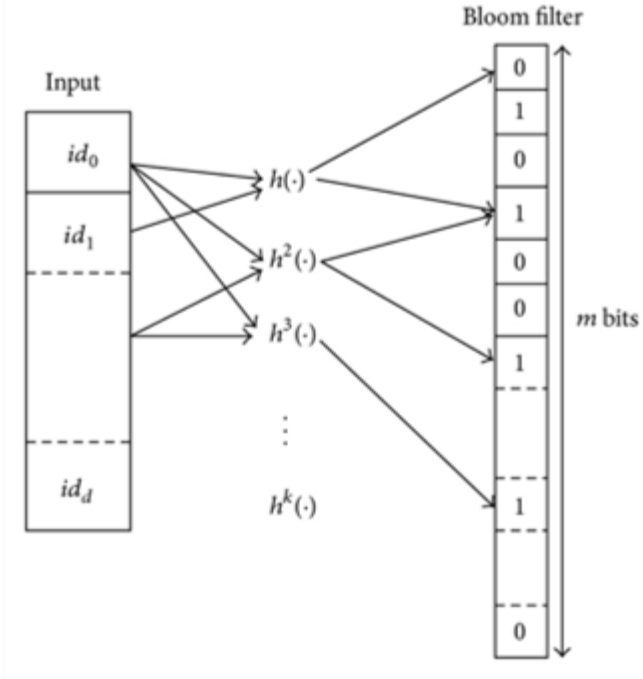
- Le strutture di dati probabilistiche, note anche come sketches, sono ampiamente utilizzate nelle reti per monitorare grandi quantità di traffico di dati
- Consentono di monitorare grandi set di dati attraverso strutture hardware compatte
- Basato su tecniche probabilistiche
- Ci concentriamo sull'uso di due sketch per la sicurezza hardware
  - Filtri Bloom → controllo approssimativo dell'appartenenza
  - Count-min sketeches→ stima della frequenza
- Sfrutta un'architettura ISA aperta (RISC-V) per inserire queste strutture leggere all'interno di un core
  - I filtri Bloom possono essere utilizzati per rilevare trojan hardware nelle memorie
  - Count Min Sketches può essere utilizzato per rilevare attacchi di temporizzazione del canale laterale

Panoramica dei filtri Bloom

- Un filtro Bloom è un bit array di m bit, inizialmente impostato su 0.
  - k diverse funzioni hash, ognuna delle quali mappa o esegue l'hashing di un elemento impostato su una delle m posizioni dell'array,
- Un filtro Bloom fornisce due funzionalità: Aggiungi elemento e Elemento query
- Una query su un elemento nel BF ha le seguenti proprietà
  - Se l'elemento è stato aggiunto in precedenza verrà trovato di sicuro → **nessun falso negativo**
  - Se l'elemento non è stato precedentemente aggiunto può essere trovato erroneamente presente → **probabilità di falso positivo**

Panoramica dei filtri Bloom (continua)

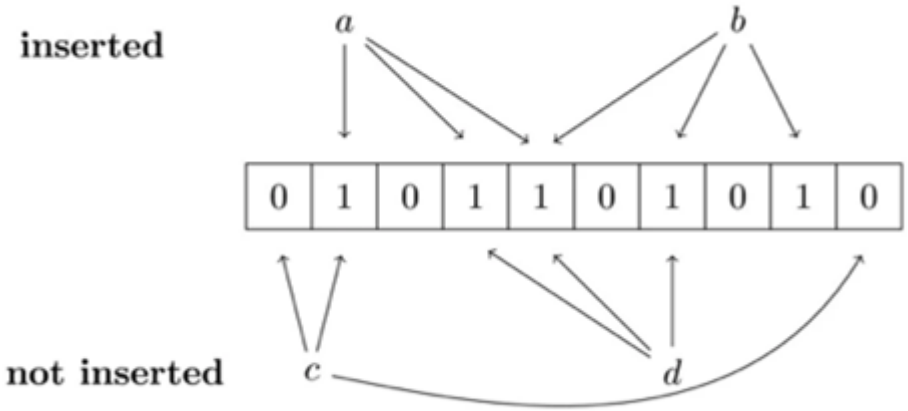
- **Aggiungi elemento** : l'elemento viene inviato a ciascuna delle k funzioni hash per ottenere k posizioni dell'array. Impostare i bit in tutte queste posizioni su 1.



Panoramica dei filtri Bloom (continua)

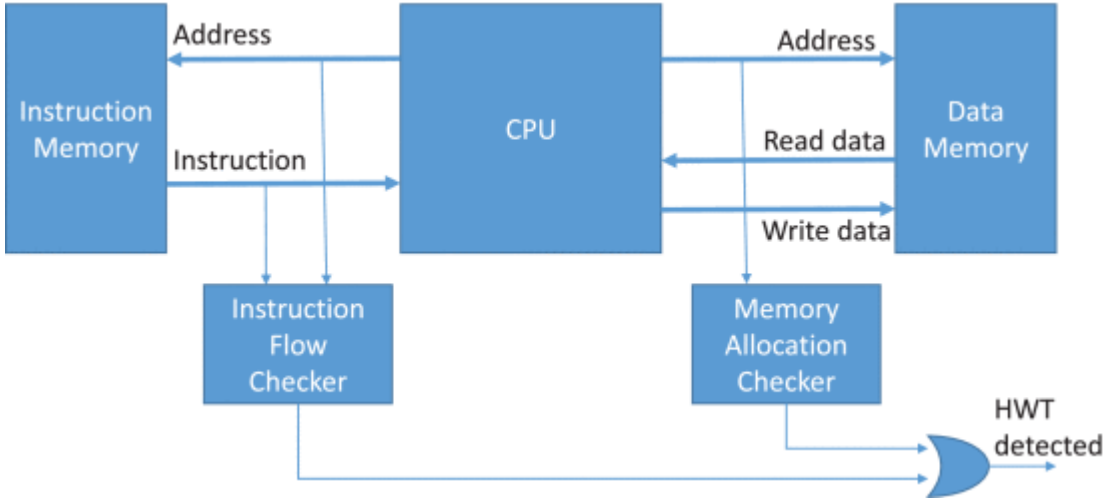
- **Elemento di query** , ovvero verifica se si trova nell'insieme di elementi aggiunti in precedenza: l'elemento viene inviato a ciascuna delle k funzioni hash per ottenere k posizioni dell'array.
  - Se uno qualsiasi dei bit in queste posizioni è 0, l'elemento sicuramente non è nell'insieme;
  - Se tutti sono 1, l'elemento è nell'insieme oppure i bit sono stati casualmente impostati a 1 durante l'inserimento di altri elementi, risultando in un **falso positivo.**
  - **La probabilità di falsi positivi dipende da m,n,k**

$$\approx \left(1 - e^{-kn/m}\right)^k.$$



Applicazione dei filtri Bloom al rilevamento dei trojan hardware

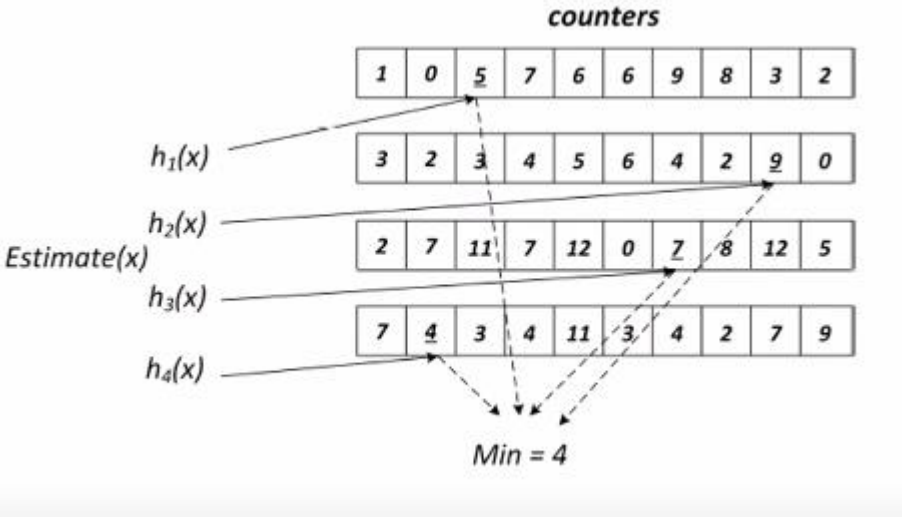
- Le pedine sono BF aggiunti per monitorare sia il flusso di istruzioni che l'allocazione della memoria.
- Esempio di controllo del flusso di istruzioni
  - Inizialmente l'IFC viene caricato con tutte le coppie (indirizzo, istruzione) del programma
  - In fase di esecuzione l'IFC controlla se l'istruzione recuperata corrisponde al programma
  - Se un trojan ha modificato la memoria per indurre codice dannoso o DOS, questo verrà rilevato
  - l'IFC è compatto e molto più piccolo di una replica della memoria delle istruzioni



A. Bolat, L. Cassano, P. Reviriego, O. Ergin e M. Ottavi, "A Microprocessor Protection Architecture against Hardware Trojans in Memories", 2020 *15th Design & Technology of Integrated Systems in Nanoscale Era (DTIS)* , Marrakech, Marocco , 2020, pp. 1-6, doi: 10.1109/DTIS48698.2020.9080961.

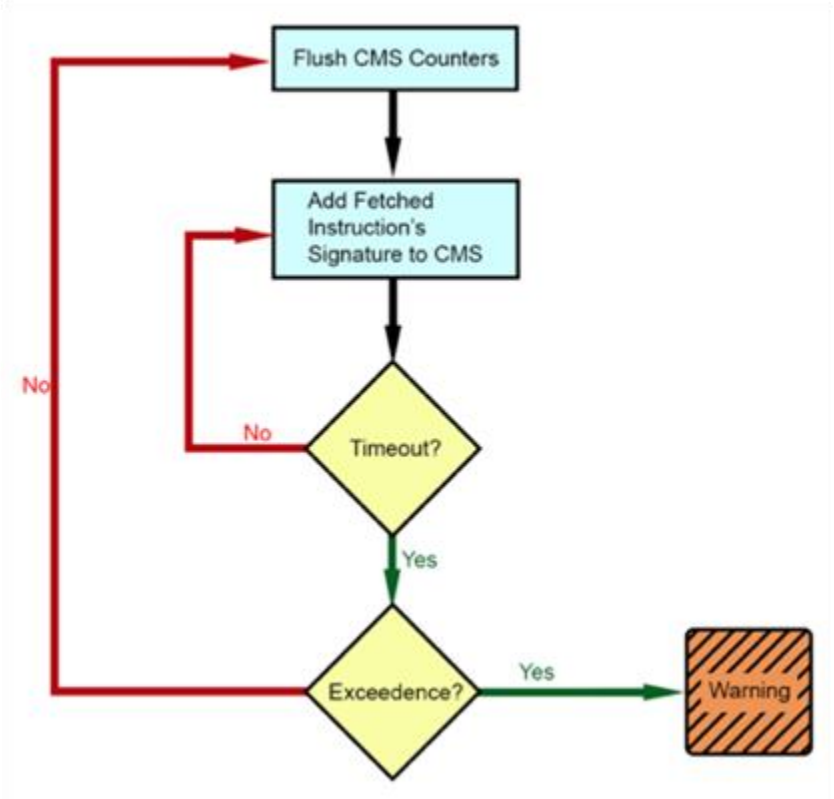
Panoramica di Count-Min Sketches (CMS)

- Utilizzato per rilevare la frequenza di modelli specifici.
- Implementazione simile ai filtri Bloom con la differenza che gli hash puntano a una serie di contatori
  - **Aggiungi ed elemento** → aumenta i contatori puntati dalle funzioni hash
  - **Controlla il numero di occorrenze** di un elemento → prendi il valore minimo dei contatori puntati dagli hash
- Un CMS non sottovaluta mai ma può sovrastimare a causa di collisioni di hash



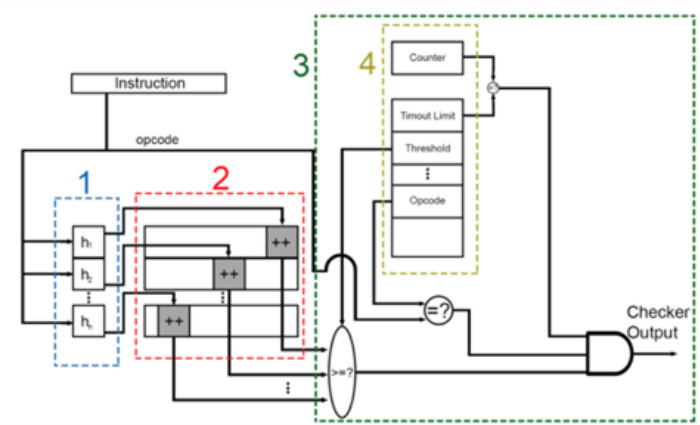
Applicazione di CMS alla rilevazione di Timing Side Channel Attack

- Il checker monitora il tasso di occorrenza delle istruzioni eseguite dal core
- L'obiettivo è trovare il verificarsi di attività sospette di attacchi di canali laterali nascosti
- Tipicamente queste attività utilizzano un numero anomalo di istruzioni



Applicazione di CMS alla rilevazione di Timing Side Channel Attack

- Il checker è composto da un CMS e da un ulteriore checker che analizza le sequenze sospette
- Quando un'istruzione viene recuperata, viene aggiunta al CMS e il suo valore precedente viene valutato rispetto a una soglia
- Le istruzioni che superano una soglia sono contrassegnate per essere ulteriormente analizzate e possono generare un avviso
- Il Checker è completamente configurabile per rilevare diversi tipi di canali nascosti



(1): Hash Logic (HL)
(2): CMS Memory (CMSM)
(3): Comparison Machine (CM)
(4): Model Description Register Unit (MDRU)

Conclusione

- La sicurezza hardware è un campo ampio che include le minacce che possono derivare **dalla filiera produttiva fino alle vulnerabilità architetturali**
- Nell'era dell'IoT, i problemi di sicurezza hardware hanno il potenziale per **interrompere il funzionamento affidabile di quasi tutto**, dalle infrastrutture ai veicoli a guida autonoma
- La progettazione per la sicurezza deve essere presa in considerazione nelle **prime fasi di progettazione**
- Il rilevamento di trojan e possibili vulnerabilità dovrebbe essere considerato un **ulteriore collaudo in entrata** per i sistemi utilizzati per applicazioni critiche per la sicurezza.

Alcuni riferimenti

On all the Hardware security topics

- Bhunia, Swarup, and Mark Tehranipoor. ***Hardware security: a hands-on learning approach***. Morgan Kaufmann, 2018.

On Hardware Trojans

- Sally Adee The Hunt for Kill Switch IEEE Design and Test

On Timing Attacks

- Lipp, Moritz, et al. "Meltdown: Reading kernel memory from user space." 27th {USENIX} Security Symposium ({USENIX} Security 18). 2018.
- Kocher, Paul, et al. "Spectre attacks: Exploiting speculative execution." 2019 IEEE Symposium on Security and Privacy (SP). IEEE, 2019.
- Canella, Claudio, et al. "A systematic evaluation of transient execution attacks and defenses." 28th {USENIX} Security Symposium ({USENIX} Security 19). 2019.

Grazie per l'attenzione