



Università
di Catania

Progetto Computer Security A.A 24/25

Modello Concettuale di Terminologia su Attività Offensiva in Cybersecurity

Realizzato dallo studente:

Vincenzo Barba

1000002746

Docente di riferimento:

prof. Giampaolo Bella



Sommario

1. Introduzione	3
2. Ambiguità e differenze nello stato dell'arte	4
3. Glossario dei termini proposto	11
4. Modello concettuale proposto.....	16
5. Applicazione del modello ad attacchi reali.....	17
5.1. TP-Link Tapo C200 (da PETIoT)	17
5.2 Hak5 USB Rubber Ducky 2022	20
6. Fonti e riferimenti bibliografici.....	22



1. Introduzione

La cybersecurity è un ambito in continua evoluzione, caratterizzato da una forte **eterogeneità terminologica** e concettuale. La molteplicità di definizioni provenienti da enti standard (NIST, ISO, MITRE, CAPEC, ecc.), insieme alla diffusione di fonti accademiche e industriali, genera spesso ambiguità che rendono complessa l'interpretazione univoca di concetti fondamentali come *vulnerabilità*, *exploit*, *attacco* o *Penetration Test*. Per tale motivo, risulta essenziale proporre un modello concettuale che raccolga e metta in relazione i principali termini tecnici, fornendo chiarezza e coerenza nell'uso.

Il presente lavoro si concentra in particolare sugli aspetti **offensivi** della sicurezza informatica, ossia sulle tecniche, le metodologie e i modelli che descrivono il comportamento degli attaccanti e le modalità con cui le debolezze dei sistemi possono essere individuate e sfruttate. In quest'ottica, i vari concetti vengono analizzati in maniera comparata tra diverse fonti, evidenziando i punti di convergenza e le aree di ambiguità.

La scelta di focalizzarsi sul dominio offensivo spiega anche alcune esclusioni o distinzioni concettuali: ad esempio, il ruolo del *Blue Team*, tipicamente orientato alle attività difensive e di monitoraggio, non è stato direttamente associato a processi come il *Vulnerability Assessment*, che in questo modello viene considerato principalmente come fase preliminare all'attività di *Penetration Testing*, in quanto sarebbe troppo riduttivo limitarlo a questa unica attività.

Accanto al lavoro di sistematizzazione teorica, viene inoltre proposta un'applicazione pratica attraverso lo studio di scenari d'attacco reali su dispositivi IoT, con particolare riferimento alla telecamera Tapo C200. Tale analisi consente di mettere in evidenza come concetti astratti trovino concreta manifestazione in contesti reali, dimostrando l'importanza di una modellazione chiara e coerente della terminologia.

L'obiettivo complessivo è dunque duplice: da un lato fornire un glossario strutturato e un modello concettuale capace di ridurre le ambiguità terminologiche nella letteratura e nella pratica della cybersecurity; dall'altro mostrare come tali definizioni possano essere applicate in maniera efficace all'analisi e alla comprensione di attacchi concreti.

2. Ambiguità e differenze nello stato dell'arte

- **Vulnerability:** NIST/ISO/UKC includono anche aspetti procedurali e organizzativi in quanto parlano di asset e controlli, mentre NVD/CAPEC/MITRE sono molto più tecnici, citando *bug*, istanza di *weakness* in HW/FW/SW che causa negativo impatto a CIA (quest'ultima citata anche da NIST). UKC/ISO non parlano di CIA. ISO/IEC 27005 adotta una nozione più ampia, includendo carenze procedurali o organizzative (es. mancanza di sostituzione periodica di supporti, interfacce complicate). Vi è ambiguità sul fatto che, essendo un'istanza di una *weakness*, debba necessariamente esserci *exploit* concreto e funzionante, quando invece esistono alcune CVE sfruttabili (attualmente) solo in teoria. Da verificare se tecnicamente si può dire che tutte le vulnerabilità derivano da bug oppure no.
- **Weakness:** non ci sono evidenti ambiguità. NIST/MITRE/CAPEC sono allineati sulla stessa definizione tecnica. In ISO non c'è una definizione vera e propria, ma se ne parla poche volte nella 27005 in modo generico (anche *weakness* procedurale).
- **Zero-Day Vulnerability:** termine non citato in fonti standard, ma da vendor tecnologici. Le definizioni concordano sull'assenza di patch al momento della scoperta, ma differiscono su un punto sostanziale: Splunk considera *zero-day* anche le vulnerabilità note al vendor ma non ancora corrette, mentre Kaspersky e Orca si limitano a quelle sconosciute al vendor (e, di conseguenza senza patch, poi quando scoperte diventano *unpatched vulnerabilities*). Nei media o blog tecnologici è spesso confusa con *0-day exploit* o si considera come un nuovo attacco.
- **N-Day Vulnerability:** non è citato in fonti standard, ma da vendor tecnologici. Secondo Horizon3 una *n-day vulnerability* può essere sia con una annessa patch disponibile che senza, mentre Fortinet restringe implicitamente il concetto solo a quelle con fix disponibile ma non installato.
- **Vulnerability Assessment:** per NIST è sotto-attività di processi più grandi (*risk/security assessment*), in particolare è uno tra i loro input. Vendor e

whitepaper divulgativi la interscambiano erroneamente con *Vulnerability Scanning*, che è invece la parte iniziale del *Vulnerability Assessment*: lo scanning è infatti effettuato tramite tool automatizzati, in aggiunta a esso va fatta un'analisi manuale per fare integrazione qualitativa, ed inoltre per individuare, classificare e valutare criticità e impatto. In ISO 27005 si parla di *Vulnerability Assessment* per vulnerabilità in senso generico, come rischi procedurali/organizzativi, meno tecnici. Testimonianze nei blog dicono che a livello aziendale (lato cliente), capita che viene richiesto un *Vulnerability Assessment* quando invece si voleva (anche) *Penetration Testing*. Dovrebbe limitarsi, invece, oltre a cercare *n-day vulnerabilities* già note, a identificare servizi e porte esposte, protocolli attivi, problemi di configurazione che, date in pasto al *Penetration Testing*, potrebbero portare alla scoperta di *0-day vulnerabilities*. PETIoT la considera una fase di *Kill Chain*, ed inoltre dentro vi include ipotesi di *Vulnerability Discovery*, che invece potrebbe avvenire durante PT.

- **Security Testing:** termine molto ampio e generico che potrebbe essere scambiato con *Penetration Testing*. In NIST SP 800-1115 si parla di *Vulnerability Assessment*, *Penetration Testing*, *code and config review*, ma non di *Risk Assessment*. NIST/OWASP ne parlano come un insieme di attività strettamente tecniche, ma non entrando mai nei dettagli tale da poter essere interpretate in senso più ampio, il che contribuisce alla confusione terminologica osservata anche nel mercato. Blog ([Ambiguous Software Testing Terminology for 2025 · Mateusz Roth](#), il cui titolo dice che dovrebbe spiegare i termini ambigui cyber) e fonti non standard, anche vendor come getAstra ([O 1 - What is Security Testing and Why is it Important? - ASTRA](#)) fanno confusione e ne parlano in modo più generico, includendo *Risk Assessment*. In realtà quest'ultimo dovrebbe essere distinto dal *Security Testing*, ma potrebbe usarne i risultati.
- **Kill Chain:** non sembra esserci ambiguità nel termine in sé, però ogni *Kill Chain* ha una sua granularità diversa, per questo può essere visto come una pluralità di modelli con lo stesso nome generico, fortemente dipendente dal modello di riferimento. Da non confondere con MITRE ATT&CK che invece è un framework che descrive le azioni dell'avversario (TTP) non in modo sequenziale (modello comportamentale). Da non confondere con *Attack Pattern* (CAPEC) che è un modello statico focalizzato su tecniche di attacco specifiche, come SQL Injection o Clickjacking, organizzate per

categorizzazione (es. architettura, memoria, GUI) (https://capec.mitre.org/about/attack_comparison.html): esso è invece come un “design pattern” della sicurezza, descrive in modo strutturato come un certo tipo di attacco viene realizzato, indipendentemente dal contesto.

In sintesi, la *Kill Chain* è la struttura del processo di attacco, mentre CAPEC descrive i singoli “mattoni tecnici” e ATT&CK ne mostra l’uso operativo.

Per esempio, Cyber Kill Chain: “L’attacco segue 7 fasi: reconnaissance → delivery → exploitation...”,

ATT&CK: “Nella fase exploitation un attaccante può usare Spear Phishing Attachment (T1566.001)”,

CAPEC: “Spear Phishing Attachment è un attack pattern, con prerequisiti, modalità e mitigazioni”.

PETIoT presenta 6 fasi, Unified Kill Chain di Paul Pols è composto da 18 fasi, mentre Cyber Kill Chain di Lockheed Martin ne ha 7. La *Kill Chain* PETIoT non include solo passi offensivi, ma anche *Vulnerability Assessment*. PETIoT parla di MITRE ATT&CK e DEFEND come fossero *Kill Chain*, anche se tecnicamente non lo sono. PETIoT ha fasi di Information Gathering e Traffic Analysis che nella UKC potrebbero essere mappate alla fase di Discovery.

- **Exploit:** c’è qualche disguido sul fatto che l’exploit sia un sostantivo o un verbo. CAPEC lo definisce come un’entità (oggetto, input, codice, stringa), mentre OWASP (e molte fonti non standard, blog ecc..) dice che è un’azione progettata per trarre dei vantaggi, focalizzata su attività dell’attaccante. CAPEC parla di “weakness (or multiple weaknesses)”, riconoscendo che un exploit può richiedere la combinazione di più debolezze (es. cattiva validazione input + configurazione errata). In letteratura tecnica questo concetto a volte manca, portando a un approccio troppo “uno a uno” (*vulnerabilità* → *exploit*), quando in realtà molti attacchi sono multi-step. Alcuni testi, blog, usano “*exploit*” come sinonimo di “*payload*” confondendoli, ma tecnicamente: *exploit* = metodo o codice per sfruttare *vulnerabilità*; *payload* = ciò che *l’exploit* trasporta/esegue dopo lo sfruttamento (es. reverse shell, o genericamente malware, codice malevolo). Molti *exploit* diversi possono sfruttare la stessa *vulnerabilità* (*:1).

Un *exploit* che garantisce accesso (anche senza *payload*) potrebbe costituire già di per sé una violazione della sicurezza, perché permette l'uso non autorizzato di risorse del sistema target. Il *payload*, in questi casi, non è ciò che rende l'attacco dannoso, ma ciò che lo rende "utile" all'attaccante, traducendo l'accesso in un obiettivo concreto (es. furto dati, persistenza, sabotaggio). Esiste un'ambiguità nel dire "*exploit* per *vulnerabilità*" e "*exploit* per *weakness*". Valutare se includere tecniche (tool), oltre che dell'esecuzione di input, codice, in quanto il rischio è che, con una definizione troppo rigida, casi borderline come analisi passiva, sniffing, sfruttamento di misconfigurations tramite tools, restino fuori pur essendo *exploit*.

- **Exploitation:** non ci sono conflitti terminologici evidenti, ma la differente terminologia tra modelli (*Kill Chain*, ATT&CK) va esplicitata per evitare confusioni; MITRE ATT&CK ne parla in modo simile come *Execution*, che è una tattica tecnica focalizzata sull'esecuzione del codice malevolo. PETIoT sottolinea il fatto che nonostante una vulnerabilità teoricamente ci sia, non è detto che in un certo ambiente e contesto sia sfruttabile tramite un *exploit* dando un esito positivo. In PETIoT c'è un overlap riguardo la fase di *Exploitation* (in cui è inclusa creazione *exploit* + eventuale *payload*) con "*Weaponization*" della Cyber Kill Chain e con "*Resource Development*" di Unified KillChain (in cui si fa solo preparazione *exploit* + *payload*), che le distinguono da *exploitation* (intesa invece come pura esecuzione dell'*exploit*). Un'altra ambiguità possibile nasce dalla considerazione di fasi intermedie quali social engineering, lateral movement, privilege escalation, come *exploitation*, dato che esse non sfruttano una vulnerabilità tecnica intesa come tale, ma potenzialmente rompono proprietà di sicurezza.
- **Attack Scenario:** nessuna fonte rilevante, può essere perciò considerato un termine non standardizzato, di conseguenza seguono soli appunti. Da distinguere rispetto *Kill Chain* che è un modello astratto di fasi; lo scenario potrebbe invece essere una particolare istanza narrativa (ad alto livello) di quelle fasi applicata a un caso reale, composta da: attore (o tipo di attore), obiettivo, asset coinvolti, sequenza di passi (mappabile a *Kill Chain*). Da non confondere con *Attack Pattern* che è invece la descrizione tecnica di un singolo tipo di attacco. Esempi di scenari d'attacco illustrativi:

https://www.researchgate.net/figure/Six-different-types-of-attack-scenarios_fig2_286246299.

Un *Attack Scenario* potrebbe essere definito da un insieme di precondizioni iniziali e condizioni in corso, sia lato attaccante che sullo stato del sistema target. Qualsiasi variazione sostanziale di queste condizioni modifica la sequenza possibile, generando uno scenario d'attacco differente o una sua variante.

- **Attack Pattern:** Non ci sono ambiguità, per il fatto che se ne parla solo in CWE e CAPEC (entrambi di MITRE), perciò seguono soli appunti. Differenziare rispetto MITRE ATT&CK, in cui una tecnica è più operativa e legata a una tattica, mentre in CAPEC (che è letteralmente un catalogo strutturato di *Attack Pattern*) un *Attack Pattern* è concettualmente più astratto e generalizzato, e comprende: obiettivo, prerequisiti, metodi di esecuzioni e esempi di *exploit*. Esempio: CAPEC può descrivere “SQL Injection” come pattern, mentre ATT&CK può descrivere “SQL Injection via unprepared statements” come tecnica. *Attack Vector* è invece solo una parte del pattern, il quale descrive come viene veicolato lo sfruttamento. Durante un *Attack Scenario* si possono sfruttare uno o più *Attack Pattern*.
- **Attack Vector:** sinonimo di threat vector o path in ISO e NIST. UKC, PETIoT, CWE e CAPEC lo usano come sinonimo di delivery method/channel. Non implica la violazione di una proprietà di sicurezza. Potrebbe essere legato sia all'accesso iniziale che a movimenti successivi nell'ambiente (lateral movement vectors). Da non confondere con *l'exploitation* che è lo step successivo, mentre il vector è solo il canale. Potrebbe far parte di *Attack Pattern*. Potrebbe essere elemento di collegamento tra attaccante e superficie di attacco. In calcolo CVSS può essere di tipo network, adjacent (stessa rete), local (accesso logico), physical (accesso fisico).
- **Attack Surface:** IBM include anche i metodi/mezzi usati dagli attaccanti per arrivare al target, ma quelli sono gli *Attack Vector*, e come altri vendor, fa un abuso di notazione dicendo che è somma di *vulnerabilità*. In realtà potrebbe essere l'insieme dei punti esposti (entry points) che possono avere *vulnerabilità* sfruttabili tramite un vettore ed eventualmente exploit al fine di violare una o più proprietà di sicurezza del sistema target. NIST e

OWASP non includono le *vulnerabilità*. Si dovrebbero considerare anche punti fisici (usb, accessi interni) o social engineering.

- **Penetration Testing:** ambiguità in blog o vendor con *Red Teaming*: *Red Team* = scenario-based, fissa obiettivo strategico, effettua test di detection/response; *Penetration Testing* = test tecnico per sfruttare/scoprire vulnerabilità. In sintesi, si distingue dal *Red Teaming* perché non ha come obiettivo primario la valutazione delle capacità di rilevamento e risposta in senso ampio, ma la verifica tecnica delle vulnerabilità. Può prendere come input la lista di *vulnerabilità n-day* individuate nel *Vulnerability Assessment*, la loro criticità, le porte e servizi esposti, i protocolli utilizzati, al fine di individuare eventuali funzionalità inedite. Può anche combinare più vulnerabilità tra loro. Può portare a *Vulnerability Discovery*. In genere in *Red Team*, il *Penetration Testing* è integrato. La discovery può essere attività autonoma o conseguenza del PT a seconda del contesto, ma concettualmente non coincide con l'obiettivo primario del PT, che è dimostrare l'impatto. UKC e NIST spiegano che non necessariamente deve eseguire tutti gli step della *Kill Chain*.
- **Vulnerability Discovery:** non esiste una definizione standard univoca del termine nei framework principali (NIST, ISO, OWASP), e la sua collocazione varia: può essere considerato un'attività autonoma riguardante la scoperta di *vulnerabilità 0-day* (vedi CERT), parte di un *Vulnerability Assessment* (se intesa come identificazione di *n-day vulnerability* in un dato sistema, vedi IBM).
- **Attacco:** vi è ambiguità sul fatto che anche un tentativo fallito viene considerato attacco (ISO, NIST, CWE), mentre per CAPEC è solo un tentativo andato a buon fine. Riguardo l'inclusione di *exploit*: CAPEC lo lega direttamente all'uso di *exploit*, ISO, NIST, CWE non lo menzionano; infatti, potrebbe per esempio essere un attacco di social engineering, perciò in attacchi di tipo umano/psicologico, *l'exploit* tecnico manca: c'è piuttosto un “abuso di fiducia” o “manipolazione comportamentale”. Secondo CAPEC fa parte di *Kill Chain*, inteso come momento effettivo di *exploitation*, quando in realtà dovrebbe essere un processo corrispondente a quello del PT, ma semplicemente non è contrattualizzato e quindi illecito. Tutti gli *exploit* sono parte di attacchi, ma non tutti gli attacchi includono exploit. La letteratura moderna e gli standard internazionali riconoscono

anche la rottura di altre proprietà critiche oltre CIA, come autenticazione, autorizzazione, non ripudio, corretto uso delle risorse, mentre qualche definizione classica e vecchia è ferma alla sole proprietà di sicurezza CIA. Per evitare ambiguità varie si potrebbe distinguere tra *Attack Process* = sequenza di fasi (alcune preparatorie, altre di compromissione effettiva) e *Attack Event* = singola azione che rompe effettivamente una proprietà di sicurezza.



3. Glossario dei termini proposto

- **Vulnerability:** istanza concreta di una o più *weakness* in un componente HW/FW/SW o nella loro combinazione, la cui potenziale *exploitation* implicherebbe certamente la compromissione di una o più proprietà di sicurezza (CIA, uso di risorse, autenticazione, autorizzazione) relativi al target. Molte vulnerabilità derivano da bug, intesi come comportamenti inattesi rispetto alla funzionalità prevista, ma non tutte: alcune emergono da configurazioni deboli o da scelte progettuali non sicure. Nota: il termine si riferisce esclusivamente a *vulnerabilità* di natura tecnica e non include debolezze di tipo procedurale ed organizzativo (*risks*). Fonti prevalenti: NIST, MITRE, CAPEC, NVD
- **Weakness:** condizione o difetto intrinseco, di natura tecnica, ma generica, cioè non collegato ad uno specifico SW/FW/HW, che potrebbe dar luogo a *vulnerabilità* in caso di riscontro concreto in un sistema target. Fonti prevalenti: NIST, MITRE
- **Vulnerability Database:** collezione che raccoglie *vulnerabilità* specifiche, ciascuna identificata con un codice univoco (es. CVE ID) e descritta tramite metadati standard (ad esempio punteggio CVSS, prodotto e versione coinvolti, data di pubblicazione e riferimenti tecnici). Fonte prevalente: NIST
- **Weakness Database:** collezione strutturata che classifica e descrive categorie di *weakness* (di progettazione, implementazione o configurazione), indipendenti da prodotti particolari, utilizzate come base per la classificazione delle *vulnerabilità* note. Fonte prevalente: MITRE
- **Zero-Day Vulnerability:** *vulnerabilità* potenzialmente sconosciuta al fornitore del sistema target e scoperta (ed eventualmente sfruttata) da terzi, per la quale il vendor non ha ancora avuto tempo di sviluppare e distribuire contromisure, quindi è intrinsecamente una *unpatched vulnerability*. Non è ancora stata pubblicata in modo ufficiale. In definitiva può essere ufficialmente resa pubblica senza patch, oppure dichiarata insieme al fix

trovato, e in entrambi i casi cessa di essere classificata come *0-day*.
 Fonte prevalente: Splunk

- **N-Day Vulnerability:** *vulnerabilità* già divulgata pubblicamente (public disclosure) in modo lecito, per la quale può esistere o meno una patch correttiva. La “N” indica il numero di giorni trascorsi dalla registrazione ufficiale (es. assegnazione di un CVE).
 Fonte prevalente: Horizon3
- **Vulnerability Assessment:** processo sistematico di identificazione, valutazione e priorizzazione delle *vulnerabilità* note (*N-Day Vulnerabilities*) in un sistema, attraverso metodi di analisi automatizzati e manuali, con l’obiettivo di misurare il livello di esposizione e fornire indicazioni di mitigazione. Include anche analisi di misconfiguration e rilevamento di servizi, porte aperte, e protocolli utilizzati (quindi potenzialmente esposte a *vulnerabilità* nuove). Non include l’effettivo sfruttamento delle *vulnerabilità*, che è compito del *Penetration Testing*, né la valutazione complessiva del rischio degli asset, propria del *Risk Assessment*.
 Fonti principali: NIST, Rapid7
- **Security Testing:** insieme di attività tecniche finalizzate a valutare la sicurezza di un sistema, mediante metodi di verifica come *Vulnerability Assessment*, *Penetration Testing*, code review e configuration analysis. L’obiettivo è identificare debolezze sfruttabili e validare l’efficacia delle misure di protezione implementate. Non comprende il *Risk Assessment*, che è un processo distinto e più ampio, nonostante quest’ultimo possa utilizzare i risultati del *Security Testing* come input.
 Fonte principale: IST SP 800-115 (NIST)
- **Kill Chain:** modello concettuale che rappresenta un attacco informatico come una sequenza di fasi tipiche che un avversario può seguire (es. ricognizione, weaponization, delivery, exploitation, ecc.). La *Kill Chain* fornisce una visione sequenziale e tattica del ciclo di vita dell’attacco, ma non richiede che tutte le fasi vengano necessariamente percorse: a seconda dello scenario e del contesto, alcune fasi possono essere saltate, ripetute o modificate.

- **Exploit:** qualsiasi tecnica, metodo, input o strumento utilizzato per sfruttare una *vulnerability*, col fine di violare una proprietà di sicurezza del sistema. È distinto dal *payload*, che rappresenta il contenuto o codice eseguito eventualmente dopo l'*exploit*, anche se spesso vengono consegnati insieme.

Fonte principale: CAPEC

- **Exploitation:** fase offensiva in cui una *vulnerabilità* di un sistema viene concretamente sfruttata tramite un *exploit*, producendo un effetto indebito sul sistema target. Essa comprende sia l'uso di *exploit* già disponibili, sia l'adattamento/creazione di *exploit* ad hoc, nonché l'eventuale associazione di un *payload*.

Fonti principali: PETIoT, UKC

- **Vulnerability Discovery:** evento di individuazione di vulnerabilità nuove in sistemi o applicazioni, non ancora note pubblicamente (*0-day vulnerabilities*), ottenuto per mezzo di attività offensive. Nota: da distinguere rispetto la semplice identificazione della presenza di vulnerabilità già note (*n-day vulnerabilities*) all'interno di un sistema, che avviene invece durante un *Vulnerability Assessment*.

Fonte prevalente: CERT

- **Attack Scenario:** rappresentazione narrativa e contestuale di un attacco informatico, con focus sul cuore dell'attacco, descritto come una sequenza di passi che un attore malevolo può compiere contro specifici asset. È caratterizzato da: precondizioni iniziali, che definiscono il contesto e i vincoli di ingresso; condizioni in corso, che influenzano lo svolgimento delle singole fasi; attore/i coinvolti, obiettivi, asset target. Una variazione sostanziale delle condizioni iniziali o in corso produce uno scenario differente o una sua variante.

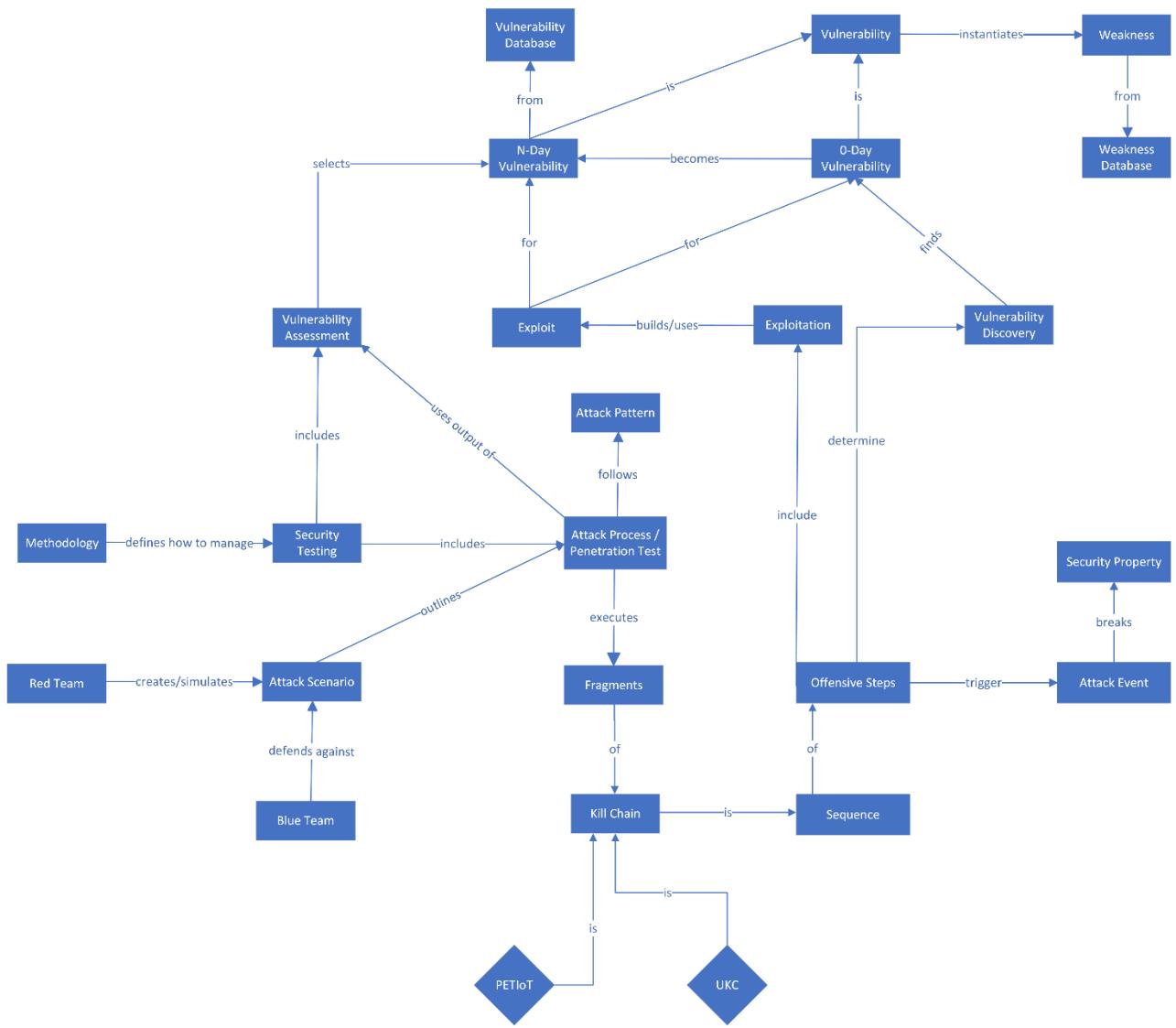
- **Attack Pattern:** descrizione astratta e ricorrente del modo in cui una o più *weakness* possono essere sfruttate da un attaccante per compromettere un sistema. Cattura l'approccio comune, i prerequisiti, i metodi di esecuzione e possibili esempi di *exploit*, senza scendere nel dettaglio di un'implementazione specifica.

Fonte principale: CAPEC

- **Attack Vector:** canale, metodo o percorso attraverso cui un attaccante accede o tenta di accedere a un sistema target per veicolare un attacco. Può riguardare sia l'accesso iniziale (es. phishing e-mail, USB infetta, sito compromesso) sia fasi successive come il lateral movement. Rappresenta il mezzo di ingresso o trasporto, distinto dall'*exploitation* (che è lo sfruttamento effettivo della vulnerabilità).
- **Attack Surface:** l'insieme dei punti di ingresso esposti (*entry points*) — digitali, fisici o umani — attraverso cui un attaccante può interagire con un sistema o un'organizzazione. Comprende interfacce, servizi di rete, API, dispositivi fisici, accessi interni e canali di ingegneria sociale. Non coincide con le *vulnerabilità* in sé, ma con le aree potenzialmente sfruttabili per mezzo di *Attack Vectors* ed eventualmente con *exploit*.
Fonti prevalenti: NIST, OWASP
- **Penetration Testing:** analisi di sicurezza offensiva e autorizzata in cui vengono simulate le tattiche, le tecniche e le procedure di un attaccante reale per valutare la resilienza di un sistema informatico, usando parti di opportune killchain. L'obiettivo è di dimostrare l'impatto concreto di un attacco riuscito, anche sfruttando attivamente le *Vulnerabilità N-Day* sorte durante il *Vulnerability Assessment*. Può anche portare alla scoperta di *0-Day Vulnerability* nel sistema target, concentrandosi sull'analisi manuale di potenziali funzionalità inedite del sistema target.
Fonte prevalente: NIST
- **Attack Process:** attività offensiva non autorizzata, andata a buon fine o meno, che ha il fine di compromettere le proprietà critiche di sicurezza (riservatezza, integrità, disponibilità, autenticazione, autorizzazione, uso delle risorse ...) di un sistema. Si manifesta come un processo articolato, cioè una sequenza di azioni che includono fasi preparatorie e fasi di sfruttamento effettivo. Gli attacchi possono essere di natura tecnologica (tramite *exploit*, strumenti informatici) o comportamentale/organizzativa (tramite manipolazione psicologica, inganno, abuso di fiducia).
Fonti prevalenti: NIST, ISO
- **Attack Event:** istante o evento in cui oggettivamente avviene la rottura di una o più proprietà di sicurezza di un determinato sistema.

- **Red Team:** gruppo autorizzato di professionisti della sicurezza che emula in modo realistico le capacità e le tecniche di un potenziale avversario, conducendo threat emulations o *Attack Scenarios* contro l'organizzazione target. L'obiettivo non è soltanto sfruttare vulnerabilità tecniche, ma dimostrare l'impatto di attacchi completi sugli asset critici e valutare l'efficacia delle difese (*Blue Team*) in un contesto operativo reale. A differenza del *Penetration Testing*, che è spesso focalizzato sulla verifica tecnica e sull'impatto delle singole vulnerabilità, il *Red Team* adotta un approccio più ampio, scenario-based, coinvolgendo la difesa dell'intera organizzazione.
- **Blue Team:** gruppo di professionisti della sicurezza responsabile della difesa di un'organizzazione, che contrasta attacchi reali o simulati (ad esempio quelli condotti da un *Red Team*). Le attività di un *Blue Team* includono valutazioni operative della sicurezza di rete, identificazione di minacce e rischi, monitoraggio continuo, risposta a incidenti, e la proposta di tecniche di mitigazione e miglioramento. Opera in un contesto difensivo e reattivo, con l'obiettivo di rafforzare le difese esistenti e garantire la resilienza del sistema. È riduttivo dire che fa solo *Vulnerability Assessment*, motivo per il quale nel modello concettuale non vi è una relazione, essendo il modello stesso incentrato sulle attività offensive.

4. Modello concettuale proposto



5. Applicazione del modello ad attacchi reali

5.1. TP-Link Tapo C200 (da PETIoT)

Nel paper che introduce la *Kill Chain* PETIoT, vengono illustrati anche una serie di attacchi dimostrativi verso la telecamera IP TP-Link Tapo C200 in ambiente domestico/IoT. L'attacco è stato condotto con un approccio black-box. I sistemi target coinvolti sono: Tapo C200, la telecamera IP connessa alla rete domestica e comunicante (potenzialmente) con server esterni, e con l'app mobile dell'utente; App proprietaria Tapo, il SW ufficiale per smartphone che consente l'accesso al video stream in seguito a rispettiva autenticazione.

Lo **scenario d'attacco** generico prevede che l'attaccante si trovi nella stessa rete locale della telecamera e dell'utente (*Attack Vector*: adjacent). In questo contesto, l'aggressore può assumere diversi comportamenti (e quindi generare diversi scenari):

1. causare un Denial of Service inviando pacchetti in eccesso fino a saturare le risorse della telecamera;
2. intercettare le notifiche di rilevamento di movimenti, sfruttando il fatto che, pur cifrate, hanno dimensione costante e quindi riconoscibile;
3. accedere al flusso video in chiaro qualora l'utente utilizzi software di terze parti che trasmette tramite RTSP senza cifratura.

Il secondo scenario di attacco prevede, come precondizione fondamentale, che l'accesso allo stream video avvenga tramite il SW proprietario dell'app Tapo (in cui il video stream è cifrato con AES key accordata precedentemente). Il terzo scenario necessita invece che l'app Tapo sia settata in modo da eseguire lo stream video tramite software di terze parti (in questo caso la comunicazione del video stream è in chiaro). Un ulteriore scenario, non approfondito dagli autori, si verificherebbe se l'utente non fosse nella stessa rete della telecamera, poiché la comunicazione passerebbe da un server intermedio: esso, presumibilmente, precondizionerebbe i tre scenari d'attacco visti in precedenza.

Gli attacchi, essendo dimostrativi della *Kill Chain* PETIoT presentata, seguono ovviamente le sei fasi incluse in essa. Tra i tipici step offensivi delle altre *Kill Chain* non si evincono, tra tutte, la fase di *reconnaissance* poiché, questa mira a identificare e selezionare gli obiettivi dell'attacco, mentre in questo caso il

dispositivo target è già dato; anche la *weaponisation* risulta ridondante dato che gli attacchi usano kit di tool già pronti, e non c'è nessun *exploit* da scrivere; *installation* e *C2* non sono necessarie in quanto non serve alcun SW malevolo o comunicazione con server dell'attaccante; infine *Actions on objectives* avviene durante *exploitation*. Discorsi simili valgono per gran parte delle fasi della *Kill Chain* UKC.

Tramite lo scanner Nessus è stato condotto uno scanning automatico, il quale però non ha rivelato alcuna vulnerabilità nota (*N-Day Vulnerability*). Tramite le successive fasi di raccolta informazioni, analisi manuali condotte per mezzo di esperienza pregressa nel campo (*Vulnerability Assessment*), sono state scoperte funzionalità inedite; sono poi state testate le potenziali vulnerabilità solitamente associate alla tipologia di funzionalità inedita individuata (*Penetration Testing*) sino all'individuazione di tre *0-Day Vulnerability* (*Vulnerability Discovery*):

- **Vulnerabilità 1** – neutralizzazione impropria di pacchetti in entrata permette DoS: la telecamera non utilizza meccanismi per filtrare il traffico, quindi evaderebbe qualsiasi richiesta di messaggio il prima possibile, senza verificare l'autenticità del chiamante e del contenuto. Ciò significa che un aggressore potrebbe tentare di inondarla di traffico e renderla non disponibile. CWE-707: Improper Neutralisation (*weakness*). *Attack Pattern*: CAPEC-125: Flooding.
- **Vulnerabilità 2** – Entropia insufficiente in notifiche cifrate permette violazione di rilevamento movimenti: la taglia della notifica, nonostante sia cifrata, è sempre di 523 bytes. Questo permette, nonostante non si intervenga nello schema crittografico (di per sé sicuro), di discernere tra notifiche di rilevamento dei movimenti rispetto alle restanti. CWE-331: Insufficient Entropy (*weakness*). *Attack Pattern*: CAPEC-94: Adversary in the Middle (AiTM), CAPEC-158: Sniffing Network Traffic. Nota: L'attaccante, oltre ad intercettare, potrebbe anche fare drop delle notifiche e convincere la vittima che non c'è alcun movimento in casa.
- **Vulnerabilità 3** – trasmissione in chiaro di stream video permette violazione da agenti indebiti. Il SW di terze parti riceve lo stream tramite protocollo RTSP sulla porta 554 e non supporta cifratura. Ogni qual volta è fatta una richiesta, il video è inviato in chiaro con encoding H264. Una decodifica (decoding H264) da parte dell'attaccante permetterebbe di

ottenere il video. CWE – 319: Cleartext transmission of sensitive information (*weakness*). *Attack Pattern*: CAPEC-94: Adversary in the Middle (AiTM), CAPEC-158: Sniffing Network Traffic.

- **Exploit vulnerabilità 1** - intenso scan con Nessus porta crash telecamera, e potrebbe essere dato da: memoria insufficiente specifica del TCP/IP stack, anche se non si è certi a causa della mancanza di informazioni sull'implementazione del protocollo TCP/IP. Causa completa perdita di disponibilità di ogni servizio offerto dalla telecamera. *L'exploit* qui è l'uso di un tool di flooding (es. Nessus stesso o script DoS) che invia input malevoli (pacchetti in eccesso). La mitigazione può avvenire a livello kernel o firewall con strategie anti DoS.
- **Exploit vulnerabilità 2** - intercettazione pacchetti con Ettercap (rottura della confidenzialità dei movimenti rilevati) e analisi pacchetti di lunghezza 523 ogni 10 minuti. Tramite iptables, inoltre, l'attaccante potrebbe fare drop di queste notifiche (rottura proprietà di disponibilità). Qui l'*exploit* è l'analisi passiva + pacchetti crafted di drop con iptables. È più borderline: è *exploitation* tramite manipolazione traffico, ma non c'è un “*exploit code*” classico. Si può però comunque qualificare come exploit (è un metodo tecnico, usa un tool). La mitigazione potrebbe essere la randomizzazione di taglia dei frame con detection di movimento.
- **Exploit vulnerabilità 3** - MITM tra Tapo C200 e device con app. Usato Wireshark e estrattore h264 per intercettare e poi ricostruire il video dai pacchetti. Implica rottura di confidenzialità, integrità (perché l'attaccante potrebbe rimpiazzare frame con altri o modificarli) e disponibilità (possibile drop). L'*exploit*, come nei casi precedenti, può essere considerato l'utilizzo dei tool prima descritti. La mitigazione si potrebbe ottenere aggiungendo un tunnel cifrato durante la comunicazione: in particolare viene proposto un Raspberry Pi 4 Model B come access point della Tapo C200 per modificare il traffico in transito applicando un livello di cifratura.

5.2 Hak5 USB Rubber Ducky 2022

Lo **scenario d'attacco** generale prevede come target un laptop che accetta input da periferiche USB. L'attaccante dispone di un dispositivo Hak5 USB Rubber Ducky configurato per emulare una tastiera (HID – Human Interface Device). L'attaccante ottiene accesso fisico diretto alla macchina (ad esempio in un ufficio, laboratorio, reception) oppure sfrutta una tecnica di USB drop / social engineering, lasciando il dispositivo USB in luoghi dove un utente potrebbe prenderlo e poi inserirlo per curiosità. Una volta collegato, il dispositivo viene automaticamente riconosciuto come tastiera legittima dal sistema operativo, senza ulteriori verifiche di autenticità. In pochi secondi, il dispositivo inietta sequenze di tasti programmati (in Duckyscript) che aprono terminali e lanciano comandi malevoli, consentendo escalation di privilegi o l'installazione di backdoor, restando invisibile all'utente.

L'*Attack Vector* è l'interfaccia USB HID, che viene abusata per trasportare comandi non autorizzati, e in aggiunta, se utilizzate, tecniche di social engineering per permettere l'inserimento dell'USB. Gli *Attack Pattern* possibilmente seguiti sono: CAPEC-150 – Collect Data from Removable Media, CAPEC-153 – Input Injection. Le fasi della Unified *Kill Chain* seguite sono: Reconnaissance e Social Engineering (se non si ha accesso fisico al sistema target); Delivery (inserimento fisico del dispositivo USB malevolo); Exploitation (iniezione dei comandi tramite keystroke injection); Actions on Objectives (esfiltrazione dati, apertura di backdoor, escalation), e potenzialmente molte altre una volta che si è dentro il target.

Le *weakness* sfruttate sono: CWE-1204 – Improper Access Control of Peripheral Device Interface; CWE-203 – Observable Discrepancy / Inadequate Authentication; CWE-640 – Weak or Missing Device Authentication; CWE-280 – Improper Handling of Insufficient Permissions or Privileges. La possibile definizione di *vulnerabilità* è: il sistema operativo accetta input da qualunque dispositivo HID collegato, senza procedure di verifica/autenticazione dell'origine.

L'*exploit* è keystroke injection (praticamente integrato già in firmware hak5 usb rubber ducky), cioè il fatto che il dispositivo emuli una tastiera e invii automaticamente sequenze di tasti senza essere bloccato dal sistema, che crede invece di star ricevendo input umano. Il *payload* è composto dai comandi effettivamente scritti in Duckyscript ed

eseguiti poi sul sistema, come download di tool malevoli, creazione di backdoor, furto credenziali.

Le proprietà di sicurezza violate potrebbero essere: integrità (possibile alterazione dello stato del sistema con comandi arbitrari); segretezza (possibile furto di dati o credenziali) e autorizzazione/autenticazione (bypass dei controlli, in quanto il dispositivo viene accettato come tastiera trusted senza autenticazione, o come conseguenza della perdita di segretezza e attacchi in cascata). Le mitigazioni possibili potrebbero essere whitelist di dispositivi USB, e strumenti specializzati che possono rilevare la velocità di digitazione innaturale (Ducky digita centinaia di caratteri al secondo, molto più di un umano), o anche EDR.

6. Fonti e riferimenti bibliografici

NIST: <https://csrc.nist.gov/glossary/>

MITRE: <https://www.cve.org/ResourcesSupport/Glossary/>,
<https://cwe.mitre.org/documents/glossary/index.html>

NVD: <https://nvd.nist.gov/vuln>

CAPEC: <https://capec.mitre.org/about/glossary.html>

UKC: <https://www.unifiedkillchain.com/assets/The-Unified-Kill-Chain.pdf>

ISO 27000:2018: <https://www.iso.org/standard/73906.html>

Splunk: https://www.splunk.com/en_us/blog/learn/zero-day.html

Kaspersky: <https://www.kaspersky.it/resource-center/definitions/zero-day-exploit>

Orca-Security: <https://orca.security/glossary/zero-day-vulnerability/>

Fortinet: <https://www.fortinet.com/blog/psirt-blogs/importance-of-patching-an-analysis-of-the-exploitation-of-n-day-vulnerabilities>,
<https://www.fortinet.com/resources/cyberglossary/attack-vector>

Rapid7: <https://www.rapid7.com/blog/post/2021/06/10/attack-surface-analysis-part-1-vulnerability-scanning>

ISO 27002:2022: <https://www.iso.org/standard/75652.html>

ISO 27005:2022: <https://www.iso.org/standard/80585.html>

NCSC: <https://www.ncsc.gov.uk/guidance/penetration-testing>

OWASP: <https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/stable-en/03-appendices/05-glossary>,
https://cheatsheetseries.owasp.org/cheatsheets/Attack_Surface_Analysis_Cheat_Sheet.html

PETIoT e TP-Link attack: <https://www.iris.unict.it/handle/20.500.11769/551924>

IBM: <https://www.ibm.com/think/topics/vulnerability-management>,
<https://www.ibm.com/think/topics/attack-surface>

Attacco USB Rubber Ducky 2022: <https://docs.hak5.org/hak5-usb-rubber-ducky/ducky-script-basics/keystroke-injection/index.html>