

Open Source e cybersecurity per un software sicuro

sicurezza0gni tipo di software include degli elementi *Open Source*. E oggi, particolarmente nel contesto dell'Industria 4.0, non c'è attività lavorativa che non implichi l'utilizzo di software: qualunque sia la tipologia di business o il settore in cui si opera, il software (con le sue inevitabili componenti *Open Source*) è un fattore abilitante per la maggior parte delle operazioni, rappresentando una componente essenziale per il lavoro di tutti noi.

L'onnipresenza dell'*Open Source*

L'obiettivo del report "*Open Source Security and Risk Analysis*" (**OSSRA**) – curato dal Synopsys Cybersecurity Research Center (CyRC) in base alle ricerche svolte nel 2021 su oltre 2400 *code bases* commerciali usate in 17 organizzazioni – è fornire alle aziende un quadro completo della propria filiera operativa. Ciò richiede, naturalmente, consapevolezza circa le origini e il funzionamento degli strumenti di lavoro utilizzati al suo interno: tecnologie, application security, sistemi operativi e piattaforme digitali a cui sono quotidianamente affidate funzioni centrali e spesso critiche, quali la condivisione di dati relativi ai clienti o utenti finali, la gestione di macchine connesse (come nel caso delle infrastrutture OT) o il funzionamento degli stessi SOC fondamentali a garantire la sicurezza.

La ricerca mostra **probabilità vicine al 100% circa l'utilizzo di componenti Open Source all'interno di qualsiasi software. Questi elementi rappresenterebbero addirittura la maggioranza del codice, con una media rilevata del 78%.**



L'analisi offre anche indicazioni pratiche su come individuare e gestire questa tipologia di codice, per ridurre al minimo gli inevitabili rischi connessi al suo impiego, quali vulnerabilità non patchate o conflitti di licenze, circostanze che possono generare impatti di vario livello; inclusi problemi legati alla responsabilità sul piano legale, talvolta tali da compromettere la stessa sopravvivenza dell'azienda. I processi di software development devono avere come obiettivo la protezione e la sicurezza.

Perché utilizzare l'*Open Source*?

L'utilizzo di software Open Source, oltre ad essere inevitabile, comporta innumerevoli vantaggi. Lo conferma la sua estrema popolarità: tra i principi fondanti dell'OS vi sono infatti la gratuità e la possibilità di apportare infinite modifiche per rispondere alle esigenze degli utilizzatori, fintanto che siano rispettati i termini previsti dalle varie licenze.

Un altro punto a favore dell'*Open Source* è la capacità di fornire i "mattoni" con cui costruire architetture digitali complesse, mettendo a disposizione degli sviluppatori materiali gratuiti da "ricombinare" in nuove forme a tutto vantaggio della creatività (nonché del budget). Ecco perché la maggior parte dei prodotti software odierni non viene creata da zero ma, al contrario, partendo da componenti già esistenti che vengono di volta in volta ri-assemblati per fare fronte a nuovi e diversi obiettivi.

Includere l'*Open Source* nei processi è ormai imperativo, considerato che – rileva ancora il report – rappresenta *"la base fondante di ogni applicazione a cui ci affidiamo"*.

Il software *Open Source* è sicuro?

Se da un lato il software OS non appare intrinsecamente meno affidabile dei software proprietari o commerciali, dall'altro

non risulta neanche *più* sicuro; vista la rilevata pervasività di elementi *Open Source* all'interno di ogni tipo di software, si tratta certo di un'informazione rilevante. Al riguardo il report OSSRA 2022 (il settimo dedicato al tema) evidenzia che *"l'Open Source è ovunque: ciò determina la necessità di una sua adeguata implementazione. Identificare, monitorare e gestire questi elementi risulta vitale in un'ottica di software security"*. La centralità del software nei processi, con l'annessa pervasività del software libero al loro interno, fa infatti sì che i suoi pregi e difetti finiscano per influenzare e condizionare tutti gli ambiti in cui viene utilizzato.

Come ogni prodotto umano, nessun software è perfetto; e l'*Open Source* non fa eccezione.

Fra i trend esposti dalla ricerca, è comunque incoraggiante notare come la percentuale di codice che presenta almeno una vulnerabilità di origine *Open Source* appaia in lieve calo (dall'84% del 2020 all'81%). Ancora più significativo **il declino (dal 60% al 49%) della presenza di vulnerabilità ad alto rischio**. Per quanto positive, queste tendenze non si traducono nell'assenza di problemi: il dato dell'81% risulta ancora a favore dei potenziali attori malevoli, e non di chi intenda contrastarli. Allo stesso modo, il fatto che la metà delle librerie di codice sorgente presenti vulnerabilità note implica una superficie d'attacco molto ampia a disposizione dei cybercriminali.



Vulnerabilità dell'*Open Source*

Uno degli indicatori di quanto sia urgente, per le organizzazioni, sforzarsi di mettere in sicurezza il software libero e ridurre la superficie esposta al rischio cyber proviene dalla rilevanza dei settori coinvolti: tra quelli esaminati ci sono infatti IoT, aviazione e *automotive*,

trasporti e logistica, che mostrano vulnerabilità in oltre il 60% del codice impiegato. In comparti altrettanto essenziali (internet e *mobile apps*, settore educativo ed energetico, marketing, FinTech, eCommerce e *retail*, industriale e manifatturiero, robotica e *software as a service/SaaS*) la percentuale risulta uguale o superiore al 50%.

Inoltre, alcune delle vulnerabilità note appaiono persistenti e persino in crescita: secondo gli esperti, tra le *code bases* esaminate *“l’88% conteneva versioni non aggiornate delle componenti Open Source [...] anche laddove aggiornamenti o patch siano disponibili, queste non vengono applicate con regolarità”*.

Va ricordato che si parla di vulnerabilità pubblicamente note: questo significa che anche i nostri nemici le conoscono e fanno di poterle sfruttare a proprio vantaggio. Come sintetizzato dal report, *“si può concludere che alcuni team di DevSecOps facciano fatica a tenere il passo dei rischi veicolati dall’Open Source”*.

Rischi e vantaggi dell’Open Source nel lungo periodo

La buona notizia emersa dalla relazione redatta da Synopsys – azienda americana che si occupa della qualità del software, in forte crescita in Italia grazie al regional sales manager Emanuele Buriali d’Arezzo – è che proprio una delle peggiori vulnerabilità di origine *Open Source* rilevate potrebbe, in una prospettiva di lungo periodo, aiutare le organizzazioni a prestare maggiore attenzione alla sicurezza dei software.

Log4Shell, un gruppo di vulnerabilità riscontrato nella libreria *Open Source Apache Log4j*, era già presente in centinaia di milioni (se non miliardi) di sistemi, siti e servizi prima di diventare pubblico nel dicembre 2021. Si tratta di vulnerabilità di tipo *“remote code execution”*: quindi idonee a permettere agli attaccanti di ottenere il

controllo remoto dei dispositivi che utilizzano applicazioni in Java, uno dei linguaggi di programmazione più diffusi al mondo.

Proprio in ragione del suo impatto, l'indagine ha rilevato come Log4Shell abbia spinto numerose organizzazioni a prendere atto di molte questioni relative all'uso dell'*Open Source* nel loro business:

- i progetti *Open Source* sono creati e animati da volontari, non da operatori commerciali; di conseguenza, patch e aggiornamenti di sicurezza non vengono “offerti” agli utenti ma devono essere “scaricati” da questi ultimi.
- Alcuni progetti OS finiscono per essere abbandonati e non subire più alcuna manutenzione: pertanto, eventuali difetti o vulnerabilità che emergano al loro interno non vengono presi in carico da nessuno.
- Se i progetti *Open Source* più popolari possono contare su un'ampia comunità impegnata a “prendersi cura” del codice, esistono milioni di progetti meno diffusi in cui solo una decina di persone (o meno) si occupa della loro manutenzione; ciò può significare che un progetto non riceve aggiornamenti da anni.
- Molti sviluppatori, presi dall'entusiasmo per ciò che una componente *Open Source* può offrire ai loro prodotti, tendono ad applicarla senza verificarne l'affidabilità; questo può portarli a ignorare eventuali rischi e a tralasciare quei controlli di sicurezza che sono, invece, richiesti per ogni software commerciale o proprietario.
- Le componenti *Open Source* non sono di per sé più sicure o meno sicure del software “proprietario”; una loro cattiva gestione, in ogni caso, può comportare significativi pericoli per il business.

Come tutelarsi dalle vulnerabilità dell'*Open Source*?

Un buon punto di partenza può essere il **Software *Bill of Materials* (SBOM)**, che offre un inventario dettagliato di tutte le componenti presenti in una determinata collezione di codice sorgente insieme a informazioni sul gruppo di volontari che l'ha creata, restrizioni imposte da ogni licenza attiva, eventuali vulnerabilità note e collegamenti con altre componenti software (dette *dependencies*) necessarie al suo funzionamento. Si tratta, in sostanza, di una "lista degli ingredienti" relativa alla *supply chain* del software. SBOM svolge una funzione impossibile da effettuare "a mano": ossia scavare nelle profondità della *dependency chain*, per garantirne piena e completa visibilità.

Ancora secondo l'indagine OSSRA, ***"le software dependencies coincidono con i maggiori rischi all'interno della supply chain. L'unico modo per ridurre al minimo tali rischi è utilizzare SBOM, che traccia le dependencies e i relativi pericoli, consentendo di pianificare le azioni necessarie ad affrontarli nel corretto ordine di priorità"***.

Indubbiamente, un programma SBOM può rappresentare una risorsa fondamentale in caso vengano scoperte vulnerabilità rilevanti. Nel citato caso di Log4j, un'organizzazione dotata di uno SBOM aggiornato avrebbe potuto condurre una semplice ricerca sul proprio database per scoprire se quella specifica libreria OS fosse presente in qualsiasi punto della sua catena di software; sapendo, in caso positivo, di dover velocemente individuare e applicare le necessarie patch. Con una soluzione di questo tipo si sarebbe potuto evitare gran parte del caos derivato da Log4Shell così come le notti insonni trascorse da molti sviluppatori e *security team* nel periodo natalizio.

Come confermano gli analisti di Synopsys, infatti, ***"la corsa alle attività di remediation è stata spesso frutto di una***

*mancata consapevolezza, da parte delle organizzazioni, sulla presenza e la posizione di componenti Log4j all'interno dei loro sistemi. [Pertanto] è importante ricordare che **il problema non è l'Open Source in sé, quanto la sua insufficiente o inadeguata gestione**".*

Condividi sui Social Network:

<https://www.ictsecuritymagazine.com/articoli/open-source-e-cybersecurity-per-un-software-sicuro/>