Detekcija i načini otkrivanja ranjivih servisa te pretraga sustava kako bi se utvrdilo je li na sustavu iskorištena ranjivost u biblioteci log4j za sistem administratore

1) "Whitebox" (lokalne) provjere:

Na Linux i Windows poslužiteljima te na računalima zaposlenika možemo pretraživati *filesystem* i pronaći softver koji koristi **"log4j2".**

Powershell naredba - Windows:

gci 'C:\' -rec -force -include *.jar -ea 0 | foreach {select-string "JndiLookup.class" $_} | select -exp Path

Linux naredba:

find / 2>/dev/null -type f -name "*.jar" -print0 |xargs -0 grep -i "jndilookup"

Po rezultatima tih naredbi može se pronaći Java softver koji bi mogao biti zahvaćen ovom ranjivošću. Tada je moguće dalje istražiti je li ova ranjivost primjenjiva (pogledati jesu li se autori softvera oglasili o tome), je li dostupna sigurnosna zakrpa ili je li moguće primijeniti druge mitigacije [1]

Pomoću ovih naredbi trebalo bi biti moguće pronaći dobar dio zahvaćenog softvera. Moguće je da neće detektirati neke ranjive servise, primjerice ako su obfuscirali kod, ako se radi o ugniježđenim JAR datotekama (JAR arhiva unutar JAR arhive) ili ako se radi o softveru koji koristi neki drugi format (npr. WAR).

Može se razmisliti i o upotrebi *custom* alata koji su razvijeni za detekciju softvera koji koriste log4j2 biblioteku, naravno prije tog je potrebno pregledati izvorni kod tih alata. [2][3][4]

Ako je Vaša organizacija razvila nekakav Java softver, potrebno je provjeriti je li korištena biblioteka "log4j2" i ažurirati je ako je potrebno. Bitno je rekurzivno provjeriti sve zavisnosti (primjerice pomoću maven dependency:tree plugina).

2) Provjera korištenog softvera

Bilo bi idealno kada bi postojao centralni inventar korištenog softvera u organizaciji, pa bi se onda mogle provjeriti obavijesti svakog dobavljača softvera (Apache, Cisco, Elasticsearch) o ovoj ranjivosti.

Ako takav inventar ne postoji, moguće je provjeriti neku od lista do sad otkrivenog ranjivog softvera [5][6]. Treba imati na umu da takve liste neće uključivati sav ranjivi softver.

Posebno savjetujemo da se povremeno provjerava "**Cisco vulnerability advisory**", budući da je velika količina proizvoda još uvijek "pod istragom" [7].

Važno je napomenuti da ova ranjivost ne utječe samo na sustave koji su javno izloženi na Internetu, već potencijalno utječe i na bilo koji sustav do kojeg zlonamjerni *string* može doći. Posebnu pažnju potrebno je obratiti na sustave na koje se šalju logovi (npr. **ELK instance**, **Splunk**).

Također, zbog relativno laganog načina izvršavanja ranjivosti, ni interni sustavi na koje se ne šalju nikakvi logovi nisu u potpunosti zaštićeni (primjerice napadač pošalje zaposleniku poveznicu na mail, i korisnik klikom na link "napadne" interni sustav).

Budući da je ranjivost javna već nekoliko dana, nakon otkrivanja ranjivog sustava potrebno je analizirati logove kako bi se utvrdilo je li sustav već kompromitiran [8].

3) Dinamičko testiranje

Kao dodatna provjera moguće je i dinamičkim testiranjem, odnosno "*fuzzingom*" provjeravati sigurnost sustava.

Savjetujemo svima da testiraju servise za koje smatraju da bi mogli biti ranjivi putem jednog od alata navedenog u referencama. Postoji nekoliko alata koje je moguće koristiti [9][10]. Profesionalni pentest alati kao što su **Burpsuite Pro** ili **Nessus** također imaju ekstenzije za provjeru ove ranjivosti [11][12]

Za kritičnu infrastrukturu potrebno je ove alate modificirati tako da ne koriste vanjske servise poput "dnslog.cn", kako ne bi curile informacije o ranjivim servisima. Po mogućnosti, preporučujemo da se modificira "payload" kako bi se izbjegli "false positive" rezultati (nije svako posjećivanje napadačevog URL-a nužno zbog ranjivosti) [13]

Ovakvim automatskim provjerama neće biti obuhvaćene svi mogući načini na koje bi netko mogao iskoristiti ranjivost, ali vjerujemo da će pokriti većinu slučajeva koje će koristiti napadači slabijih vještina.

Reference:

[1] https://logging.apache.org/log4j/2.x/security.html

[2] https://github.com/darkarnium/CVE-2021-44228

[3] https://github.com/mergebase/log4j-detector

[4] https://gist.github.com/righettod/0f2a7491a312d1ff5823b73058e55016

[5] https://github.com/authomize/log4j-log4shell-affected

[6] https://www.techsolvency.com/story-so-far/cve-2021-44228-log4j-log4shell/#affected-products

[7] https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-apache-log4j-qRuKNEbd

[8] https://gist.github.com/Neo23x0/e4c8b03ff8cdf1fa63b7d15db6e3860b

[9] https://github.com/fullhunt/log4j-scan

[10] https://github.com/Diverto/nse-log4shell

[11]https://github.com/PortSwigger/active-scan-plus-plus/commit/b485a0744140533d877ce244603502b42f9c6656

[12] https://www.tenable.com/plugins/was/113075

[13] https://twitter.com/bojanz/status/1469386131544621056

CERT.hr - Odjel za Nacionalni CERT

Hrvatska akademska i istrazivacka mreza - CARNET

Josipa Marohnica 5, 10000 Zagreb

tel: +385 1 6661 650, fax: +385 1 6661 767

e-mail za upite: ncert@cert.hr

e-mail za prijavu incidenata: incident@cert.hr

www.cert.hr

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

Zavod za elektroničke sustave i obradbu informacija

Laboratorij za sustave i signale

Unska 3, 10000 Zagreb

Tel: (01) 6129 963 Fax: (01) 6129 889

E-mail: ured-kontakt@lss.hr

www.lss.hr