Sigurnost operacijskih sustava i aplikacija

Laboratorijska vježba

Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva

Bilježenje sistemskih i operativnih zapisa u ELK-u i rekonstrukcija događaja

Andrej Lovei

Zagreb, 9. 5. 2025.

Uvod

Bilježenje zapisa (logiranje) je jako važan dio održavanja sustava jer se u logovima vidi tko je što kada napravio. Za velike sustave je potrebno automatizirati ili dovoljno dobro filtrirati logove. Tu nam u spas dolazi ELK (Elastic Stack), sustav za obradu i vizualizaciju podataka.

Zadatak

U velikoj tvrtki koja se bavi web hostingom vrlo je važna sigurnost i nadzor aktivnosti nad serverima i njenim sustavima. Ovdje se za svakog zaposlenika može tražiti pregled lokalnih aktivnosti ili događaja na njegovom radnom računalu. Tu se može implementirati logger koji prati zapis tipkovnice, pokrete i aktivnosti miša te sadržaj međuspremnika. Uz te podatke se na centralizirani sustav za analizu i vizualizaciju logova može tražiti i slanje sistemskih događaja kao na primjer Windows Event-ova (provjera sumnjivih događaja). Za tu svrhu je ELK (Elasticsearch, Kibana i Logstash) zajedno sa Python aplikacijom savršen.

Teorijski podloga

Za dobivanje podataka u Python aplikaciji se koriste paketi pynput i pyperclip koji nisu dio standardne Python biblioteke. Pynput omogućava dobivanje podataka o kretanju miša i tipkovnice, dok pyperclip omogućava čitanje međuspremnika (clipboard). Ti podaci se zasebno spremaju u posebne datoteke.

Druga aplikacija dohvaća Windows događaje kategorije security iz Windows Event Log-a pomoću pywin32. Pojedini događaj se serijalizira u JSON format i zapisuje u datoteku.

ELK[1] ima puno aplikacija od kojih su za vježbu korištene Elasticsearch, Kibana, Logstash i Winlogbeat.

Elasticsearch je centralizirani pretraživatelj baziran na Apache Lucene pretraživatelju. Dok se izvršava, on aktivno pohranjuje i indeksira podatke koji mu dolaze.

Logstash je sažimatelj zapisa koji kontinuirano uzima podatke iz različitih ulaznih izvora, filtrira ih i strukturira te šalje dalje u Elasticsearch.

Kibana je vizualizacijski alat koji radi nad Elasticsearch-em. Korisnik uz Kibanu može analizirati i vizualizirati podatke kroz vrijeme što olakšava analiziranje sigurnosnih incidenata i kontinuirano promatranje korisničkih akcija.

Winlogbeat je sakupljač zapisa koji šalje Windows Event zapise direktno Logstashu u JSON formatu. To sam koristio dok nisam napisao aplikaciju u Python-u. Winlogbeat se samo instalira i pokrene kao servis i bez problema se vide svi događaji u Kibani.

Dakle, za ovu vježbu, Logstash je između Elasticsearch-a i Python aplikacija i kontinuirano prati log datoteke koje se pune podacima koje aplikacije pošalju. Za aplikaciju koja prati lokalne aktivnosti korisnika filtrira se pomoću grok-a čime se dobiva vrijeme zapisa (timestamp), razinu zapisa te poruku. Kod aplikacije koja prati sistemske događaje, koristi se codec => json s kojim se specificira da treba raspakirati JSON poruke.

Aplikacije se lagano proširuju i izmjenjuju. Na primjer, može se filtrirati određeni eventID ili određeni tip događaja, ali to je lakše napraviti samo u Kibani (spomenuto radi manje memorijske iscrpljenosti jer log datoteke mogu postati velike).

Postavke za vježbu

Sve sam radio u Operativnom sustavu Windows 11. Python verzija je 3.12.0. Pakete koje sam instalirao su pynput, pyperclip i pywin32 (pip install <paket>).

Elasticsearch, Kibanu, Logstash i Winlogbeat sam preuzeo s elastic.co stranice (9.0.0 verzije). Raspakirao sam ih u zajednički direktorij (LAB) i pokretao iz bin direktorija (elasticsearch.bat, kibana.bat, bin\logstash -f configFile.conf (iz logstash direktorija u kojem je i configFile.conf) i Start-Service winlogbeat). Dvije su konfiguracijske datoteke za Logstash (jedna po aplikacija). Svi detalji instalacije su na stranici (nije komplicirano, Kibanu raspakirati 7zip-om jer obično raspakiravanje traje užasno dugo – poznat bug). Pokretanje alata traje par minuta tako da treba biti strpljiv. Elasticsearch će biti na localhost:9200, no to nije bitno već Kibana koja je na localhost:5601. Prvo treba pokrenuti Elasticsearch, potom Kibanu pa aplikaciju i Logstash. Također, jedna od specifičnosti je da se putanje moraju pisati apsolutno u konfiguracijskim datotekama.

U istom direktoriju gdje su i alati kreirati logs direktorij (pojedinačne datoteke će se same stvoriti pri pokretanju). app.py i windowsLogs.py također staviti u isti direktorij LAB. windowsLogs.py je potrebno pokrenuti s administratorskim pravima.

Rješenje vježbe

U aplikaciji za korisničke akcije, svake sekunde se gleda međuspremnik i piše sadržaj u datoteku clipboard.log. Isto tako, za miš i tipkovnicu su definirani Listeneri koji slušaju cijelo vrijeme dok se ne prekine aplikacija i zapisuju u svoje datoteke (mouse.log i keyboard.log). Za miš se detektiraju akcije pomicanja miša, pritiska lijevog ili desnog gumba te pomicanja gore-dolje. Format vremena sam namjestio da bude u ISO8601 formatu da ga lakše filtriram sa grok-om, makar se može i bez toga nego napraviti custom timestamp.

Druga aplikacija prikuplja događaje iz Windows Event Log-a iz kategorije Security (može se namjestiti na i na Application i System. Svaki događaj se individualno serijalizira u JSON te sprema u windowsEvents.log.

Logstash je posrednik koji filtrira log datoteke i šalje ih Elasticsearch-u.

Za korisničke zapise koristi se grok filter kojim se linije datoteka parsiraju u timestamp, loglevel i logMsg za filtriranje u Kibani. Za zapise druge aplikacije koristi se codec => json jer su zapisi u JSON formatu pa se samo raspakiraju.

Logstash kontinuirano čita datoteke i šalje strukturirane zapise u Elasticsearch.

U Kibani potom možemo detaljno sve analizirati: po vremenu (razdoblje kada su zapisi nastali), razini zapisa, što točno piše u zapisu (npr. tražimo kada je sve netko pritisnuo Enter tipku, lako se filtrira: logMsg: „Special key Key.enter pressed“), ID-u događaja… Vizualizacije su u obliku grafova u kojima možemo odabrati hoćemo li prikaz u postotcima ili ukupnom broju događaju… Prvo se ide na Discover te Data Views i onda Create data view. Tada se u Index pattern upisuje

ono što smo stavili pod index kod output-a u Logstash konfiguracijskoj datoteci. Nakon toga možemo filtrirati i sve opisano. Također, ako odemo pod Visualize Library i Create new visualization -> Lens, možemo odabrati tip vizualizacije (Bar, Line, Pie, Table) te podatkovni izvor (isto onaj index iz confFile.conf-a). Tada možemo mišem povući razna polja (fields). Na primjer možemo Bar tipom prikazati na horizontalnoj osi log.file.path.keyword što je putanja te na vertikalnoj osi Records. Tako dobivamo koliko je zapisa u kojoj datoteci.

Zaključak

Naučio sam kako se lako može pratiti korisnička uporaba ulaznih uređaja poput tipkovnice i miša te ekstrakcija sadržaja međuspremnika. Također sam naučio dohvatiti Windows sistemske događaje pomoću skripte u Python-u. Naučio sam osnovni rad s Python Logger-ima.

Uz to, ELK je odličan sustav za nekomplicirano filtriranje podataka te vizualizaciju i analizu zapisa u stvarnom vremenu. Posebno mi se svidjela odlična vizualizacija u Kibani.

Literatura

[1] The Complete Guide to the ELK Stack, Dotan Horovits - https://logz.io/learn/complete-guide-elk-stack/#latest-on-the-elk-stack

[2] Sekoia, Winlogbeat - https://docs.sekoia.io/integration/categories/endpoint/winlogbeat/

[3] Kibana dokumentacija - https://www.elastic.co/docs/reference/kibana

[4] Parsing Logs with Logstash - https://www.elastic.co/docs/reference/logstash/advanced-pipeline

[5] Python logger output dates in IS8601 format - https://stackoverflow.com/questions/50873446/python-logger-output-dates-in-is8601-format

[6] pyperclip 1.9.0 - https://pypi.org/project/pyperclip/

[7] pynput 1.8.1 - https://pypi.org/project/pynput/

[8] Json codec plugin - https://www.elastic.co/docs/reference/logstash/plugins/plugins-codecs-json