VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií

Technická správa

PCAP NetFlow v5 exportér

18. novembra 2024 Maryna Kucher, xkuche01

Obsah

1 Úvod **3**

2 Návod na použiti **3**

3 Návrh **3**

4 Implementace **4**

5 Omezení implementace programu **4**

6 Testovani **4**

7 Závěr **5**

8 Literatura **5**

1 Úvod

Projekt byl zaměřen na extrakci informací o TCP tocích ze souboru PCAP a jejich odeslání ve formátu NetFlow v5 do síťového kolektoru. Program pracuje se soubory PCAP, které obsahují zachycený síťový provoz, a na základě těchto dat identifikuje a sloučí toky TCP. Tyto informace budou poté odeslány do kolektoru NetFlow.

2 Návod na použiti

Vstupní parametry

- -h : Napoveda.
- *pcap file path* : Cesta k souboru PCAP, který obsahuje zachycený síťový provoz.
- *host* : Adresa kolektoru, kam budou odesílány zprávy.
- *port* : Port kolektoru pro příjem zpráv.
- -a *active timeout* : Aktivní timeout.
- -i *inactive timeout* : Inaktivní timeout.

Příklad spuštění

- ./p2nprobe 192.168.1.100:2055 /path/to/traffic.pcap -a 30 -i 60

Tento příkaz načte soubor /path/to/file.pcap, zpracuje TCP toky a odešle je na kolektor na IP adrese 192.168.1.100 na portu 2055. Aktivní timeout bude 30 sekund a inaktivní timeout 60 sekund.

3 Návrh

Po prostudování principu fungování NetFlow a rozdělení na pakety jsem definovala následující části vývoje projektu:

1. Zpracování vstupních argumentů. Zahrnuje parsování vstupu, kontrolu argumentů a převod hostname na IP adresu (funkce get-host-by-name převzata z projektu IPK).
2. Zpracování paketů ze souboru PCAP. Pomocí cyklu procházíme jednotlivé pakety. Filtrujeme TCP pakety a z TCP a IP hlaviček získáváme informace potřebné pro vytváření flow.
3. Kontrola časů existujících flow. Na základě časové hodnoty z paketu a uživatelem nastavených časovačů (aktivní a neaktivní timeout) ověřujeme, zda nevypršela platnost existujících flow. Pokud ano, tyto toky jsou odstraněny z hlavní tabulky flow (aby aktuálně zpracovávaný paket nebyl přidán k expirovanému toku) a přidány do seznamu flow pro export.
4. Identifikace odpovídajícího toku. Určujeme, zda existuje tok se stejnými parametry (src IP, dst IP, src port, dst port) jako aktuálně zpracovávaný paket. Pokud ano, přidáme paket k existujícímu toku (aktualizujeme informace, jako je počet paketů, čas posledního paketu apod.). Pokud podobný tok neexistuje, voláme funkci, která jej vytvoří.
5. Export zbývajících flow po zpracování souboru PCAP. Jakmile je soubor PCAP kompletně zpracován, program exportuje všechny zbývající toky.
6. Uvolnění prostředků a ukončení programu. Na závěr program uvolní všechny zdroje a ukončí svou činnost.

4 Implementace

Během vývoje jsem se setkala s několika problémy.

Určení času

Protože úkolem projektu je analyzovat PCAP soubor s již zaznamenanými pakety, a nikoliv přijímat pakety v reálném čase, bylo nutné vyřešit, jak pracovat s časovými limity (timeouty) a relativním časem. Při spuštění programu ukládám čas spuštění (boot-time) pomocí funkce gettimeofday. Poté při odesílání dat na kolektor vypočítávám čas prvního a posledního paketu v toku relativně k tomuto boot-time. Při provádění matematických operací s časem byla důležitá přesnost, proto čas převádím na mikrosekundy před kontrolou timeoutů a při výpočtu relativního času prvního a posledního paketu (funkce get-time-diff).

Export toků při vypršení časovače

Dlouho jsem přemýšlela, kdy by měl být tok exportován na kolektor při vypršení timeoutu: ihned po kontrole času toku, nebo až na konci programu? První varianta je logická z hlediska principu fungování časovačů. Avšak v zadání byla zdůrazněna nutnost co nejefektivnějšího odesílání dat na kolektor. Prioritní mi připadalo vypršení časovače, a proto po kontrole všech toků na čas okamžitě exportuji tyto toky na kolektor (v balíčcích obsahujících maximálně 30 toků).

Ukládání toků

Toky jsou uloženy v hashovací tabulce. Tento způsob jsem zvolila kvůli možnosti snadno identifikovat toky na základě hashovacího klíče, který je vytvořen z IP adres a portů zpracovávaných paketů (funkce create-hash-key).

5 Omezení implementace programu

1. **Kontrola uživatelských vstupů**
   Vstupní argumenty nejsou podrobeny detailní kontrole, protože tato funkcionalita nebyla součástí hlavního cíle úkolu, a proto není implementována.

2. **Možnosti optimalizace rychlosti**
   Program lze optimalizovat z hlediska výkonu, protože při zpracování každého paketu je nutné:

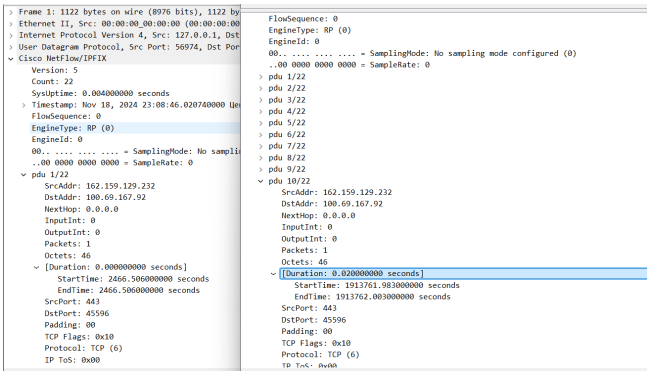
   - Projít celou tabulku toků a zkontrolovat časovače.
   - Následně znovu v téže tabulce vyhledávat tok s odpovídajícím hashovacím klíčem.

6 Testovani

Pro testování byla použita aplikace `softflowd` jako příklad NetFlow exportéru a Wireshark pro analýzu paketů. Postup byl následující:

1. Spustila jsem `softflowd`, zachytila a uložila pakety v soubor pomocí `nfdump`, a poté otevřela vytvořený PCAP soubor ve Wiresharku.

2. Pro moji aplikaci jsem provedla podobné kroky a porovnávala výsledky s těmi, které byly získány pomocí `softflowd`.



Bohužel `softflowd` není příliš vhodný pro testování aktivních a neaktivních časovačů, a proto byla kontrola exportu po vypršení času provedena ručně a nebyla příliš detailní. Nicméně, všechny ostatní parametry (počet paketů v toku, počet bajtů, IP adresy a další) byly správné a v souladu s výsledky z `softflowd`.

Během vývoje jsem narazila na problém, kdy u některých toků, které obsahují pouze jeden paket, jsou čas pro start a end různé. Moje aplikace v případě takového toku přiřazuje stejné hodnoty pro start a end čas, což se mi zdá správnější z hlediska logiky NetFlow.

Byla také ověřena funkčnost programu na serveru `Merlin`. Program byl úspěšně zkompilován a spuštěn na serveru.

7 Závěr

Během vývoje programu jsem se naučila, jak efektivně zpracovávat a exportovat NetFlow toky z PCAP souborů, jak implementovat časovače pro správu toku a jak odesílat data na NetFlow kolektor pomocí protokolu UDP. Dále jsem se seznámila s principy analýzy a filtrace paketů v PCAP souborech, konkrétně s analýzou TCP paketů, a jak tyto informace přetvářet na NetFlow exporty.

8 Literatura

Literatúra

[1] Cisco Systems. (n.d.). *NetFlow Collection Engine 5.0.3 User Guide*. Retrieved from `https://www.cisco.com/c/en/us/td/docs/net_mgmt/netflow_collection_engine/5-03/user/guide/format.html#wp1005625`

[2] Google Inc. (n.d.). *SoftFlowd: NetFlow Exporter*. Retrieved from `https://code.google.com/archive/p/softflowd/`

[3] Kaseya. (n.d.). *How to view NetFlow in WireShark*. Retrieved from `https://helpdesk.kaseya.com/hc/en-gb/articles/115003522631-How-to-view-NetFlow-in-WireShark`