Dokumentácia k projektu v predmete ISA
Varianta: Monitoring SSL spojení

Autor:

Peter Hudeček (xhudec34)

Dátum:

18.11.2020

Obsah

1 Úvod

2 Teória

3 Implementácia

4 Testovanie

5 Zdroje

1 Úvod

Cieľom projektu bolo vytvoriť program v jazyku C / C++ , ktorý bude slúžiť ako nástroj pre zobrazenie informácií o SSL komunikáciách z pcap súboru alebo live zachytávania.

Spustenie sslsniff vyžaduje root oprávnenie:

-Spustenie pre výpis komunikácií zo súboru:

sudo ./sslsniff -r nazov_suboru

-Spustenie pre výpis z live zachytávania paketov:

sudo ./sslsniff -i nazov_sietoveho_rozhrania

Príklady výstupov:

```
[user@localhost Documents]$ sudo ./sslsniff -r test.pcapng
2020-10-13 00:36:48.250934,10.0.2.15,37262,216.58.201.68,www.google.com,3129,9,0
.049670
2020-10-13 00:36:48.442245,10.0.2.15,37266,216.58.201.68,www.google.com,3128,9,0
.044605
2020-10-13 00:36:48.545851,10.0.2.15,37268,216.58.201.68,www.google.com,3128,9,0
.064215
2020-10-13 00:36:48.639927,10.0.2.15,37270,216.58.201.68,www.google.com,3130,9,0
.058713
```

```
[user@localhost Documents]$ sudo ./sslsniff -i enp0s3
2020-11-18 19:54:49.576845,10.0.2.15,43644,77.75.74.134,d32-a.sdn.cz,4599,20,0.2
66000
2020-11-18 19:54:49.576673,10.0.2.15,43642,77.75.74.134,Qa2-a.sdn.cz,4599,20,0.2
66249
```

2 Teória

SSL/TLS protokol sa nachádza medzi transportnou a aplikačnou vrstvou a delí sa na vyššiu a nižšiu podvrstvu. Vyššia vrstva sa delí na 4 pod-protokoly:

- Handshake protokol:
  - zabezpečuje vzájomnú autentizáciu klienta a servera, výmenu kľúča pre enkryptovanie komunikácie
- ChangeCipherSpec protokol
  - zabezpečuje aby parametri dohodnuté v handshake-u boli aplikované
- Alert protokol
  - používa sa pre prenos výnimiek a potencionálnych problémov
- Application Data protokol
  - prenáša data z aplikačnej vrstvy cez bezpečný kanál

Nižšia vrstva sa skladá z 3 informácií:

- Typ TLS záznamu
- Verzia TLS
- Dĺžka záznamu

3 Implementácia

Pri implementácii bola pre zachytávanie TCP komunikácie využitá inšpirácia z minuloročného projektu v predmete IPK = ipksniffer.c , táto implementácia využíva hlavne knižnice pcap a netinet pre zachytávanie a parsovanie paketov. Pre spracovanie parametrov z štandardného vstupu bola použitá knižnica getopt. Pre uloženie parametrov zo štandardného vstupu bola vytvorená štruktúra ***isaSettings***. Pri spustení bola zvolená možnosť spustiť program iba s jednou možnosťou, buď **-r** pre spustenie zo záznamu z pcap súboru alebo **-i** pre live spustenie pre názov sieťového zariadenia, iné spustenie vyvolá chybu. Pre ukladanie záznamov o komunikáciách bola vytvorená štruktúra ***communication*** a následne vektor pre uloženie všetkých týchto komunikácií. Tento vektor je implementovaný ako globálny z dôvodu nemožnosti pridania ďalšieho parametra pre callback funkciu pre *pcap_loop()* viz nižšie.

Implementované funkcie:

- `struct isaSettings setSettings(struct isaSettings s, int argc, char *argv[]);` - Funkcia overí a spracuje parametre zo štandardného vstupu a uloží ich ako nastavenie do štruktúry isaSettings, funkcia taktiež otvára pcap súbor / live komunikáciu.

- `void process_transport_layer(u_char *args, const struct pcap_pkthdr *header, const u_char *buffer)` - Funkcia pracuje ako callback funkcia pre `pcap_loop()`. Funkcia parsuje ip hlavičku a tcp hlavičku. Pri novom tcp handshake-u ukladá dáta o komunikácii.Pre spracovanie TLS komunikácie je z tejto funkcie volaná funkcia `process_tls_header()`. Pri ukončení tcp komunikácie zavolá funkciu `print_communication()`.

- `void print_communication(communication comm,communication end);` - Funkcia vypíše údaje o TLS komunikácii na štandardný výstup. Vo vnútri funkcie je taktiež zabezpečený výpočet trvania komunikácie a konverzia času.

- `void process_tls_header(const u_char *data, int dataSize, int commindex, struct communication);` - Funkcia spracováva TLS časť paketu a ukladá informácie o komunikácii do momentálnej komunikácie = komunikácia na pozícii `commindex`, vo vektore communications.

Problematická časť:

Medzi časť implementácie, ktorú chcem podrobnejšie vysvetliť patrí časť uvedená v obrázku nižšie. Daná časť slúži ako kontrola TLS hlavičky pre správnu TLS verziu, získanie dĺžky v danom pakete a následné spracovanie podľa typu pod-protokolu. Táto implementácia môže pôsobiť neznámemu oku mätúco, avšak je veľmi jednoduchá. V cykle sa posúvame postupne po bytoch cez časť paketu za tcp hlavičkou. Ak narazíme na indikátor jedného z typu TLS pod-protokolov, skontroluje sa jeho verzia a následne sa uloží počet bytov do štruktúry o zázname danej komunikácie

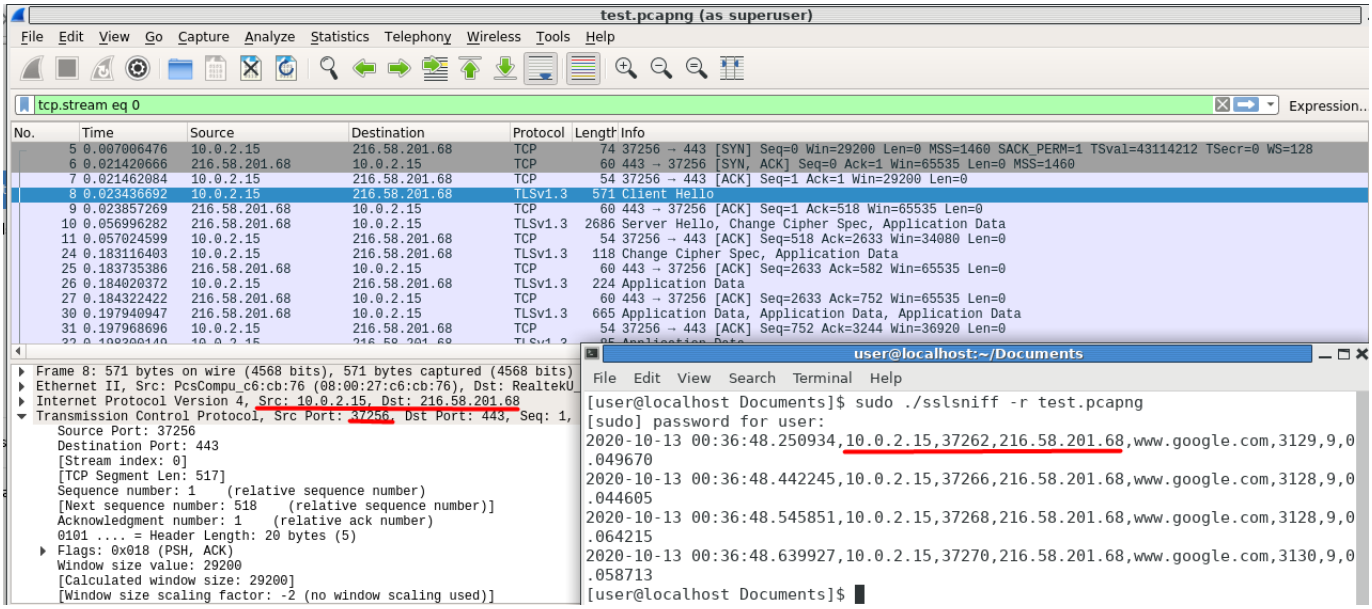
```
int datapointer = 0;

while(datapointer < dataSize )
{
    switch(data[datapointer]){

        case CHANGE_CIPHER_SPEC:
            datapointer++;
            if(data[datapointer] == 3){
                datapointer++;
                if(data[datapointer] == 0 || data[datapointer] == 1 || data[datapointer] == 2 || data[datapointer] == 3 || data[datapointer] == 4){
                    datapointer++;
                    currPkt.bytes = (unsigned long)(data[datapointer]<<8)+data[datapointer+1];
                    communications[commindex].bytes = communications[commindex].bytes + currPkt.bytes;
                }
            }
            break;
```

4. Testovanie

Program sslsniff bol testovaný s pomocou využitia open-source analyzátora paketov **WireShark**. Zachytené TLS komunikácie boli porovnávané s TLS komunikáciami zachytenými programom WireShark. Pre zobrazenie TLS komunikácie v programe WireShark bola využitá možnosť Follow TCP-Stream, kde sa následne manuálne kontroloval počet bytov, paketov, správny čas a ostatné parametre. V obrázku nižšie je zobrazený spomínaný postup.



5 Zdroje

1. **Dokumentácie použitých knižníc**
2. **Traffic Analysis of an SSL/TLS Session - Autor:** Álvaro Castro-Castilla **Publikované:** 23. 12. 2014 http://blog.fourthbit.com/2014/12/23/traffic-analysis-of-an-ssl-slash-tls-session/
3. **Manpage of PCAP - Autori:** The Tcpdump Group. **Publikované:** 29. 01. 2020 https://www.tcpdump.org/manpages/pcap.3pcap.html