Estensione Lua per Wireshark

Un plug-in per analizzare anomalie a livello trasporto in una cattura

Progetto realizzato da Irene Trevisi

Sommario

1

Lo scopo

Questo tool ha come principale obbiettivo di aiutare gli analisti individuando gli host che potrebbero essere coinvolti in problemi (lentezza o interruzioni di servizio) legati alla rete, esaminando le informazioni a livello trasporto ed i messaggi icmp.

La tecnologia usata

Il plugin è stato realizzato con il linguaggio di scripting interpretato Lua, integrato in Wireshark.

Lo sviluppo e il testing è stato fatto con Wireshark Version 3.6.8 (v3.6.8-0-gd25900c51508), con Lua 5.2.4.

Come usare il tool

Il file con estensione .lua viene caricato ad ogni avvio di wireshark se viene posizionato nella <global configuration directory> o nella <personal plugin directory> se Lua è abilitato.

Il path della personal plugin folder è %APPDATA%\Wireshark\plugins su sistemi windows e ~/.local/lib/wireshark/plugins per sistemi unix-like.

Si rimanda alla documentazione di Lua in Wireshark per ulteriori dettagli https://wiki.wireshark.org/Lua, e all'appendice B per i path alle cartelle https://www.wireshark.org/docs/wsug_html/#AppFiles.

Sono forniti dei file pcap di test, che coprono diversi scenari.

Il funzionamento

Ogni pacchetto viene analizzato:

- Per i flussi tcp vengono conteggiati i pacchetti duplicati e le ritrasmissioni in ciascuna direzione
- Per i flussi udp vengono ricercate le risposte, se presenti, o i messaggi di icmp error reporting
- Per i messaggi icmp viene analizzato lo scambio di richieste/risposte, o la presenza di error reporting message

Una volta analizzati tutti i pacchetti i dati raccolti vengono nuovamente analizzati per trovare i flussi che rappresentano un'anomalia, infine i dati vengono ulteriormente aggregati per una stampa sintetica.

Il codice è dettagliatamente commentato, e fa parte della documentazione.

L'output

Il tool di default stampa un output minimale, ma è possibile scegliere due livelli più dettagliati.

Normal mode

Di seguito è riportato un esempio di output minimale

```
queste sono le comunicazioni soggette a anomalie
Completezza TCP: 1 : SYN - 2 : SYN-ACK - 4 : ACK - 8 : DATA - 16 :
FIN - 32 : RST
192.168.113.2 89.46.110.70 [1]
     tcp 55181 443 -> [3] | incomplete 1[2]
     tcp 55182 443 -> | incomplete 1[4]
     icmp ->

192.168.113.2 8.8.8.8
     udp 57185 53 <- no pkt [5]
     udp 64385 53 <- no pkt | -> icmp [6]
```

[1] Vengono stampati gli ip degli host coinvolti della comunicazione
[2] Per ciascuna tipologia di conversazione intercorsa tra gli host (per la quale è stata riscontrata un'anomalia) viene stampata una riga: tcp, udp, icmp. Per tcp e udp vengono stampate anche le porte.
[3] è evidenziata con una freccia ciascuna direzione in cui si sono riscontrate anomalie, nel caso di tcp indica ritrasmissioni o pacchetti duplicati
[4] per tcp viene stampata la completezza calcolata da wireshark per lo stream
[6] per udp si indica l'assenza di pacchetti in una direzione (quella indicata dalla freccia)
[7] o la presenza di messaggi di error reporting icmp per il verso indicato dalla freccia

Verbose mode

Alla stampa dell'out normale viene preposta la stampa dei flussi per i quali sono state riscontrate anomalie, ed una loro aggregazione (dalla quale vengono calcolate le frecce della stampa minimale)

```
TCP l'analisi dei pacchetti tcp che hanno dato luogo ad anomalie
1: l'indice di stream
     reverse_retransmission: 0
     direction_dup_ack: 0
     ip_srg: 192.168.113.2
     completeness: 1
     port_dst: 443
     direction_retransmission: 4
     ip_dst: 89.46.110.70
     reverse_dup_ack: 0
     port_srg: 55182

0:
     reverse_retransmission: 0
     direction_dup_ack: 0
     ip_srg: 192.168.113.2
     completeness: 1
     port_dst: 443
     direction_retransmission: 6
     ip_dst: 89.46.110.70
     reverse_dup_ack: 0
     port_srg: 55181
```

I dati raccolti per lo stream

UDP *l'analisi dei pacchetti udp che hanno dato luogo ad anomalie*

192.168.113.28.8.8.85364385: *la combinazione ordinata di ip e porte*

```
     reverse_icmp: false
     datagram: 2
     ip_srg: 192.168.113.2
     ip_dst: 8.8.8.8
     icmp: true
     port_dst: 53
     reverse_datagram: 0
     port_srg: 64385

192.168.113.28.8.8.85357185:
     reverse_icmp: false
     datagram: 2
     ip_srg: 192.168.113.2
     ip_dst: 8.8.8.8
     icmp: false
     port_dst: 53
     reverse_datagram: 0
     port_srg: 57185
```

I dati raccolti per la conversazione assumendo che dati gli host h1, h2 e le porte p1, p2, i pacchetti da h1:p1 -> h2:p2 e quelli da h2:p2->h1:p1 fanno parte della stessa comunicazione

ICMP *l'analisi dei pacchetti icmp che hanno dato luogo ad anomalie*

```
192.168.113.289.46.110.70:
     no_resp: false
     ip_srg: 192.168.113.2
     ip_dst: 89.46.110.70
     reply: false
     err: true
     request: false
```

Sono evidenziate la presenza di request/response icmp, o errori

questi sono i dati grezzi per i quali sono state rilevate anomalie:

192.168.113.289.46.110.70: *gli host coinvolti in un'anomalia rilevata*

tcp: *l'analisi tcp ha rilevato un'anomalia*

44355181: *le porte su cui è avvenuta la comunicazione*

```
          inbound: false
          port_srg: 55181
          incomplete: 1
          port_dst: 443
          outbound: true
     44355182:
          inbound: false
          port_srg: 55182
          incomplete: 1
          port_dst: 443
          outbound: true
ip_srg: 192.168.113.2
ip_dst: 89.46.110.70
icmp: true
reverse_icmp: false
```

Indica la direzione in cui sono state riscontrate anomalie, interpretando inbound ed outbound rispetto a ip_sorgente e destinazione

Gli ip degli host

Rilevazione di anomalie su icmp

```
     udp:
192.168.113.28.8.8.8:
     tcp:
     ip_srg: 192.168.113.2
     ip_dst: 8.8.8.8
     icmp: false
     reverse_icmp: false
     udp: l'analisi udp ha rilevato un'anomalia

          5357185:
               inbound_icmp: false
               inbound: true
               port_srg: 57185
               port_dst: 53
               outbound_icmp: false
               outbound: false
          5364385:
               inbound_icmp: false
               inbound: true
               port_srg: 64385
               port_dst: 53
               outbound_icmp: true
               outbound: false
```

Indica la direzione in cui sono state riscontrate anomalie, interpretando inbound ed outbound rispetto a ip_sorgente e destinazione, oppure se un pacchetto trasmesso ha dato luogo ad un error reporting message icmp

Debug mode

In questa modalità, oltre a quanto già stampato in modalità verbose, vengono stampati tutti i flussi elaborati, anche se non hanno evidenziato anomalie.

Le strutture dati utilizzate

Sono state utilizzate diverse strutture dati atte a memorizzati i dati rilevanti in ciascuna fase dell'analisi.

L'analisi TCP

la chiave della struttura è il campo tcp.stream, calcolato da wireshark per ogni flusso riconosciuto come tale.

Il primo pacchetto che viene elaborato per ciascun flusso definisce l'ip sorgente e l'ip destinazione, affinché sia possibile tenere traccia degli errori calcolati sulle due direzioni.

Le informazioni vengono mantenute in una tabella lua così costruita:

```
{
     ip_srg=<una rappresentazione dell'ip sorgente>,
     ip_dst=<una rappresentazione dell'ip destinazione>,
     port_srg =<una rappresentazione della porta sorgente>,
     port_dst =<una rappresentazione della porta di destinazione>,
     completeness = <il valore di completezza calcolato da wireshark per
lo stream>,
```

```
	direction_dup_ack =0, //incrementato nel Corso dell'analisi
	direction_retransmission = 0, //incrementato nel Corso dell'analisi
	reverse_dup_ack =0, //incrementato nel Corso dell'analisi
	reverse_retransmission = 0 //incrementato nel Corso dell'analisi
}
```

L'analisi UDP

La chiave è una combinazione ordinata di ip e porte, così da identificare una comunicazione tra gli stessi hosts:porte ina base alla definizione.

Il primo pacchetto che viene elaborato per ciascun flusso definisce l'ip sorgente e l'ip destinazione, affinché sia possibile tenere traccia degli errori calcolati sulle due direzioni.

```
{
	ip_srg=<una rappresentazione dell'ip sorgente>,
	ip_dst=<una rappresentazione dell'ip destinazione>,
	port_srg =<una rappresentazione della porta sorgente>,
	port_dst =<una rappresentazione della porta di destinazione>,
	datagram = 0; //incrementato nel Corso dell'analisi
	reverse_datagram=0, //incrementato nel Corso dell'analisi
	icmp = false,
	reverse_icmp=false
	//qualora una comunicazione ip_srg:port_srg->ip_dst:port_dst causi
la ricezione id un icmp error report messagge viene settato il valore di
icmp, analogamente per reverse_icmp per report ai pacchetti da
ip_dst:port_dst ->ip_srg:port_srg
}
```

L'analisi ICMP

La chiave è una combinazione di ip sorgente e ip destinazione, nell'ordine della comunicazione analizzata dal protocollo.

```
{
	ip_srg=<una rappresentazione dell'ip sorgente>,
	ip_dst=h2,
	no_resp =<una rappresentazione dell'ip destinazione>,
	request = false,
	reply = false,
	err = true
```

```
      //man mano che vengono analizzati i pacchetti icmp tra gli host srg
e dst o si ricevono errori vengono aggiornati i campi

}
```

L'elaborazione finale

Viene calcolata una quarta struttura, per aggregare i risultati delle analisi fatte su tcp, usp e icmp.

La chiave è una chiave ordinata degli ip degli host coinvolti nella comunicazione.

Viene memorizzato ip sorgente ed ip destinazione del primo flusso aggiunto per ciascuna chiave, e serviranno come riferimento per le direzioni.

È una tabella multidimensionale, per ciascuna conversazione tcp e udp viene aggiunto un array che da informazioni su quella conversazione bidirezionale.

```
{
      ip_srg=tostring(value["ip_srg"]),
      ip_dst=tostring(value["ip_dst"]),
      tcp=  {
            port_srg=value["port_srg"],
            port_dst=value["port_dst"],
            outbound=outgoing and true,
            inbound=(not outgoing) and true,
            incomplete = incomplete
            },
      udp=  {
            port_srg=value["port_srg"],
            port_dst=value["port_dst"],
            outbound=ingoing,
            inbound=outgoing,
            outbound_icmp=ingoing_icmp,
            inbound_icmp=outgoing_icmp
            },
      icmp=false,
      reverse_icmp=false
}
```

Il contenuto di questa tabella viene stampato in verbose e debug mode.