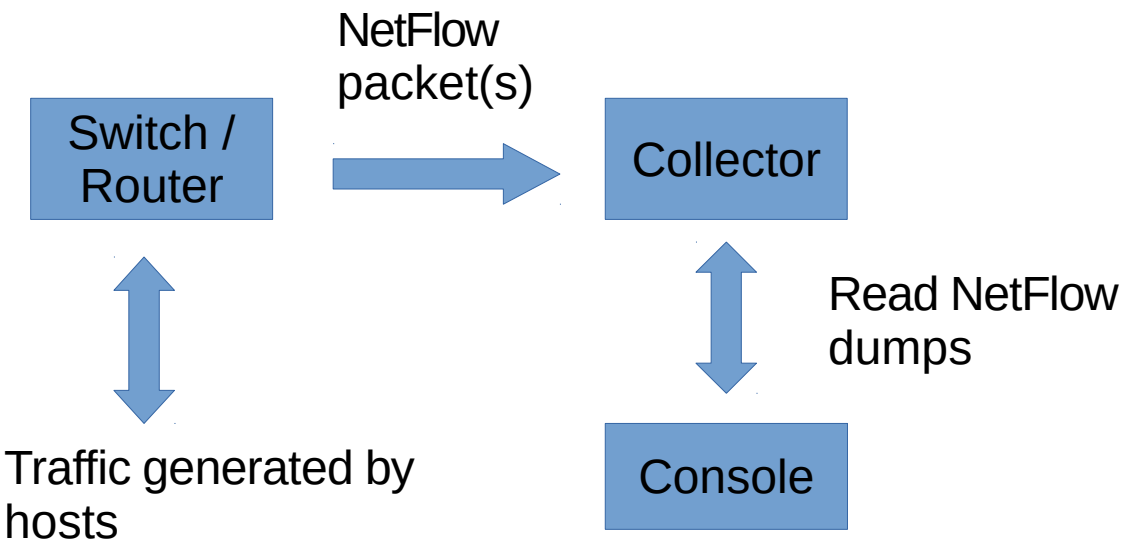
NetFlow Analysis: Detecting covert channels on the network

Detecting malicious traffic by using NetFlow data

By: Joey Dreijer, Student OS3

Gathering NetFlow data

- Router/Switch sends flow stats to external collector
- Collector receives and stores flow details
- Parser/interface reads flow from collector dump



NetFlow in short

- NetFlow data not just a 'term'
  - NetFlow (v9) specified in RFC3954
  - NetFlow commonly used from v5 and up

- NetFlow standardized to sent 'flow' characteristics
  - Stats such as bytes, packet number, port, session timer
  - Implemented in different (multi-vendor) routers/switches
  - Does not include packet content
  - ***Request and response two different flows***
  - Often used for network performance measurement

Data required for research

- NetFlow collector stored the following details (using v5):
  - Source Address
  - Destination Address
  - Source Port
  - Destination Port
  - (TCP Flags)
  - Bytes send
  - Packets send
  - Time

```
Date flow start          Duration Proto     Src IP Addr:Port          Dst IP Addr:Port   Packets    Bytes Flows
2014-06-30 19:45:39.253   116.103 TCP         10.0.2.15:50494 ->     62.69.166.15:80          46     6442     1
2014-06-30 19:45:39.253   116.103 TCP       62.69.166.15:80    ->        10.0.2.15:50494       47    42669     1
2014-06-30 19:45:39.375   115.985 TCP         10.0.2.15:33675 ->    74.125.136.94:80          8     1142     1
2014-06-30 19:45:39.375   115.985 TCP      74.125.136.94:80    ->        10.0.2.15:33675        7      640     1
2014-06-30 19:45:39.395   115.961 TCP         10.0.2.15:46931 ->     62.69.166.18:80          11     2230     1
```

Note: NetFlow v5 is dinosaur old. Use v9 or IPFIX instead for more stats.

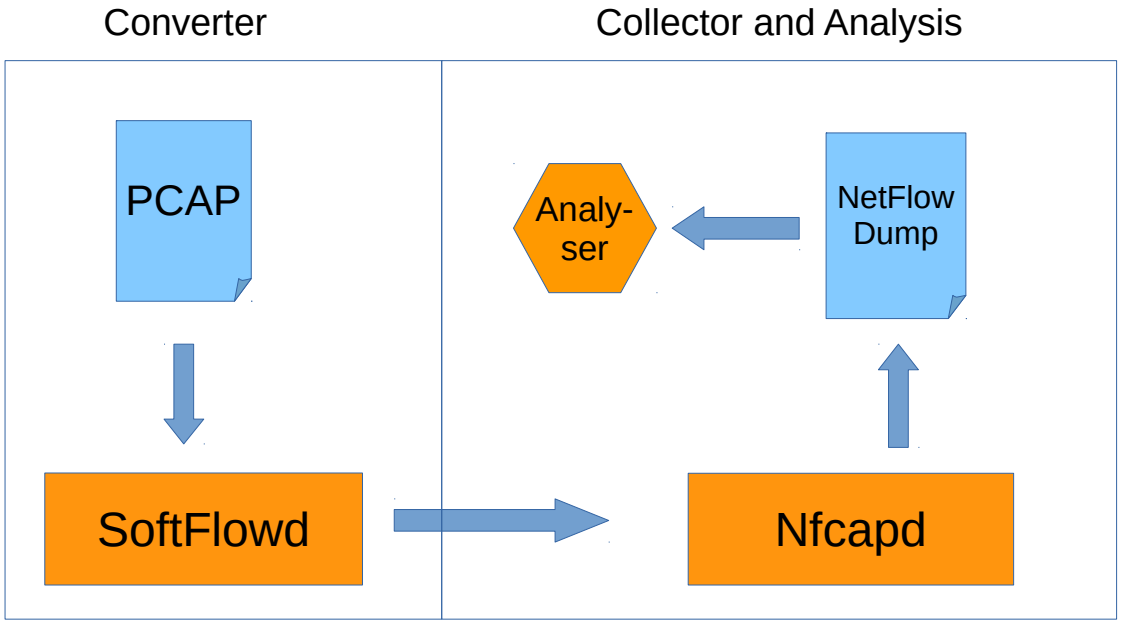
Data required for research

- Combining request/response to get the following data:
  - Source Address
  - Destination Address
  - Source Port
  - Destination Port
  - (TCP Flags)
  - ***Bytes Incoming***
  - ***Bytes outgoing***
  - ***Packets incoming***
  - ***Packets outgoing***
  - Average session time

Date flow start	Duration	Proto	Src IP Addr:Port		Dst IP Addr:Port	Out Pkt	In Pkt	Out Byte	In Byte	Flows
2014-06-30 19:45:39.395	115.961	TCP	10.0.2.15:46931	<->	62.69.166.18:80	10	11	2550	2230	2
2014-06-30 19:45:39.375	115.985	TCP	10.0.2.15:33675	<->	74.125.136.94:80	7	8	640	1142	2
2014-06-30 19:45:39.396	115.961	TCP	10.0.2.15:46932	<->	62.69.166.18:80	7	8	712	1517	2

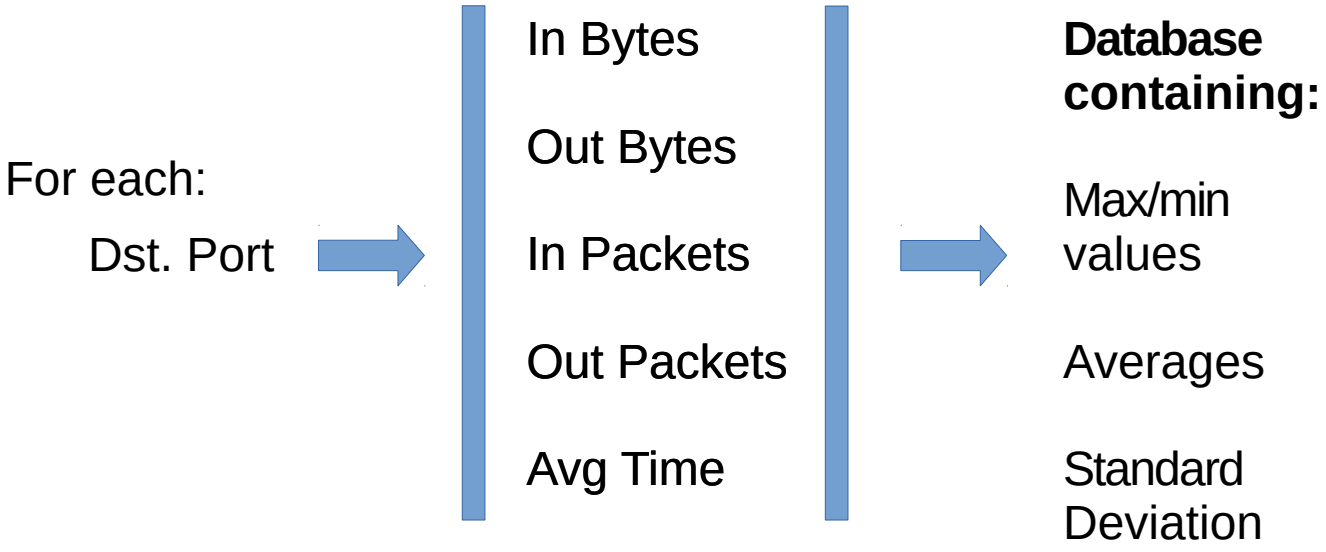
Collecting NetFlow data

- SoftFlowd sends NetFlow data to collector (nfcapd). Optional: Pcap or Interface as input
- NetFlow data stored in binary format
- Format parsed by Python wrapper and nfdump (custom patched pynfdump_altered)



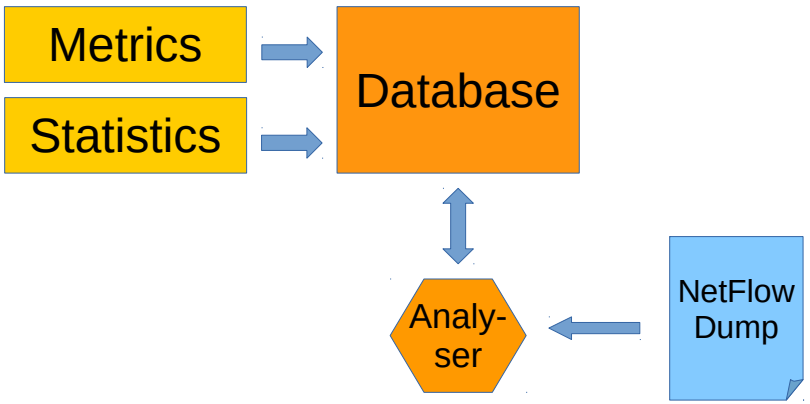
Initial protocol analysis

- Gathering 'known-good' traffic
- Generating 'known-bad' traffic
  - Comparing differences / similarities
  - Storing usefull comparison data



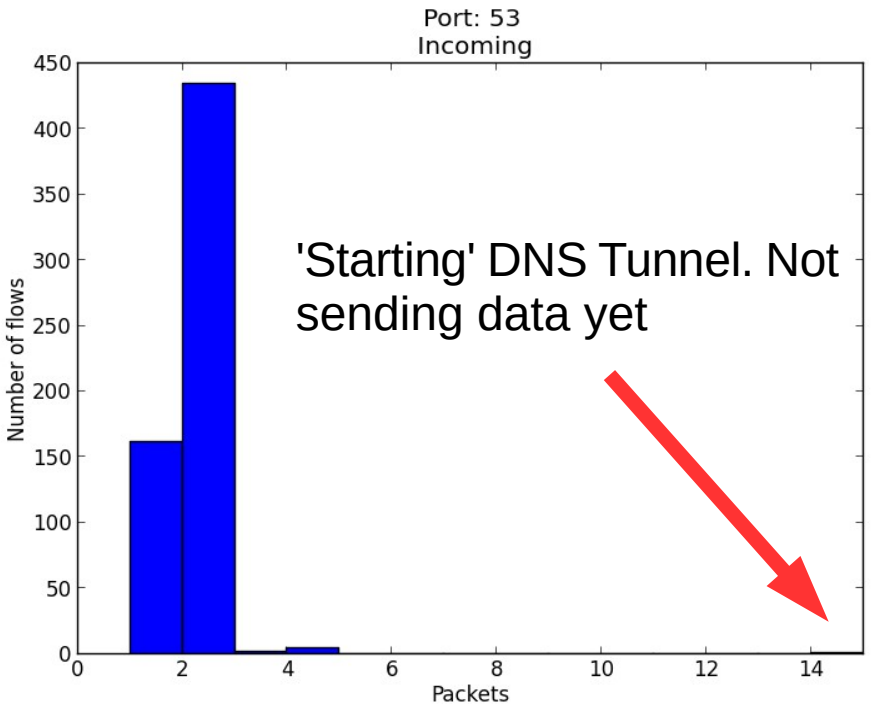
Comparing NetFlow data

- Traffic analysis; comparing 'real-time' binary (nfdump) vs stored (MySQL)
- 'Anomaly detection' based on selected metrics/profile
- Maximum range via standard deviation
  - **Note:** Only *if* possible. Not all traffic can be normalized

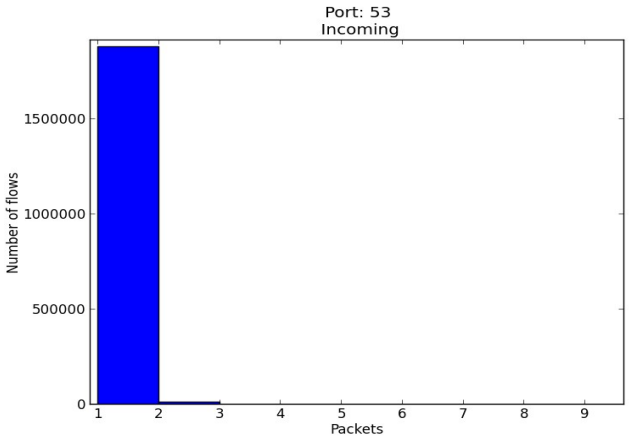


Detecting Tunnels / Covert Channels

- **Example 1:** DNS Tunnels
- DNS may have 'normal behaviour'
- Tunneling via DNS abnormal statistics based on metric x?
- Verify differentation per metric

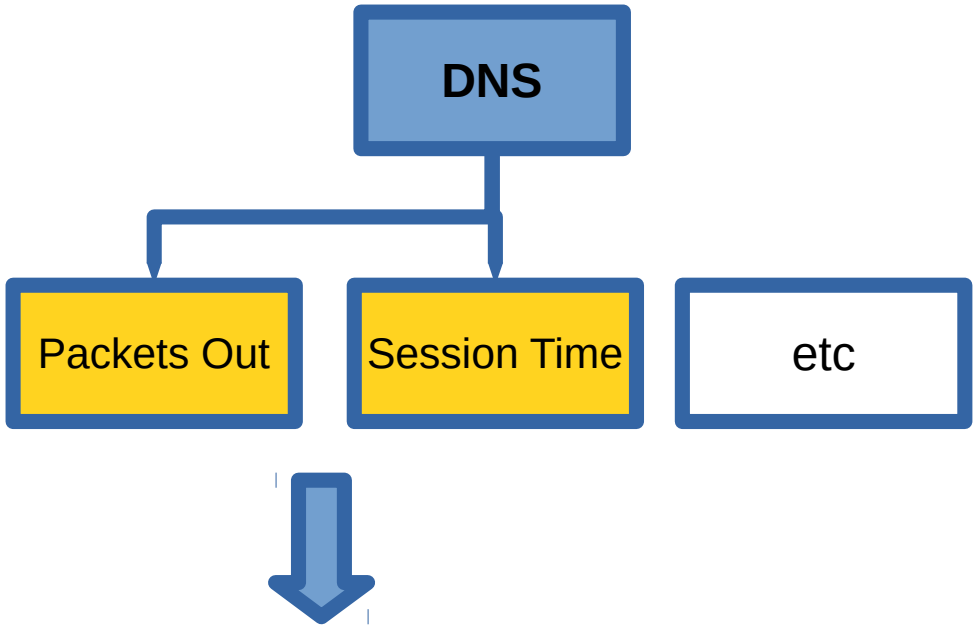


Compared to +- 2 million DNS Flows



Detecting Tunnels / Covert Channels

- Previous examples done via anomaly detection
- Known-good database used as reference
- Pre-defined profile (ie. alert only if packets **and** time mismatch by x)



anomaly = (max difference * standard deviation) + average

If *anomaly* **is larger** than **current** *flow:*
If *packetAnomaly* **and** *timeAnomaly:*
Generate Alert

Detecting Tunnels / Covert Channels

- Why are multiple metrics important? (and/and policy)
- NetFlow parser shows incorrect flows with much traffic
- True automated anomaly detection shows many FP's

- **Example:**

  10.10.0.2:50001 → 8.8.8.8:53

  Packets: 4, time: 4001 seconds (….?)

- Actually 2 DNS requests on different times
  - However, identical source port and destination lets 'nfdump' think it is the same flow –> results in False Positive

Detecting Tunnels / Covert Channels

- Comparing with realistic dataset
- 17 million flows from GuestNet
  - Literal flow dump, can contain 'malicious' flows
  - Both bad and good traffic?

- 2 million DNS responses
  - Results in 0,0005% hits based on combined metrics
    - Includes previous 'bug' with multiple sessions combined due to identical ports and destinations
    - Uncertain if actual tunnels inside dump

Other uses

- **Example 2:** NMAP Scan
- Aggregated NetFlow shows requests and response
- NetFlow shows flow with no responses for filtered ports

- Probability 'x' amount of ports do not reply within 'y' amount of time based on 'z' amount of retries/packets

```
2014-07-01 12:42:33.146    0.000 TCP        10.0.2.15:57693 <->  145.100.104.55:3000       0       1       0       60     1
2014-07-01 12:42:31.408    0.000 TCP        10.0.2.15:36016 <->  145.100.104.55:9595       0       1       0       60     1
2014-07-01 12:42:33.222    0.000 TCP        10.0.2.15:57954 <->  145.100.104.55:33         0       1       0       60     1
2014-07-01 12:42:32.474    0.000 TCP        10.0.2.15:57230 <->  145.100.104.55:1248       0       1       0       60     1
2014-07-01 12:42:30.242    0.000 TCP        10.0.2.15:39538 <->  145.100.104.55:1055       0       1       0       60     1
2014-07-01 12:42:33.220    0.000 TCP        10.0.2.15:60249 <->  145.100.104.55:1075       0       1       0       60     1
2014-07-01 12:42:32.207    0.000 TCP        10.0.2.15:39512 <->  145.100.104.55:1044       0       1       0       60     1
2014-07-01 12:42:32.763    0.000 TCP        10.0.2.15:59968 <->  145.100.104.55:255        0       1       0       60     1
```

Other uses

- Small problem with portscans....

- Nfcapd holds a default 5 minute NetFlow cache
- Not all flows stored after cache timer
  - Waits for finished sessions before storing flow
  - Half open TCP sessions will be cached untill timeout
  - Timeout can last 20 minutes depending on config

DEMO

Conclusion

- NetFlow only sends limited amount of information
  - Does not say anything about packet contents
- Fairly easy to detect 'well-know' and publicly available tunnels and scans
- Covert Channels / tunnels always possible; attacker has all the time in the world.
  - Craft pingtunnel to send fixed size packets every second to conform the 'default' behaviour.