DDoS Detection

How to know if you are attacked or partake in an attack

Klaus Möller

WP8-T1

Webinar, 15th of February 2021

Public

www.geant.org
What we will cover today

• Introduction to the detection task
• Sensors used in DDoS detection
  – Short Introduction to NetFlows
  – Example of a detection system: NeMo
• Detection
  – Workflow
  – Structured Traffic Analysis
• Traffic Details
  – Control Server, Bots, D(R)DoS
  – Backscatter
Introduction to Detection
Challenges/Obstacles in DDoS Detection

- Sensor needs to be in path of the traffic type to be detected
- Distinguishing malicious traffic (C&C, D(R)Dos) from legitimate
  - Low false positive rate
- Reliable detection
  - Low false negative rate
- Timely
  - No use if too late
- Actionable
  - Results must allow mitigation or other useful action

Critical for acceptance and usability!
Sensors

www.geant.org
Sensor Placement

- **ISP**: Ingress/egress points into network
  - At least the most important ones (better all of them)
  - Alternatively: Core links/routers (fewer sensors needed)
- **Victim network**: Link(s) to ISP(s)
  - Sometimes only link to vital on-premise servers
- **Placement dictated by available resources**
  - Processing power, bandwidth, memory, or bus-slots in routers/switches
  - Rack space (mitigation needs a lot more)
  - Ultimately a question of available budget
Sensor Types

- **Packet sniffers** – tcpdump, wireshark, etc.
  - 1:1 copy of network packets, huge amounts of data
- **Flow data** – NetFlow, sFlow, Argus, AppFlow, NetStream, etc.
  - Reduced amount of data, but still usable for accounting and security purposes
- **Various values read from system or SNMP MIB**
  - CPU load, bandwidth used, error rates, queue usage, etc.
- **Miscellaneous data**
  - Routing tables
  - Customer Relationship Management (CRM): contacts, billing, etc.
  - Cabling, system location, hardware information, etc.
NetFlow

- Traffic is observed by **probes** at **observation points (IPFIX)**
  - Can be dedicated hardware probes, but often build into routers and switches
- Data from probes is aggregated by the **exporter** that sends flow records to a **collector** that stores the flow records data while the **analysis application** analyzes the traffic in the context of intrusion detection, traffic profiling, etc.
- Protocol for the data exchange between exporter and collector has been standardized as NetFlow (RFC 3954)
  - Later standard that builds on NetFlow: IP Flow Information Export (IPFIX, RFC 7011/12)
  - Storage format is **not** standardized (but conversion-tools exist)
(Net)Flow Records

- **Flow:** any number of packets observed in a specific time slot and sharing a number of properties
  - Source & destination IP address
  - IP protocol number (e.g. ICMP, TCP, UDP, etc.)
  - TCP/UDP/SCTP source & destination port numbers, or ICMP type & code
  - IP Type of Service (TOS)
  - By definition: Flows are unidirectional
  - Application data (layer 5+) not part of the flow data

- **Flow record:** the above information plus
  - Number of packets & bytes seen in the timeslot
  - More data: input/output interface, AS number, next hop address and more
    - Depending on the NetFlow protocol version used
Sampled NetFlow

- Evaluating every packet consumes too many resources on high-speed links
  - Sampling reduces number of packets taken into account: 1 out of n
  - n: Sample Rate (typically 100 - 1,000,000)
  - Result is called **Sampled NetFlow**
  - Still accurate enough for a general traffic picture and DDoS detection
  - More privacy protection friendly (except for n = 1:)
  - Might not detect small, short-lived flows at larger values of n

- Do not confuse with **sFlow** (Sampled Flow, RFC 3176)
  - Samples of counters
  - (Random) samples of packets or **application operations**
NeMo - Network Monitoring

System to detect and mitigate DDoS attacks in the German NREN (DFN)

Also a GÉANT 4-3 project: WP8, Task 3.3

Also a GÉANT 4-3 project: WP8, Task 3.3

www.geant.org
NeMo - Alarm Analysis GUI
Detection

www.geant.org
Detection Workflow – Base lining

- If you don’t know what’s normally going on in your network
  - How will you ever know when something unusual happens?
  - When things stop working/people complain?
  - It’s too late to start base lining then

- Even when outsourcing or automating (AI), an overview is needed
  - How else will you know if you’re being ripped of or what the AI is learning?

- Know your network, esp. traffic distribution
  - Most active source and destination IP addresses (“top talkers”)
  - Network link utilization
  - Transport & application distribution
  - Traffic changes over time – trends, recurrences (work hrs, holidays, ...)

www.geant.org
Structured Traffic Analysis 1/4: Statistics

• Protocol hierarchy breakdown
  - IPv4/IPv6, TCP, UDP, HTTP, SSH, DNS, etc.
  - Gives a first idea with what to deal (e.g. ICMP flood, UDP flood) and which service (port number) is being attacked
Structured Traffic Analysis 2/4: Size(s) matter

- Packet size distribution
  - Many small packets → possible sign of packet switching attack
  - Many large packets → possible sign of bandwidth exhaustion attack
Structured Traffic Analysis 3/4: Sessions (Flows)

• Look for sessions (flows)
  - Incoming vs. outgoing traffic
  - Top talkers (IP addresses)

• Known Good/Bad IP addresses
  - Partners/Customers
  - WoT, Shadowserver, MISP, etc.
Structured Traffic Analysis 4/4: Full packet captures

• Sometimes needed
  – Easy to get with sFlow
  – Or via port mirroring of switches or dedicated probes at critical points
  – But need to set up sensors in advance

• Gives insight into
  – Application type of attacks

• Check samples against NIDS to look for exploits of vulnerabilities
  – Zeek (Bro), Suricata, Snort, Yara, etc.

• Don’t forget decryption for TLS or VPNs

• Check with your DPO (esp. with little/shaky evidence)
Traffic Characteristics

www.geant.org
DDoS Traffic Characteristics: C&C Server

- From Attacker (via Proxy) to C&C Server
  - Traffic type may vary: HTTPS, VPN, or other

- From Bots to C&C server (cmd pull) or
  - Short lived connections (usually just one HTTP GET request)
  - Small amount of data transferred (bot cmd, bot config, sometimes code updates)
  - Server IP address may co-host legitimate websites

- From C&C server to Bots (cmd push)
  - Will need open port on the Bot
    - Traffic may be piggybacked on top of other traffic (HTTP, DNS, etc.)
  - Or reverse connection
    - Usually long-lived

- Bottom line: too hard, don’t bother, unless you have a lead to follow
DDoS Traffic Characteristics: Bots vs. Clients

• Bots to Victim traffic
  – Source IP address: Spoofed (random)
    • When source addresses are filtered: subnet of the bot or the bot itself
  – Lots of “empty” sessions:
    • Low number of packets,
    • Very little data transferred, small packets (unless flooding)

• Normal (high usage) traffic
  – Lower number of source IP-addresses
    • Often known, like backup servers, customers, partners, etc.
  – Sessions do actually transfer data - more symmetric traffic distribution
  – Is there a reason?
    • Backup time, “slashdotted/heise effect”, launch of service, ...?
DDoS Traffic Characteristics: DRDoS Traffic

•Protocols:
  – Usually ICMP or UDP - easy spoofing
  – Rarely TCP - needs application that can be triggered

•From Amplifiers/Reflectors to victim
  – Source address of amplifier is not spoofed
  – Often that of known open amplifiers (→ Shadowserver)

•From Bots to Amplifiers/Reflector
  – Bandwidth used usually not suspicious
    • Small packets
    • Bot distributes traffic across many amplifiers/reflectors
    • Unless sensor is placed in front of the reflector
DDoS Backscatter

- DDoS traffic may elicit responses from victim
  - I.e. TCP SYN-ACK packets in response to TCP SYN (floods)
  - Or ICMP unreachable, or
  - Application responses, ...
- To random IP addresses if bots spoof the source IP address
  - If not spoofed, directly back to the bots IP address
  - Responses to DRDoS traffic will go to back amplifiers/reflectors
DDoS Backscatter Detection - *Network Telescope*

- Technology used is the same as for other DDoS traffic
  - Sensors, collectors, analysers, etc.
- To distinguish from other traffic, look only for incoming traffic to unused (dark) IP addresses
  - "Darknet", if interspersed with live addresses → "Greynet"
  - Other names: "network motion sensors", "network sink", "blackhole monitor"
  - Best if IP address space was never used in production (very rare today)
  - Doesn’t need to be continuous
  - Amount of DDoS traffic seen by sensors would be proportional to the number of IP addresses covered by sensors
  - Assuming perfectly random distribution with spoofed IP addresses
DDoS Backscatter Detection - Traffic Patterns

- Source IP address is that of the victim
- Random destination IP addresses, no coherence
- Source port that of the attacked service
  - Usually port 80/tcp or 443/tcp
- Destination ports random, usually ephemeral ports (> 1023)
  - May see some “ladder” if DDoS tool uses changing port numbers
- Layer 5+ contents depend on type of DDoS
  - Will not be present in flow data - full packet captures needed
- Traffic may be from multiple DDoS techniques as attackers employ them at once against a target
Detection Systems
What have you learned?

• Analysis looks easy
  – Have some nice tools
  – Structured approach
  – I can do that:

• Not to stall optimism, BUT
  – Examples shown are labs/low usage networks
  – Analysis on busy production networks is much harder
  – Most of today's DDoS attacks are using more than one vector
  – Attackers adapt to countermeasures → i.e. change tactics & techniques

• Practice, practice, practice, ...

• And then you need to mitigate the attack → next session
Thank you

Any questions?

Next course: **DDoS Mitigation**

17\textsuperscript{th} of February 2021

www.geant.org
References:


• Shadowserver Foundation: https://www.shadowserver.org/
NetFlow Tools

- Pmacct: https://github.com/pmacct/pmacct/
- NFStream: https://www.nfstream.org/
- Softflowd: https://github.com/irino/softflowd
- SLiK Suite:
  - FlowViewer GUI for SILK tools:
- Nfdump: https://github.com/phaag/nfdump
- Nfsen-ng: https://github.com/mbolli/nfsen-ng
- GoFlow: https://github.com/cloudflare/goflow
  - https://github.com/cloudflare/flow-pipeline
- Dynamite NSM: https://dynamite.ai/dynamitensm/
  - https://github.com/DynamiteAI/dynamite-nsm
- Security Onion: https://securityonionsolutions.com/
RFCs


