Lessons Learned from Two DDOS Attacks on Resolver Hosts

Dejan Donin Brian Somers
Cisco Systems, Inc.
{ddonin,bsomers}@cisco.com

OARC 45 – 2025

Introduction

- Cisco's resolver fleet regularly experiences large-scale distributed denial of service (DDOS) attacks.
- Normally, attacks are mitigated by distributing traffic over installed resolver capacity.
- Operational issues are rare, but two notable incidents did occur.
- Thankfully, customer impact was very limited.
- This talk presents how these attacks were detected and remediated.

Detection and Initial Response

- Both incidents were detected via alarms from the resolver fleet.
- Alarms indicated delayed traffic servicing and delayed configuration updates.
- The underlying causes for resolver issues were different in each attack.

Incident 1: DNSCrypt Traffic Surge

- Random Label traffic attack started at one datacenter.
- Hostname was not found in cache - request was sent to Authoritative Nameserver.
- Due to the volume of traffic - the Authoritative Nameserver starts blackholing requests DC resolvers.
- Team attempted blacklisting attacking IPs - this was of limited value due to the large pool used.

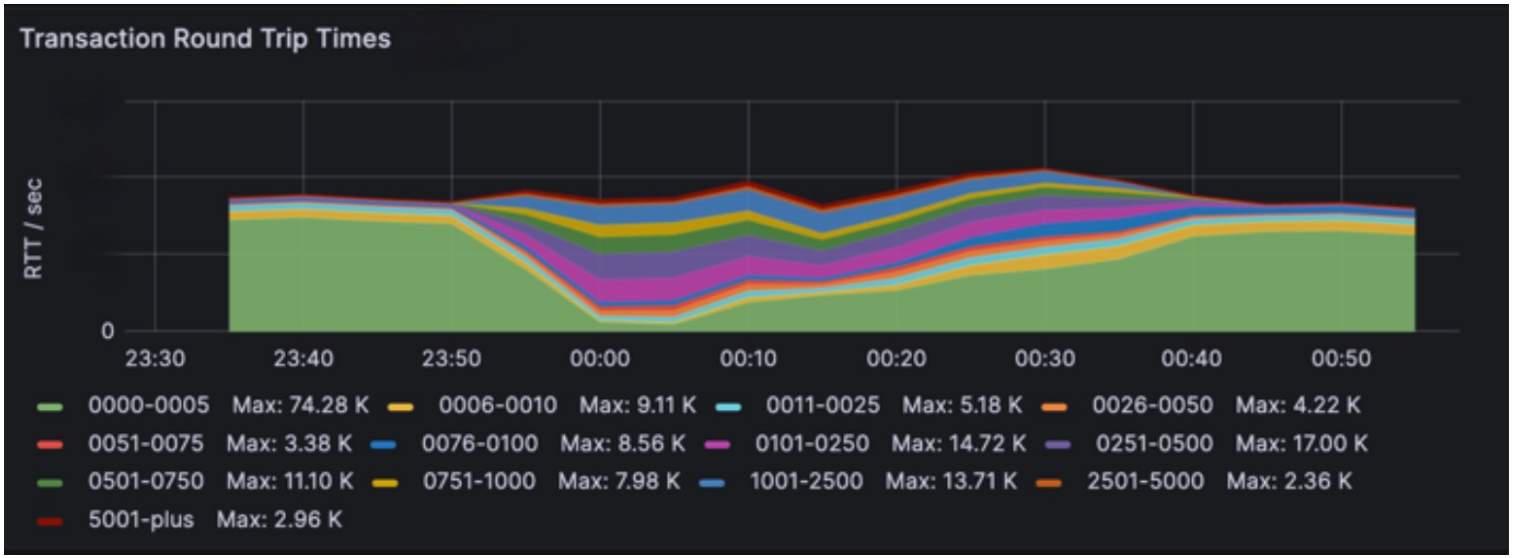

Figure 1: RTT measurement counts during the incident.

Incident 1: DNSCrypt Traffic Surge - Cont'd

- DNSCyrpt traffic showed a sudden increase during the incident.
- DNS Network Address Translation algorithm employed by the resolvers in case authority does not respond
- This is particularly common in the case of DDOS attack
- Resolvers in one data center (DC) referred more queries to a different DC's authority servers.
- This referral from one resolver to another one is implemented using DNSCrypt
- Root cause: caching of lock contention used for DNSCrypt transmission encryption in query referrals was inadvertently set incorrectly.
- This led to serialized keypair regeneration, creating bottlenecks under load.

Incident 2: Short-Lived DDOS

- The second DDOS attack was very short-lived and difficult to analyze.
- Processor thread states captured during the event revealed many threads spinning in a lock.
- The lock controls access to the list of in-transit upstream queries.
- This list is used to control chaining of queries to upstream authority
- Query's domain name hash (folded into 12 bits) determines which of 4096 locked lists is used.
- Multiple locks reduce contention, but only with good hash distribution.

Hashing and Attack Vector

- Implementation hashed the first qname label and target IP address.
- Reasoning: these are the most volatile transmission data parts.
- Result: random label attacks against <const>.<random>.<domain> always hashed to the same value and lock.

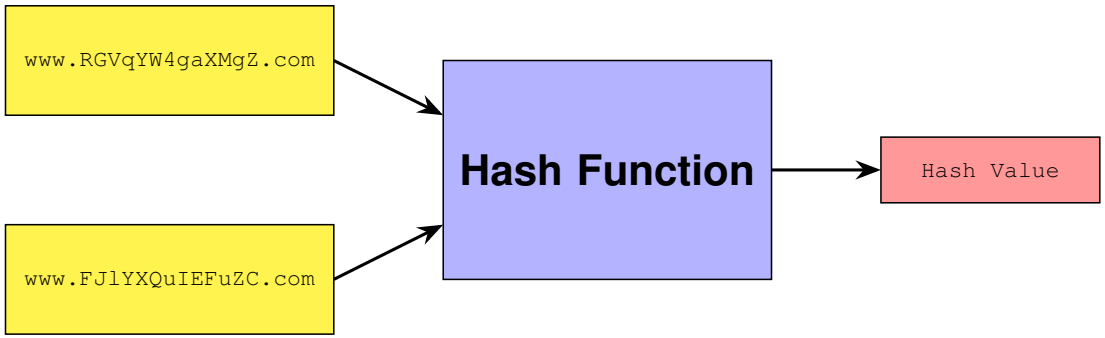

Figure 2: Two distinct domain labels are processed by a hash function, resulting in a single hash value used for lock selection.

Resolution and Lessons Learned

- Both incidents resolved via resolver software upgrades improving lock contention mechanisms.
- Each incident affected different resolver resources.
- Lock contention issues escaped detection despite extensive application and performance testing.
- Emphasizes the need for DDOS-type tests in the software release pipeline.

Conclusion

- DDOS attacks can expose subtle performance and contention issues in resolver infrastructure.
- Proactive monitoring and targeted testing are essential for robust DNS operations.
- Continuous improvement of detection and mitigation strategies is necessary.