Tales of the unexpected - handling unusual DNS client behaviour

UKNOF29 – Cathy Almond, ISC

Well-known DNS problems

- Reflection Attacks
  - Small queries with spoofed sender
  - Large replies hit spoofed victim
  - Mitigation focus on authoritative servers
  - Response Rate Limiting (RRL)
  - Inbound query rate limiting (firewalls/ filters) may also be deployed

Well-known DNS problems

- Malicious Domains and Sites
  - Mitigation focus on recursive servers
  - Block access or redirect clients
  - Local authoritative zones (labour-intensive to maintain)
  - Response Policy Zones (DNS RPZ)
  - Commercial zone ‘feeds’ available
  - Similar concept to anti-spam services

Newer DNS problems

- Popular domain outages
  - Decreasing in frequency due to e.g. :
    - Anycast
    - CDN techniques
  - Increase in recursive client contexts ('waiting queries')
  - More SERVFAIL responses/timeouts
  - Potential mitigation – SERVFAIL cache (will help if the queries are the same)

Newer DNS problems

- Unusual patterns of client queries – probing and keepalive
  - TuneIn Internet Radio - <random10x.com> queries
  - Chrome random DNS requests
  - Increase in NXDOMAIN responses (cached…)
  - Mitigation – reduce TTL of negative cache *(in BIND max-ncache-ttl)*

Newer DNS problems

- ‘Collateral Damage’ Client DDoS traffic

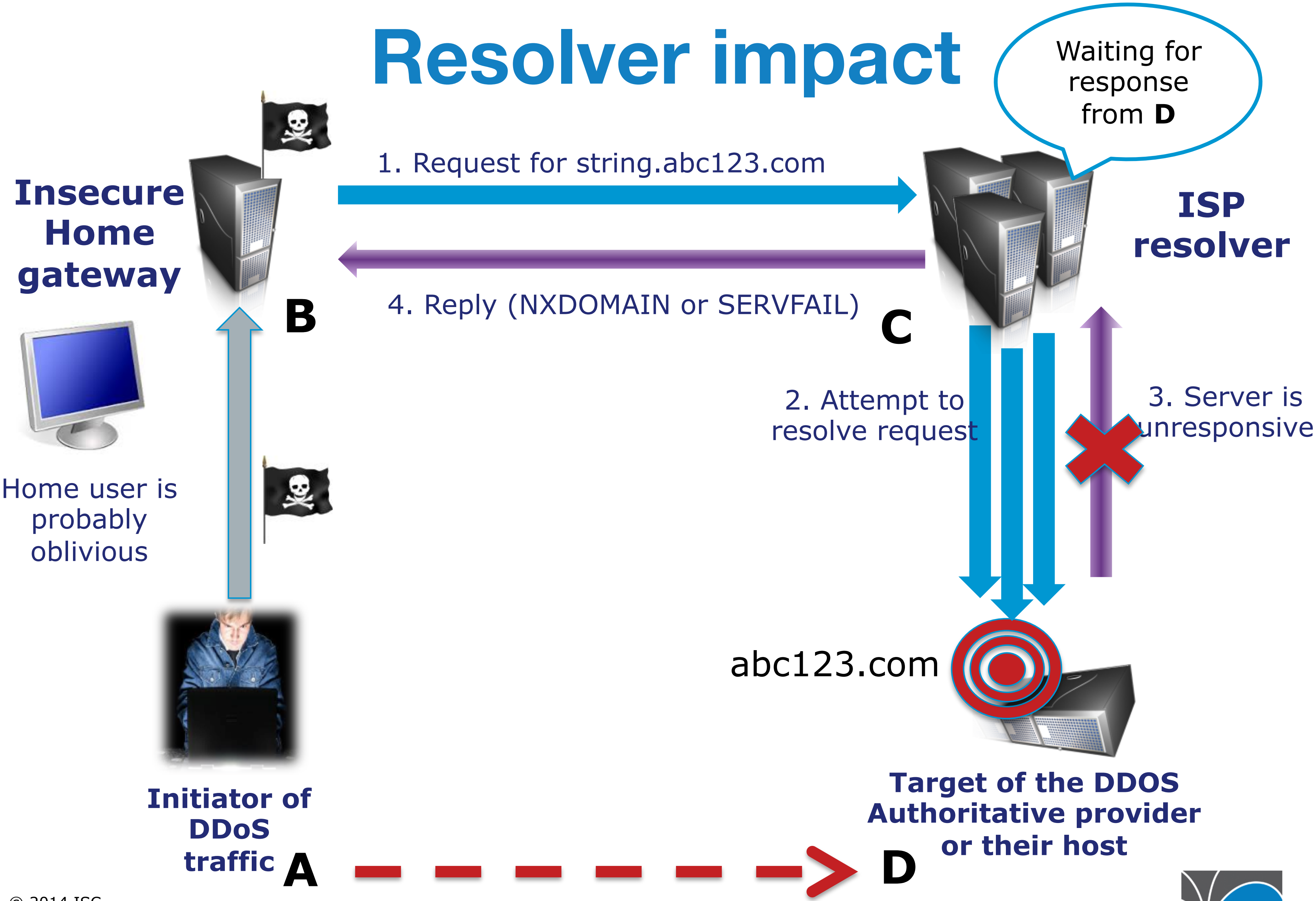
  <randomstring>.www.abc123.com

  <anotherstring>.www.abc123.com

The queries are unique and originate from a large range of different client addresses. Typically, the servers for abc123.com do not respond at all, or only sporadically to the recursive server handling the client query.

A flurry of queries will run for a day or two, then stop. The domains are genuine, and the majority appear to be for online commercial sites, often hosted in China.

Resolver impact



Waiting for response from **D**

1. Request for string.abc123.com

4. Reply (NXDOMAIN or SERVFAIL)

Insecure Home gateway

B

C

ISP resolver

Home user is probably oblivious

2. Attempt to resolve request

3. Server is unresponsive

abc123.com

Initiator of DDoS traffic

A

D

Target of the DDOS Authoritative provider or their host

Problem statement

- Authoritative servers under attack are non-responsive and tie up resolver resources waiting for replies
- So far, the impact on recursive server resources appears to be accidental - primarily due to open resolvers.
- This is a wake-up call that we need to better manage recursive resources

Mitigation Approaches

- Traffic patterns impacting all recursive servers (not just BIND)
- Mitigations suggested/introduced:
  - Network infrastructure/environment
  - Some generic to all DNS servers
  - Some specific to BIND (currently experimental) but could be adopted by other DNS server software providers.

Mitigation Approaches - 1

- Eliminate open resolvers
  - Is your recursive server an open resolver?
  - Open client CPE devices
  - Small business users forwarding local open caches to your servers
- Compromised/infected clients
  - ‘hearsay’ evidence that these exist now
  - But it’s only a matter of time…

Mitigation Approaches – 2

- Locally-created authoritative answers
  - Detect ‘bad’ domain names
  - Make recursive server temporarily authoritative for the domain being used
  - *Prevents valid queries (which wouldn’t succeed anyway)*
  - *Problem of false-positives – might need white-lists if using scripted detection*
  - *Need to undo the mitigation afterwards*

Mitigation Approaches – 3

- Response Policy Zones (DNS-RPZ)
  - Detect ‘bad’ domain names
  - Update RPZ zone to blacklist domains
  - *Prevents valid queries (which wouldn’t succeed anyway)*
  - *Problem of false-positives – might need white-lists if using scripted detection*
  - *Need to undo the mitigation afterwards*

Experimental Approaches – 1

- Hold-down Timer *(since writing, deprecated and replaced with fetches-per-server)*
  - One timer each per server per zone
  - Count how many consecutive times a server fails to respond (***holddown-threshold***)
  - When threshold reached, don't send queries to that server for ***holddown-timer*** seconds (doesn't abort any currently waiting queries)
  - Quick check – if next 'response' from server is a timeout, then hold-down immediately
  - *Ineffective with intermittent outages.*

Experimental Approaches – 3

- Rate-limiting ***fetches-per-zone***
  - Similar to clients-per-query
  - Works with unique clients
  - Default 0 (no limit enforced)
  - Tune larger/smaller depending on normal QPS to avoid impact on popular domains
  - *Could be less effective against non-responding server for many zones*

Experimental Approaches – 4

- Recursive Client Contexts soft quota
  - Old default: recursive-clients 1000; hard limit, no soft limit, queries just dropped.
  - Over 1000, soft-limit = hard limit – 100
  - New behaviour when recursive-clients <= 1000 – soft limit based on number of worker threads
  - Soft drop accepts new client and SERVFAILs oldest waiting client
  - *Less effective with high QPS*

Experimental Approaches – 5

- Random Drop Policy
  - Instead of always dropping the oldest waiting client, pick one at random
  - Configure % newest, random, oldest
  - client-drop-policy x y z;
  - Default 0 50 50
- Why?
  - Recursive client backlog build-up is similar to TCP SYN flood attack

More ideas…

- Single-socket for iterative queries to a ‘new’ server until it has proven to be responsive.
  - *One in, one out… until we know that the server is well-behaved.*
  - *Not sure how we implement a new restriction when a server ‘goes bad’?*
  - *Should help preserve internal resources*
  - *Unlikely to save recursive client backlog*

More ideas…

- Whitelists
  - *For fetches-per-zone and fetches-per-server*
- Per-server/zone settings
  - *Configurable override parameters for fetch limits on a per zone or per server basis*
- SERVFAIL cache (for client retries)
- Improved reporting & statistics

Questions and musings…

- Other ideas?
- Tuning is an art not a science – when is this is ‘good enough’ to do the job that is needed…
- How to make sure that we’re not introducing new DoS vectors?
- What about TCP?

THANK YOU!

bind-suggest@isc.org, cathya@isc.org