ISC

Internet Systems Consortium

BIND Logging

Content out of Chaos

Alan Clegg

October 30, 2019

BIND

Why Do We Log?

- Consider the times that you look at log files
  - Something is new:
    - General overview of functionality
  - Services are broken or the network is on fire:
    - Specific logs related to a specific topic
    - Higher detail than normal

Why Do We Log?

- During normal operation, logging is mostly disregarded
  - Minimal disk usage
  - Minimal processing
- During network-on-fire events, logging is important
  - Lots of output surrounding the bits that are causing problems
  - Minimal processing – without changing configuration

Logging Methodology

- BIND logs **Categories** into **Channels**
  - Categories are pre-defined
    - Collection of messages around a common theme
  - Channels are (for the most part) administrator defined
    - Definitions provide location, content, detail level and size of output
    - Detail level may be dynamic – very useful!

Logging Categories

```
client
cname
config
database
```
`default`
```
delegation-only
dispatch
dnssec
dnstap
edns-disabled
```
`general`
```
lame-servers
network
notify
nsid
queries
query-errors
rate-limit
resolver
rpz
security
serve-stale
spill
trust-anchor-telemetry
unmatched
update
update-security
xfer-in
xfer-out
zoneload
```

Logging Channels

- Predefined channels are:

```
default_syslog
default_debug
default_stderr
null
default_logfile (only created if BIND is started with -L)
```

- Others will be created by the administrator
- By default and before parsing **`named.conf`** logging goes to **`default_syslog`**

Logging Stanza Syntax

```
logging {
     category string { string; ... };
     channel string {
          buffered boolean;
          file quoted_string [ versions ( unlimited | integer ) ]
               [ size size ] [ suffix ( increment | timestamp ) ];
          null;
          print-category boolean;
          print-severity boolean;
          print-time ( iso8601 | iso8601-utc | local | boolean );
          severity log_severity;
          stderr;
          syslog [ syslog_facility ];
     };
};
```

Logging Severity

- `log_severity` is a set of levels
    - Logging at a given level includes all of the levels below

Logging Samples

- Use default logging, but in addition, send **`dnssec`** logging to a file called **`dnssec.log`**
  - Keep 5 copies (+ the active one) of 10MB each
  - Record the time and severity

```
logging {
     channel dnssec_log {
          file "/var/log/bind/dnssec.log"
               versions 5
               size 10M;
          severity debug 10;
          print-time yes;
          print-severity yes;
     };
     category dnssec { dnssec_log; };
};
```

Logging Samples

- Log all queries to a file called **`query.log`**
  - Keep 3 copies (+ the active one) of 10MB each
- This file will remain empty until we explicitly turn it on:
  - **`$ rndc querylog on`** or global option **`querylog yes;`** in **`named.conf`**

```
logging {
     channel query_log {
          file "/var/log/bind/query.log" versions 3 size 10M suffix timestamp;
          print-time yes;
     };
     category queries { query_log; };
};
```

Logging Samples

- Log **queries** to two channels, log three categories to a single channel dynamically

```
logging {
        channel query_log {
                file "/tmp/query.log" versions 5;
        };
        channel debug_log {
                file "/tmp/debug.log" size 100k;
                print-time yes;
                print-severity yes;
                print-category yes;
                severity dynamic;
        };
        category queries { query_log; debug_log; };
        category dnssec { debug_log; };
        category client { debug_log; };
};
```

Possible Errors here:
You probably want to specify both **versions** and **size**.

Deciphering the output

- Good luck!
  - Just kidding… to an extent
- Most logging is for the ISC engineering team - not the mere mortal

```
29-Oct-2019 22:16:34.068 client: debug 3: client @0x712a4ab0 192.168.77.130#56722 (d.docs.live.net): send
29-Oct-2019 22:16:34.068 client: debug 3: client @0x712a4ab0 192.168.77.130#56722 (d.docs.live.net): sendto
29-Oct-2019 22:16:34.068 client: debug 3: client @0x712a4ab0 192.168.77.130#56722 (d.docs.live.net): senddone
29-Oct-2019 22:16:34.068 client: debug 3: client @0x712a4ab0 192.168.77.130#56722 (d.docs.live.net): next
29-Oct-2019 22:16:34.068 client: debug 10: client @0x712a4ab0 192.168.77.130#56722 (d.docs.live.net):
ns_client_detach: ref = 0
29-Oct-2019 22:16:34.068 client: debug 3: client @0x712a4ab0 192.168.77.130#56722 (d.docs.live.net): endrequest
29-Oct-2019 22:16:34.518 client: debug 90: client @0x71251370 192.168.77.1#50360: received DSCP 0
29-Oct-2019 22:16:34.518 client: debug 3: client @0x71251370 192.168.77.1#50360: UDP request
29-Oct-2019 22:16:34.519 client: debug 5: client @0x71251370 192.168.77.1#50360: using view '_default'
29-Oct-2019 22:16:34.519 client: debug 3: client @0x71251370 192.168.77.1#50360: query
29-Oct-2019 22:16:34.519 queries: info: client @0x71251370 192.168.77.1#50360 (ccn.asdf.com): query:
ccn.asdf.com IN A +E(0)K (192.168.77.1)
```

- One exception: **query logging**

Deciphering Query Log Output

```
client @0x7129cc38 192.168.77.1#39584 (_http._tcp.mirror.os6.org): query: _http._tcp.mirror.os6.org IN SRV +
(192.168.77.1)
client @0x6ff4c250 192.168.77.1#57515 (mirror.os6.org): query: mirror.os6.org IN AAAA + (192.168.77.1)
client @0x6ff287f8 192.168.77.1#57515 (mirror.os6.org): query: mirror.os6.org IN A + (192.168.77.1)
client @0x6fd12ce8 192.168.77.131#21399 (imap.gmail.com): query: imap.gmail.com IN A + (192.168.77.1)
client @0x712a4ab0 192.168.77.1#44466 (alan.clegg.com): query: alan.clegg.com IN A +E(0)DK (192.168.77.1)
```

- The word `client`
- A **`@0x`** followed by the client object identifier (nothing to do with the client address)
- The IP address and port number from which the query originated (the client address)
- The query (in parenthesis), a colon and the word "query" followed by a colon

(continued)

Deciphering the output

```
client @0x7129cc38 192.168.77.1#39584 (_http._tcp.mirror.os6.org): query: _http._tcp.mirror.os6.org IN SRV +
(192.168.77.1)
client @0x6ff4c250 192.168.77.1#57515 (mirror.os6.org): query: mirror.os6.org IN AAAA + (192.168.77.1)
client @0x6ff287f8 192.168.77.1#57515 (mirror.os6.org): query: mirror.os6.org IN A + (192.168.77.1)
client @0x6fd12ce8 192.168.77.131#21399 (imap.gmail.com): query: imap.gmail.com IN A + (192.168.77.1)
client @0x712a4ab0 192.168.77.1#44466 (alan.clegg.com): query: alan.clegg.com IN A +E(0)DK (192.168.77.1)
```

- The query (2nd time, but without parenthesis) followed by the class and type of the query
- A set of flags:
  - If RD flag was set (**+** if set, **-** if not set), if signed (**S**), if EDNS was in use with the EDNS version number (**E(#)**), if TCP was used (**T**), if DNSSEC Ok was set (**D**), if CD was set (**C**), if a valid DNS Server cookie was received (**V**), and whether a DNS cookie option without a valid Server cookie was present (**K**)

(continued)

BIND

Deciphering the output

```
client @0x7129cc38 192.168.77.1#39584 (_http._tcp.mirror.os6.org): query: _http._tcp.mirror.os6.org IN SRV +
(192.168.77.1)
client @0x6ff4c250 192.168.77.1#57515 (mirror.os6.org): query: mirror.os6.org IN AAAA + (192.168.77.1)
client @0x6ff287f8 192.168.77.1#57515 (mirror.os6.org): query: mirror.os6.org IN A + (192.168.77.1)
client @0x6fd12ce8 192.168.77.131#21399 (imap.gmail.com): query: imap.gmail.com IN A + (192.168.77.1)
client @0x712a4ab0 192.168.77.1#44466 (alan.clegg.com): query: alan.clegg.com IN A +E(0)DK (192.168.77.1)
```

- The address to which the query is sent (in parenthesis)
- If any CLIENT-SUBNET option was present in the client query, it is included in square brackets in the format [ECS address/source/scope]

What about `dnstap`?

- **`dnstap`** is a flexible, structured binary log format for DNS software. It uses protocol buffers to encode events that occur inside DNS software in an implementation-neutral format.

There will be a future presentation on **`dnstap`**

- If you are in a hurry: https://kb.isc.org/docs/aa-01342

Logging Warnings

- Logging respects **`directory`** option
- Logs reside within **`chroot`** if used
- High debug levels will cause headaches:
  - Huge output or rapidly moving files
  - Messages formatted differently (breaking parsers)
- **BIND may become slow in query processing due to being busy logging**

Logging Warnings

```
29-Oct-2019 20:34:50.510 database: debug 5: expiring v6 for name 0x703c2300
29-Oct-2019 20:34:50.511 database: debug 5: dns_adb_createfind: found AAAA for name
circulum.clegg.com (0x703c2300)
29-Oct-2019 20:34:50.667 resolver: debug 10: received packet from 2400:cb00:2049:1::a29f:1981#53
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:  54619
;; flags: qr aa; QUESTION: 1, ANSWER: 2, AUTHORITY: 9, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 512
;; QUESTION SECTION:
;alan.clegg.com.                        IN      A

;; ANSWER SECTION:
;alan.clegg.com.                300     IN      A       45.33.100.174
;alan.clegg.com.                300     IN      RRSIG   A 10 3 300 (
;                                               20191108171106 20191009163235 40661 clegg.com.
;[;…]
;                                               cGMsHqlqH8L5NoiqbadX/wLIwyiA
;                                               Psk= )

;; AUTHORITY SECTION:
;clegg.com.             86400   IN      NS      ns7.dnsmadeeasy.com.
[…]
```

Additional Resources

- ISC Knowledge Base:
    - BIND Logging - some basic recommendations
        - https://kb.isc.org/docs/aa-01526
- Zytrax:
    - DNS BIND9 logging Clause
        - http://www.zytrax.com/books/dns/ch7/logging.html

Questions?

Comments?

https://www.isc.org

Kea
Modern DHCP