The Present and Future of BIND

Evan Hunt
Witold Kręcicki
Matthijs Mekking
8 February 2020

https://www.isc.org
First Things First

- BIND 9.16.0 based on 9.15 dev branch
- Due out December 2019
  January 2020
  February 19, 2020

- Major features:
  - New network system
  - DNSSEC Key and Signing Policy
  - DLV obsolete, validator code simplified
Question:

Howdy, @bind9 users, can I ask a favor?

For a talk I'm preparing, I'd really like to gather some non-developer perspectives on this question:

What are BIND's particular strengths and weaknesses as a DNS implementation?

Thanks.

1:55 AM · Jan 21, 2020 · TweetDeck

Still interested in answers; each@isc.org or @nuthaven.
The Good

- RFC conformance
- Versatility
- Familiarity
- Ubiquity
- Tools
- Documentation
- Professional support
The Less Good

- Too many features
- Not enough features
- Configuration
  - Too many options
  - Requires editing files
- Development slow due to complex code base
The Less Good

- Too many features
- Not enough features
- Configuration
  - Too many options
  - Requires editing files
- Development slow due to complex code base
The Less Good

- Too many features
- Not enough features
- Configuration
  - Too many options
  - Requires editing files
- Development slow due to complex code base
- Performance?
The Less Good

- Too many features
- Not enough features
- Configuration
  - Too many options
  - Requires editing files
- Development slow due to complex code base
- Performance?
Simplifying DNSSEC configuration

- DLV obsolete
- More consistent trust anchor configuration
- Key and Signing Policies in named ("dnssec-policy")
DLV obsolete

- isc.dlv.org replaced with an empty zone September 2017.
- All relevant code removed from validator.
Trust anchor configuration

- “trusted-keys” and “managed-keys” statements are deprecated, replaced by “trust-anchors”.
- “trust-anchors” can set both static and initializing keys, in either DNSKEY or DS format.
- Trust anchors are all now stored internally in DS format.
- Validator shrunk by 700 LOC, McCabe complexity reduced 35%.
Key and Signing Policies (KASP)

- New “dnssec-policy” statement enables configuration of key size and rollover policies for a zone.
- Key rolls fully automated within named.
- … including KSKs (soon).
Signing options (before KASP):

zone "example.com" {
    auto-dnssec maintain;
    inline-signing yes;
    dnskey-sig-validity 14;
    dnssec-dnskey-kskonly yes;
    dnssec-loadkeys-interval 3600s;
    dnssec-secure-to-insecure no;
    dnssec-update-mode maintain;
    max-zone-ttl 24h;
    sig-signing-type 65445;
    sig-validity-interval 14 3;
    update-check-ksk yes;
};
Signing options (before KASP, more realistic):

```
zone "example.com" {
    auto-dnssec maintain;
    inline-signing yes;
};
```

… with keys generated and maintained by external tools.
Signing options (before KASP):

• Keys must be generated using dnssec-keygen.
• Key rollovers and retirements must be scheduled using dnssec-settime.
• Both of these can be automated according to a key/signing policy by using dnssec-keymgr in a cron job, however:
  • No automatic checking of key state transitions.
  • Requires attention to timing.
  • No CDS/CDNSKEY support.
  • KSK rollovers manual.
Signing options (after KASP):

```dns
zone "example.com" {
    dnssec-policy default;
};
```
Signing options (after KASP):

dnssec-policy example {
  keys {
    zsk lifetime 365d algorithm ecdsa256;
    ksk lifetime unlimited
    algorithm ecdsa256;
  };
};

zone "example.com" {
  dnssec-policy example;
};
Signing options (after KASP):

- Keys are generated by named as needed.
- “auto-dnssec” and “inline-signing” are implicit; most other signing options merged into dnssec-policy.
- State transitions are monitored, illegal changes prevented - much less attention to timing needed.
- Algorithm rolls can be initiated by editing named.conf - and will occur automatically.
Key state machine

- **Rumoured**: Not fully propagated; too new to be cached everywhere.
- **Omnipresent**: Propagated to all secondaries, visible to all validators.
- **Unretentive**: Phased out, expiring from caches.
- **Hidden**: Not yet published or fully unpublished.
State machine logic:

**Chain of trust:**
- At least one DS is published.
- At least one DNSKEY matching at least one DS is published.
- All records are signed by at least one key visible to all validators.

\[
\text{rule1}(x): \\
\exists y \in K (D_{y}^{++})
\]

\[
\text{rule2}(x): \\
\exists y \in X (D_{y}^{+}K_{y}^{+}R_{y}^{+}) \quad \checkmark \\
\exists y, z \in X (D_{y}^{+}K_{y}^{+}R_{y}^{+}D_{z}^{+}K_{z}^{+}R_{z}^{+} \wedge y > z) \quad \checkmark \\
\exists y, z \in X (D_{y}^{+}K_{y}^{++}R_{y}^{+}D_{z}^{+}K_{z}^{++}R_{z}^{+} \wedge y > z) \quad \checkmark \\
\forall y \in X (D_{y}^{-} \vee \exists z \in X (K_{z}^{+}R_{z}^{+} (D_{y} = D_{z})))
\]

\[
\text{rule3}(x): \\
\exists y \in X (K_{y}^{+}S_{y}^{+}) \quad \checkmark \\
\exists y, z \in X (K_{y}^{+}S_{y}^{+}K_{z}^{+}S_{z}^{+} \wedge y > z) \quad \checkmark \\
\exists y, z \in X (K_{y}^{+}S_{y}^{++}K_{z}^{++}S_{z}^{++} \wedge y > z) \quad \checkmark \\
\forall y \in X (K_{y}^{-} \vee \exists z \in X (S_{z}^{+} (K_{y} = K_{z})))
\]
State files

- Keys generated by named include a third file, K*.state (in addition to K*.key and K*.private).
- State files are publicly readable.
- Key metadata residing in private file is now duplicated in state file (though still kept in private file as well for legacy reasons).
- Additional metadata indicates key state and transitions.
State file (KSK)

; This is the state of key 33330, for example.com.
Algorithm: 13
Length: 256
Lifetime: 16070400
Predecessor: 17530
KSK: yes
ZSK: yes
Generated: 20200207005010 (Thu Feb 6 16:50:10 2020)
Published: 20200111165010 (Sat Jan 11 08:50:10 2020)
Active: 20200111195010 (Sat Jan 11 11:50:10 2020)
Retired: 20200715195010 (Wed Jul 15 12:50:10 2020)
DNSKEYChange: 20200111195010 (Sat Jan 11 11:50:10 2020)
ZRRSIGChange: 20200111195010 (Sat Jan 11 11:50:10 2020)
KRRSIGChange: 20200111195010 (Sat Jan 11 11:50:10 2020)
DSChange: 202001111235010 (Sat Jan 11 15:50:10 2020)
DNSKEYState: omnipresent
ZRRSIGState: omnipresent
KRRSIGState: omnipresent
DSState: omnipresent
GoalState: omnipresent
State file (ZSK)

; This is the state of key 44585, for example.com.
Algorithm: 13
Length: 256
Lifetime: 31536000
Successor: 61247
KSK: no
ZSK: yes
Generated: 201908111005009 (Sat Aug 10 17:50:09 2019)
Published: 201908111005009 (Sat Aug 10 17:50:09 2019)
Active: 201908111005009 (Sat Aug 10 17:50:09 2019)
Retired: 20200207005009 (Thu Feb  6 16:50:09 2020)
DNSKEYChange: 201908111005009 (Sat Aug 10 17:50:09 2019)
ZRRSIGChange: 201908111005009 (Sat Aug 10 17:50:09 2019)
DNSKEYState: omnipresent
ZRRSIGState: omnipresent
GoalState: hidden
Not yet working:

- NSEC3 configuration.
- Querying parent to monitor DS status prior to KSK rollover.
- Signaling mechanism to inform named that DS has been submitted to parent.
- Key state monitoring via rndc.
- Purging retired keys.
- CDS/CDNSKEY.
The Less Good

- Too many features
- Not enough features
- Configuration
  - Too many options
  - Requires editing files
- Development slow due to complex code base
- Performance?
More legacy, more problems

- Original BIND architecture was designed to be both single- and multi-threaded.
- Ran on absolutely everything.
- Provides many of its own OS services, such as memory management and socket/event libraries.
- Not optimized for modern hardware.
- New capabilities added, old capabilities rarely removed.
BIND architecture

...as seen by non-developers.

Image credit:
Vicky Risk
Network manager

- New asynchronous socket API for BIND.
- Based on libuv (Unicorn Velociraptor), but designed for flexibility.
- Much more efficient design.
- Modular and extensible.
Network manager modules

Transport-independent query processing

UDP

DNS

TCP
Future network manager modules

Transport-independent query processing

- UDP
- DNS
- TLS
- HTTPS
- TCP
The Less Good

- Too many features
- Not enough features
- Configuration
  - Too many options
  - Requires editing files
- Development slow due to complex code base
- Performance?
Performance in 9.16.0

Authoritative performance, 1k zones scenario

RPS

v9.11  v9.14  v9.16.0
Performance in 9.16.0

![Graph showing recursive performance with hot cache for versions v9.11, v9.14, and v9.16.0.](Image)
Performance in 9.16.1 (under development)
Performance in 9.16.1 (under development)
The network manager is currently only used for processing client requests.

The original BIND isc_socket API is still in use for:

- Upstream queries
- RNDC
- Statistics
- ...everything else
Coming in 9.17/9.18 (2021)

- Finish conversion to network manager
- DNS over TLS
- DNS over HTTPS
- More dnssec-policy features
- ...

All content © 2020 Internet Systems Consortium, Inc.
Thank you! Questions?

• Main website: https://www.isc.org
• Software downloads: https://www.isc.org/download or https://downloads.isc.org
• Presentations: https://www.isc.org/presentations
• Main GitLab: https://gitlab.isc.org