Kea 2.0
A modern DHCP

Tomek Mrugalski,
Director of DNS Engineering, ISC
2022-Apr-12, SEE10
Tomek Mrugalski

- MSc (2003), PhD (2010), both about DHCPv6
- Started Dibbler in 2003 (complete DHCPv6 solution)
- 7 years at Intel
- IETF (since 2009)
  - DHC WG co-chair at IETF (till 2020)
  - 13 RFCs published
  - DHCPv6bis (RFC8415) as primary author
- ISC (since 2011)
  - First engineer working on Kea
  - Currently Director of DHCP engineering
ISC DHCP Legacy

- Provided in many major operating systems
- Released in 1995
- widely used, but not aging well
- ISC DHCP “development” is in maintenance mode only
- Kea is a replacement for the ISC DHCP server
- 4.4.3 released in Mar 2022. **Last release for client and relay.**
- Upcoming 4.5.0 will be server only.
- If you are running this in your network today - consider it technical debt
Kea Differences from ISC DHCP

- Extensive REST Management API
- Separate ‘backends’ leveraging popular open source DBs
  - Leases
  - Reservations
  - Server configurations
- Extensible with optional hooks libraries, including many from ISC
- Open source (MPL2), with commercial add-ons
- Available as source, or as ISC packages for popular OSes
- Both stable and development branches available
Time to Migrate to Kea

- Run the migration assistant
- Fix up 20 - 30% this doesn’t cover
- Migrate leases if desired
  https://www.isc.org/presentations/
- NANOG’76 talk
  https://pc.nanog.org/static/published/meetings//NANOG76/daily/day_2.html#talk_1998
Kea’s REST API

- Allows on-line reconfiguration of DHCPv4, DHCPv6 and DDNS servers without restarting
- Kea configuration AND the REST api, use JSON syntax (comments allowed)
- API commands are fully documented in


184 commands available and growing
API Basics

1. Send **list-commands** command:

```
# kea-shell --host ::1 --port 8080 --service dhcp6 list-commands
^D
```
2. Get list of currently supported commands in return:

```
{
    "command": "list-commands",
    "service": [ "dhcp6" ]
}
```

```json
{
    "arguments": [
        "build-report",
        "config-get",
        "config-set",
        "config-test",
        "remote-global-parameter4-del",
        "remote-global-parameter4-get",
        "remote-global-parameter4-get-all",
        ...
        "remote-subnet6-list",
        "server-tag-get",
        "shutdown",
        "statistic-{get,remove,reset}"
        "statistic-{get,remove,reset}-all",
        "version-get"
    ],
    "result": 0
}
```
Why use database ‘backends’?

- SQL data can be modified any time
- All changes applied instantly (no restart)
- Adapt your provisioning systems to write directly to the database or
- Use the API (some of these require premium hooks libraries)
- More complicated deployment, more things to install and manage (the db)
Available Backends

- **Leases (addresses, prefixes)**
- **Host reservations (per host details)**
- **Options**
- **Pools**
- **Subnets**
- **Shared networks**
- **Option definitions**
- **Global parameters**

- **DHCPv4, DHCPv6 server**

- **MySQL**
  - Lease backend
- **Postgres (2.1)**
  - Hosts backend
  - Configuration backend

<table>
<thead>
<tr>
<th>Rarely</th>
<th>Often</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changing</td>
<td></td>
</tr>
</tbody>
</table>
Recent changes in Kea 2.0

1. Significant performance boost with multi-threading
2. Addition of TLS security for connections
   - Kea - db backends
   - Kea - stork
   - Kea - api clients
3. New features
   - Cache threshold
   - Script hook
4. Stork graphical dashboard
Multi-threading (Kea 1.8)

ctrl-agent1

dhcp1

dhcp2

ctrl-agent2

HTTP

JSON/Unix

DHCP

dhcp1

dhcp2

Multi-threaded

Sequential
High Availability with Multi-threading (Kea 2.0)
Multi-threading performance boost (Kea 2.0)

For A LOT more details, see https://reports.kea.isc.org
API (Kea 2.0)
- TLS 1.3
- 3 modes:
  - Disabled
  - Encryption (client verifies server)
  - Mutual (both sides verifies each other)

Secure database connection
- MySQL (Kea 2.1)
- PostgreSQL (Kea 2.0)

More security enhancements coming!

"Control-agent": {
  "http-host": "127.0.0.1",
  "http-port": 8000,

  // TLS trust anchor (Certificate Authority).
  "trust-anchor": "my-ca",

  // TLS server certificate file name.
  "cert-file": "my-cert",

  // TLS server private key file name.
  "key-file": "my-key",

  "cert-required": true
}
Cache Threshold (2.0)

- Problem: Buggy clients renewing early
- Each renewal:
  - Host reservation lookup
  - Lease lookup
  - Logging*
  - HA: partner update*
  - DNS Update*
- Solution: cache replies
- IPv4 and IPv6

```
"subnet6": [
    {
        "subnet": "2001:db8::/64",
        "pools": [ { "pool": "2001:db8::/64" } ],
        "renew-timer": :1000,
        "valid-lifetime": 2000,
        "cache-threshold": .25,
        "cache-max-age": 600,
        ...
    }
],
```
But I want to … <your secret voodoo here>
Stork

Kea (and BIND9) Dashboard/GUI/IPAM
Stork Dashboard

One Stork server + one or more agents
- Collects data from Kea/BIND9 services
- Aggregates data
- Web interface
- Export to Prometheus/Grafana

- Server details: version, build, installed hooks, cpu, memory
- Fault monitoring: subnet utilization, HA failures, log viewer
- Statistics: DORAs, QPS, NAKs
- Config viewer: file locations, database backends, etc

- Monthly* releases
- Dashboard for now, configuration management coming up in 1.3
## Stork Dashboard - Subnet Utilization

<table>
<thead>
<tr>
<th>Subnet ID</th>
<th>Subnet</th>
<th>Addresses</th>
<th>Pools</th>
<th>Shared Network</th>
<th>App Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>192.0.5.0/24</td>
<td>50</td>
<td>42</td>
<td>84%</td>
<td>192.0.5.1-192.0.5.50</td>
</tr>
<tr>
<td>2</td>
<td>192.0.6.0/24</td>
<td>110</td>
<td>0</td>
<td>0%</td>
<td>192.0.6.1-192.0.6.40, 192.0.6.61-192.0.6.90, 192.0.6.111-192.0.6.150</td>
</tr>
<tr>
<td>3</td>
<td>192.0.7.0/24</td>
<td>50</td>
<td>50</td>
<td>100%</td>
<td>192.0.7.1-192.0.7.50</td>
</tr>
<tr>
<td>4</td>
<td>192.0.8.0/24</td>
<td>50</td>
<td>0</td>
<td>0%</td>
<td>192.0.8.1-192.0.8.50</td>
</tr>
<tr>
<td>5</td>
<td>192.0.9.0/24</td>
<td>50</td>
<td>0</td>
<td>0%</td>
<td>192.0.9.1-192.0.9.50</td>
</tr>
<tr>
<td>6</td>
<td>192.1.15.0/24</td>
<td>50</td>
<td>20</td>
<td>40%</td>
<td>192.1.15.1-192.1.15.50</td>
</tr>
<tr>
<td>7</td>
<td>192.1.16.0/24</td>
<td>150</td>
<td>39</td>
<td>26%</td>
<td>192.1.16.1-192.1.16.150</td>
</tr>
<tr>
<td>8</td>
<td>192.1.7.0/24</td>
<td>245</td>
<td>0</td>
<td>0%</td>
<td>192.1.7.1-192.1.7.20, 192.1.7.21-192.1.7.40, 192.1.7.41-192.1.7.60, 192.1.7.61-192.1.7.100</td>
</tr>
<tr>
<td>9</td>
<td>192.0.2.0/24</td>
<td>200</td>
<td>1</td>
<td>0.5%</td>
<td>192.0.2.1-192.0.2.50, 192.0.2.51-192.0.2.100, 192.0.2.101-192.0.2.150, 192.0.2.151-192.0.2.200</td>
</tr>
<tr>
<td>10</td>
<td>1.0.0.0/16</td>
<td>65,531</td>
<td>0</td>
<td>0%</td>
<td>1.0.0.4-1.0.0.255,254</td>
</tr>
</tbody>
</table>
Stork GUI - Monitoring HA Status

- Groups HA pairs
- Displays roles
  - Primary/standby
  - Load balancing
- Heartbeat status
- HA States
- Scopes served
- Last outage

### High Availability

<table>
<thead>
<tr>
<th>Local server</th>
<th>Remote server Kea@127.0.0.1△</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Status time:</strong></td>
<td>2022-02-02 16:16:09</td>
</tr>
<tr>
<td><strong>Status checked:</strong></td>
<td>4 minutes ago</td>
</tr>
<tr>
<td><strong>Role:</strong></td>
<td>standby</td>
</tr>
<tr>
<td><strong>Control status:</strong></td>
<td>offline</td>
</tr>
<tr>
<td><strong>Heartbeat status:</strong></td>
<td>failed</td>
</tr>
<tr>
<td><strong>State:</strong></td>
<td>unavailable</td>
</tr>
<tr>
<td><strong>Scopes served:</strong></td>
<td>none</td>
</tr>
<tr>
<td><strong>Last in partner-down:</strong></td>
<td>n/a</td>
</tr>
<tr>
<td><strong>Unacked clients:</strong></td>
<td>n/a</td>
</tr>
<tr>
<td><strong>Connecting clients:</strong></td>
<td>n/a</td>
</tr>
<tr>
<td><strong>Analyzed packets:</strong></td>
<td>n/a</td>
</tr>
</tbody>
</table>

### Notes

The remote server responds to the entire DHCP traffic.
Prometheus / Grafana export
Participation is Welcome!

https://gitlab.isc.org/isc-projects/kea/

https://gitlab.isc.org/isc-projects/stork/
Questions?

isc.org/kea
gitlab.isc.org/isc-projects/kea
gitlab.isc.org/isc-projects/stork