



Kea Webinar

Monitoring, Logging and Stork

Carsten Strotmann

18th November 2020

<https://www.isc.org>



Welcome

- Welcome to part five of our webinar series "the KEA DHCP Server"



About this Webinar

- Stork Dashboard for Kea
- A tour of Stork
- Other Monitoring options
- Logging in Kea
- Performance testing



Stork Dashboard for Kea



Stork Dashboard for Kea





What is Stork?

- Stork is a dashboard for Kea DHCP
 - monitoring of Kea
 - monitoring of Kea High-Availability state
 - alerting mechanisms that indicate failures, fault conditions, and other unwanted events



What is Stork?

- It is under active development
 - monthly releases
 - it is usable and useful
 - but not feature complete (as of November 2020)
 - there are rough edges



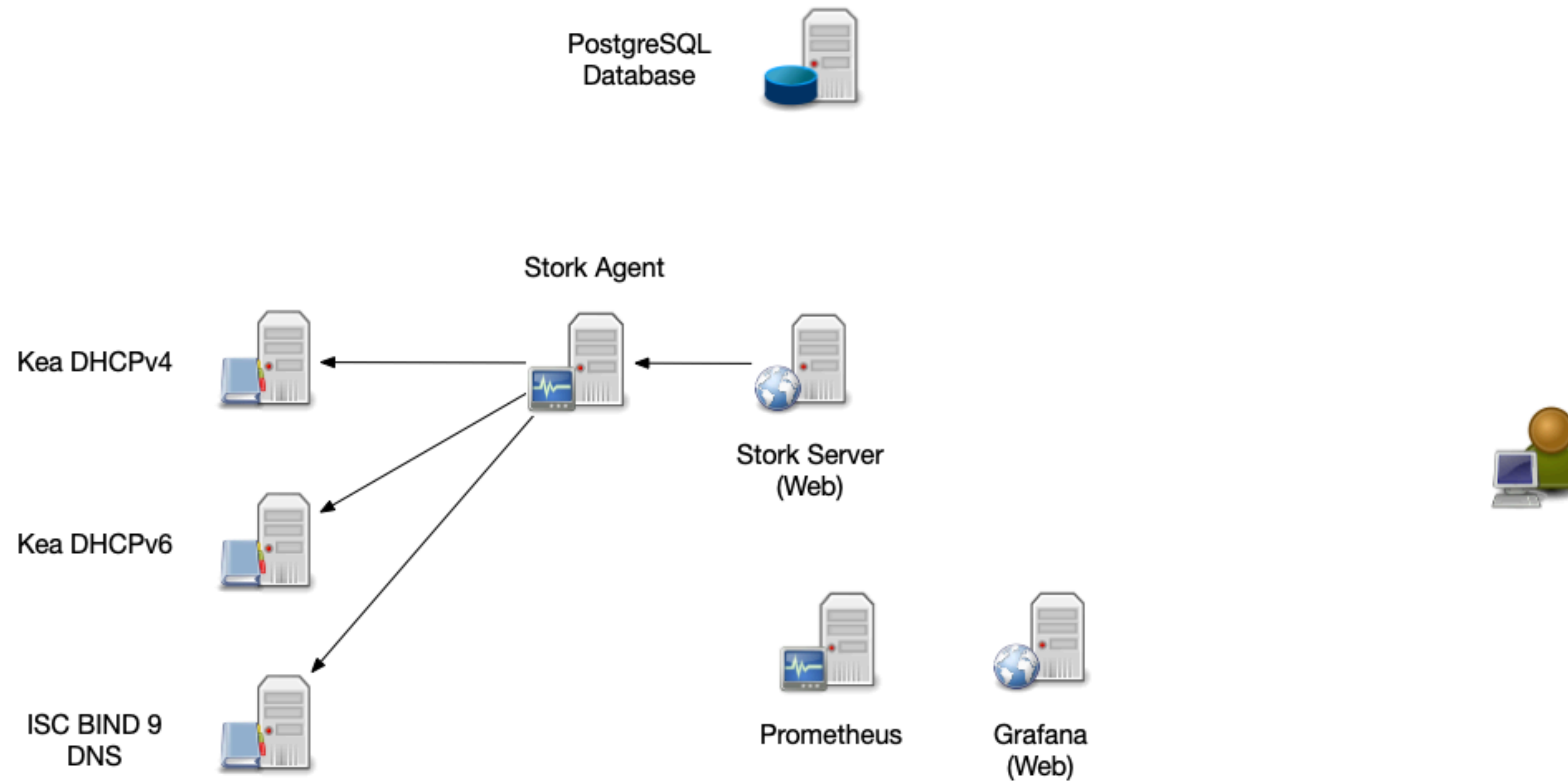
Platforms

- Stork is available for
 - Ubuntu Linux (18.04 and 20.04)
 - Fedora Linux 31, 32 and 33
 - RedHat/CentOS 7/8
 - macOS*
- Stork might work on other Unix(ish) platforms
- Stork can run co-located with a Kea service, or can run on a dedicated machine

* macOS is not and will not be officially supported but the developers use and test on macOS

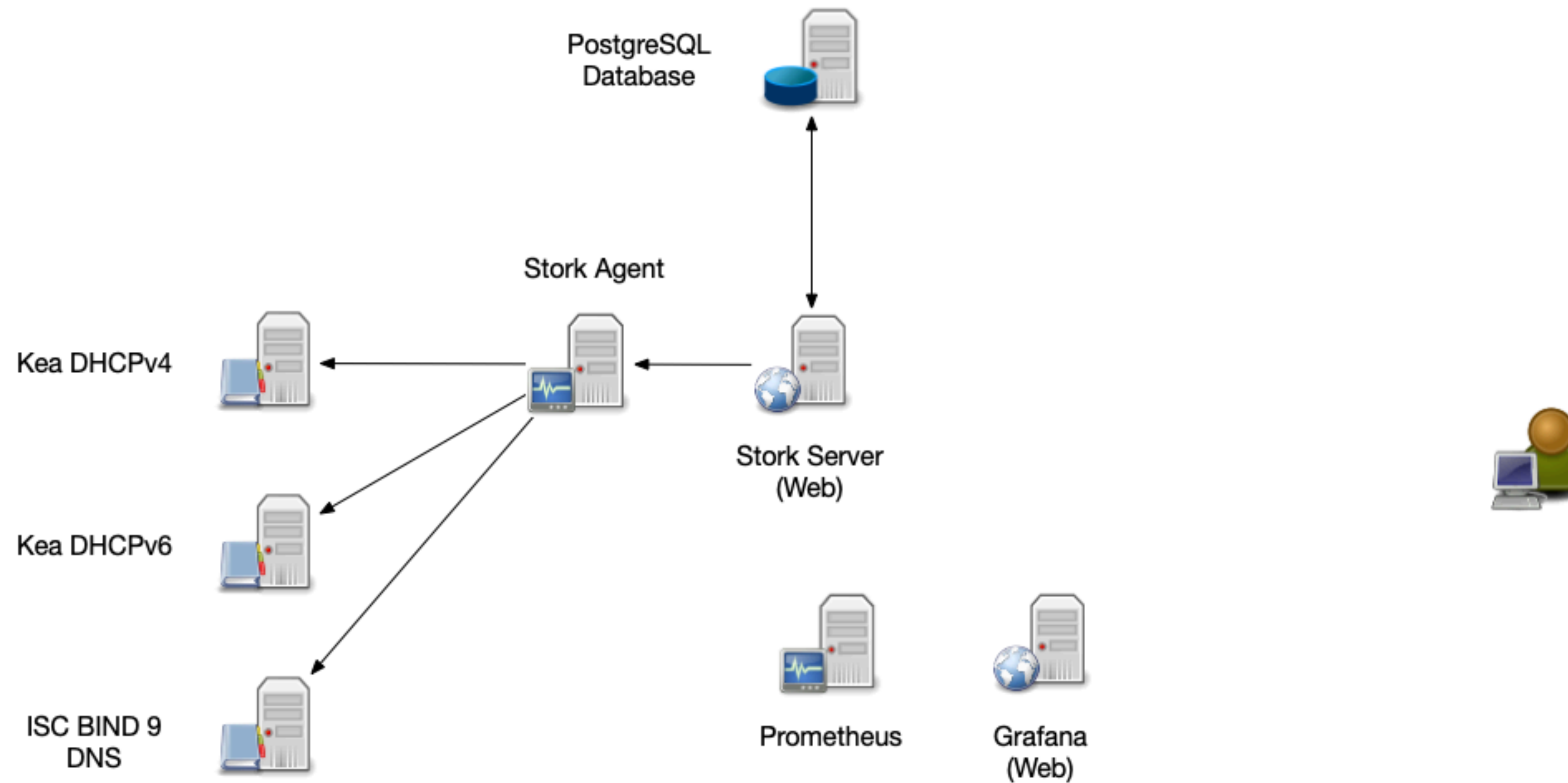


Architecture



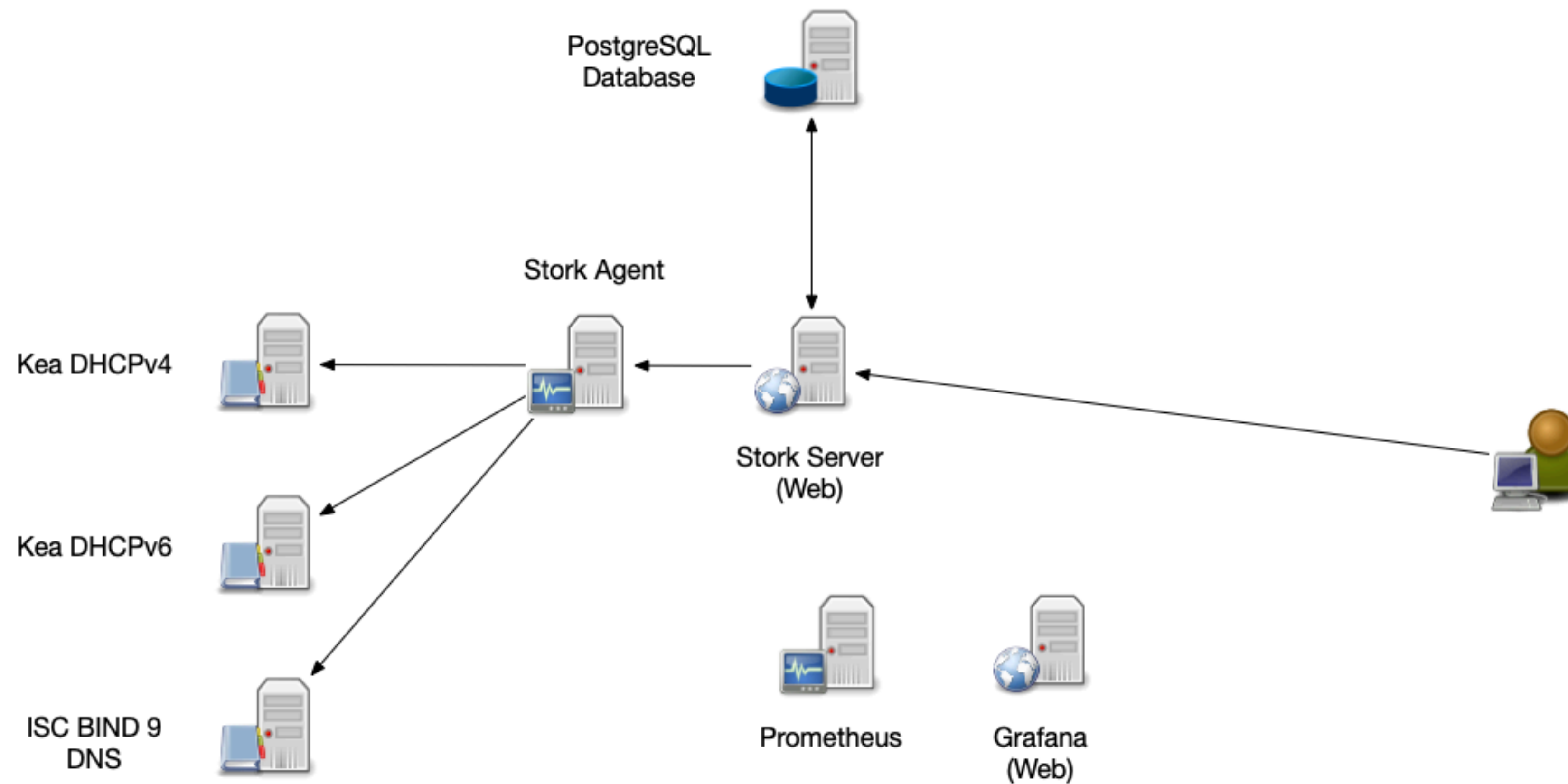


Architecture



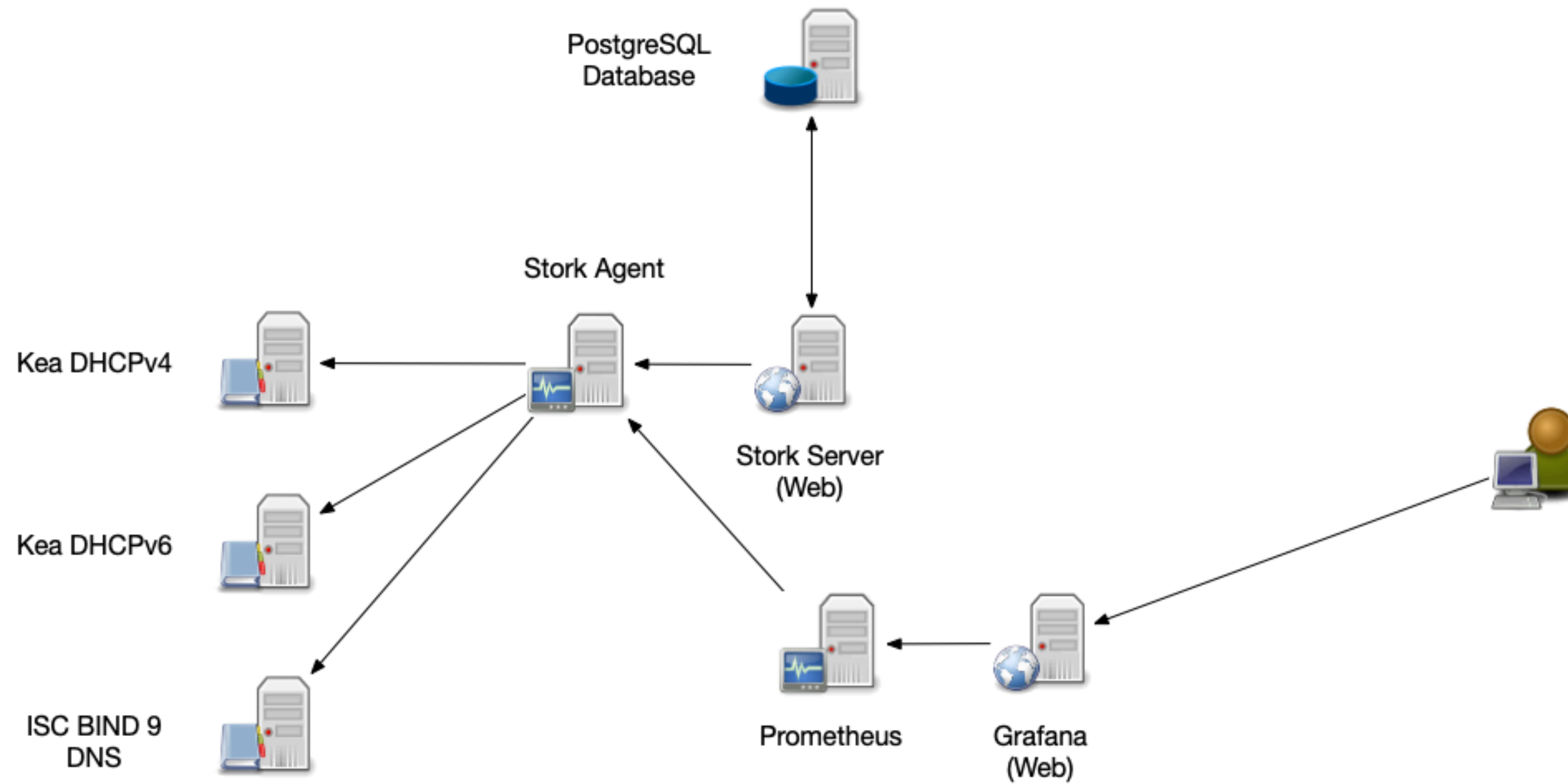


Architecture





Architecture





Requirements

- Kea Control Agent configured and running
- PostgreSQL Database (version 11 or later)



Installation (from packages)

- Packages for Stork are available in the ISC repositories from cloudsmith.io
- RedHat/CentOS/Fedora
- Debian/Ubuntu



Installation on CentOS 8

- Download and enable the repository data

```
# dnf install yum-utils pygpgme  
# rpm --import 'https://dl.cloudsmith.io/public/isc/stork/cfg/gpg/gpg.77F64EC28053D1FB.key'  
# curl -sLf 'https://dl.cloudsmith.io/public/isc/stork/cfg/setup/config.rpm.txt?distro=fedora&codename=29' > /tmp/isc-stork.repo
```

- inspect the repository data, then enable the repository

```
# less /tmp/isc-stork.repo  
# dnf config-manager --add-repo '/tmp/isc-stork.repo'
```



Installation on CentOS 8

- Update the repository database

```
# dnf makecache --enablerepo='isc-stork'
```

```
CentOS-8 - AppStream
```

```
CentOS-8 - Base
```

```
CentOS-8 - Extras
```

```
isc-stork
```

```
isc-stork
```

```
Importing GPG key 0x8053D1FB:
```

```
  Userid      : "Cloudsmith Package (isc/stork) <support@cloudsmith.io>"
```

```
  Fingerprint: 7AB5 064B 08F0 69A1 A5CC 500C 77F6 4EC2 8053 D1FB
```

```
  From        : https://dl.cloudsmith.io/public/isc/stork/cfg/gpg/gpg.77F64EC28053D1FB.key
```

```
Is this ok [y/N]: y
```

```
[...]
```

```
46 kB/s | 4.3 kB 00:00
```

```
35 kB/s | 3.9 kB 00:00
```

```
17 kB/s | 1.5 kB 00:00
```

```
186 B/s | 473 B 00:02
```

```
3.4 kB/s | 967 B 00:00
```




Installation on CentOS 8

- Install the Stork-Agent and -Server

```
# dnf install isc-stork-agent isc-stork-server
isc-stork
isc-stork-noarch
isc-stork-source
Dependencies resolved.
```

```
446 B/s | 473 B    00:01
700 B/s | 473 B    00:00
694 B/s | 473 B    00:00
```

Package	Architecture	Version	Repository	Size
Installing:				
isc-stork-agent	x86_64	0.13.0.201104144722-1	isc-stork	8.3 M
isc-stork-server	x86_64	0.13.0.201104144722-1	isc-stork	23 M

Transaction Summary

```
Install 2 Packages
```

```
Total download size: 31 M
Installed size: 68 M
Is this ok [y/N]:
```





PostgreSQL

- The Stork agent requires an PostgreSQL database to store configuration and historical monitoring data
- RedHat/CentOS 8 provides different version of the PostgreSQL database server in its AppStream repositories. Select the Version 12 (Version 10 is the default).



PostgreSQL

```
[root@kea-test ~]# dnf module enable postgresql:12
```

```
isc-stork
```

```
675 B/s | 473 B 00:00
```

```
isc-stork-noarch
```

```
687 B/s | 473 B 00:00
```

```
isc-stork-source
```

```
636 B/s | 473 B 00:00
```

```
Dependencies resolved.
```

```
=====
```

Package	Architecture	Version	Repository	Size
---------	--------------	---------	------------	------

```
=====
```

```
Enabling module streams:
```

```
postgresql
```

```
12
```

```
Transaction Summary
```

```
=====
```

```
Is this ok [y/N]:
```



PostgreSQL

```
# dnf module list postgresql
# dnf module list postgresql
Last metadata expiration check: 0:03:10 ago on Fri 13 Nov 2020 11:00:55 AM CET.
CentOS-8 - AppStream
Name                Stream                Profiles                Summary
postgresql          9.6                   client, server [d]     PostgreSQL server and client module
postgresql          10 [d]                client, server [d]     PostgreSQL server and client module
postgresql          12 [e]                client, server [d]     PostgreSQL server and client module

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```



PostgreSQL

```
# dnf install postgresql-server postgresql-contrib
Last metadata expiration check: 0:04:20 ago on Fri 13 Nov 2020 11:00:55 AM CET.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
Installing:				
postgresql-server	x86_64	12.1-2.module_el8.1.0+273+979c16e6	AppStream	5.5 M
Installing dependencies:				
libc	x86_64	60.3-2.el8_1	BaseOS	8.8 M
libpq	x86_64	12.4-1.el8_2	AppStream	195 k
postgresql	x86_64	12.1-2.module_el8.1.0+273+979c16e6	AppStream	1.4 M

Transaction Summary

```
Install 4 Packages
```

```
Total download size: 16 M
```

```
Installed size: 62 M
```

```
Is this ok [y/N]:
```





PostgreSQL

- Initialize the database

```
# postgresql-setup --initdb
* Initializing database in '/var/lib/pgsql/data'
* Initialized, logs are in /var/lib/pgsql/initdb_postgresql.log
```



PostgreSQL

- Start the PostgreSQL database system

```
# systemctl enable --now postgresql  
Created symlink /etc/systemd/system/multi-user.target.wants/postgresql.service → /usr/lib/  
systemd/system/postgresql.service.
```



PostgreSQL

- create the user stork and an empty database stork_db for Stork:

```
# su - postgres
$ psql postgres
psql (12.1)
Type "help" for help.
```

```
postgres=# CREATE USER stork WITH PASSWORD 'secure-password';
CREATE ROLE
postgres=# CREATE DATABASE stork_db;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE stork_db TO stork;
GRANT
postgres=# \c stork_db
postgres=# CREATE EXTENSION pgcrypto;
CREATE EXTENSION
postgres=# \q
```




Stork Agent configuration

- the Stork-Agent is configured via environment variables
- the variables are defined in `/etc/stork/agent.env` and will be read by the `init-system` or `systemd`



Stork Agent configuration

```
# address to bind ie. for listening
STORK_AGENT_ADDRESS=2001:db8:500::8547
STORK_AGENT_PORT=8547

# settings for exporting stats to Prometheus
STORK_AGENT_PROMETHEUS_KEA_EXPORTER_ADDRESS=192.0.2.47
STORK_AGENT_PROMETHEUS_KEA_EXPORTER_PORT=9547
STORK_AGENT_PROMETHEUS_KEA_EXPORTER_INTERVAL=60
```



Starting the Stork Agent

- once the Agent configuration is complete, the Stork-Agent can be started

```
# systemctl enable --now isc-stork-agent
Created symlink /etc/systemd/system/multi-user.target.wants/isc-stork-agent.service → /usr/lib/systemd/system/isc-stork-agent.service.
[root@kea-test ~]# systemctl status isc-stork-agent
● isc-stork-agent.service - ISC Stork Agent
   Loaded: loaded (/usr/lib/systemd/system/isc-stork-agent.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2020-11-13 11:23:28 CET; 9s ago
     Docs: man:stork-agent(8)
  Main PID: 5411 (stork-agent)
    Tasks: 6 (limit: 12210)
   Memory: 7.0M
    CGroup: /system.slice/isc-stork-agent.service
           └─5411 /usr/bin/stork-agent

Nov 13 11:23:28 kea-test systemd[1]: Started ISC Stork Agent.
Nov 13 11:23:28 kea-test stork-agent[5411]: INFO[2020-11-13 11:23:28]          main.go:75      Starting Stork Agent, version 0.13.0, build date 2020-11-04 14:47
Nov 13 11:23:28 kea-test stork-agent[5411]: INFO[2020-11-13 11:23:28] promkeaexporter.go:272 Prometheus Kea Exporter listening on 0.0.0.0:9547, stats pulling interval>
Nov 13 11:23:28 kea-test stork-agent[5411]: INFO[2020-11-13 11:23:28]          monitor.go:80   Started app monitor
```



Stork Server configuration

- the Stork-Server is configured via environment variables
- the variables are defined in `/etc/stork/server.env` and will be read by the init-system or `systemd`



Stork Server configuration

```
# database settings
STORK_DATABASE_HOST=192.0.2.55
STORK_DATABASE_NAME=stork_db
STORK_DATABASE_USER_NAME=stork
STORK_DATABASE_PASSWORD=secure-password

# ReST API settings
# STORK_REST_HOST=
# STORK_REST_PORT=
# STORK_REST_TLS_CERTIFICATE=
# STORK_REST_TLS_PRIVATE_KEY=
# STORK_REST_TLS_CA_CERTIFICATE=
STORK_REST_STATIC_FILES_DIR=/usr/share/stork/www
```



Starting the Stork Server

```
# systemctl enable --now isc-stork-server
Created symlink /etc/systemd/system/multi-user.target.wants/isc-stork-server.service → /usr/lib/systemd/system/isc-stork-server.service.

# systemctl status isc-stork-server
● isc-stork-server.service - ISC Stork Server
   Loaded: loaded (/usr/lib/systemd/system/isc-stork-server.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2020-11-13 12:22:13 CET; 2s ago
     Docs: man:stork-server(8)
  Main PID: 6984 (stork-server)
    Tasks: 7 (limit: 12210)
   Memory: 19.6M
   CGroup: /system.slice/isc-stork-server.service
           └─6984 /usr/bin/stork-server

Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13]           puller.go:38   starting Kea Hosts Puller
Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13]           puller.go:71   started Kea Hosts Puller
Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13]           puller.go:38   starting Kea Status Puller
Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13]           puller.go:71   started Kea Status Puller
Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13] eventcenter.go:118 event 'started Stork server'
Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13]           main.go:25    Starting Stork Server, version 0.13.0, build date 2020-11-04 14:47
Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13]           middleware.go:48 installed file server middleware
Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13]           middleware.go:68 installed SSE middleware
Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13]           middleware.go:19 installed logging middleware
Nov 13 12:22:13 kea-test stork-server[6984]: INFO[2020-11-13 12:22:13] restservice.go:241 started serving Stork Server           address="http://[::]:8080"
```



Prometheus

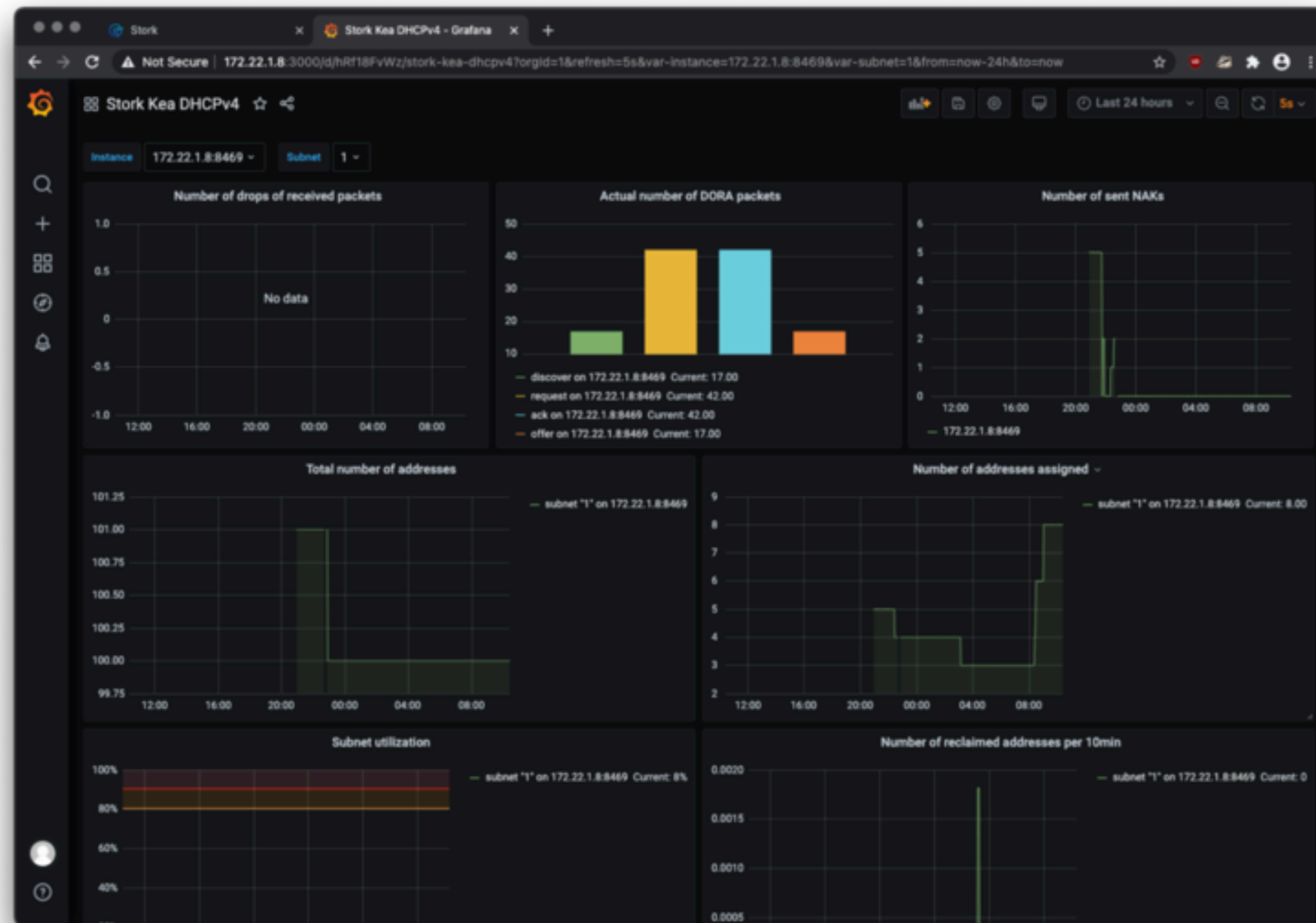
- Prometheus is a popular monitoring framework written in Go
 - <https://prometheus.io>
 - originally developed at SoundCloud
 - development is overseen by the Cloud Native Computing Foundation
<https://cncf.io/>
- Stork can export monitoring information towards Prometheus
 - Stork is an Prometheus Exporter
 - The use of Prometheus for Stork is **optional**



Grafana

- Grafana is a popular monitoring dashboard that can be used to visualize monitoring data from an Prometheus system
- <https://grafana.com/>
- Grafana provides additional visualization options for Kea DHCP data
- The use of Grafana for Stork is **optional**

Grafana





Tour a Stork

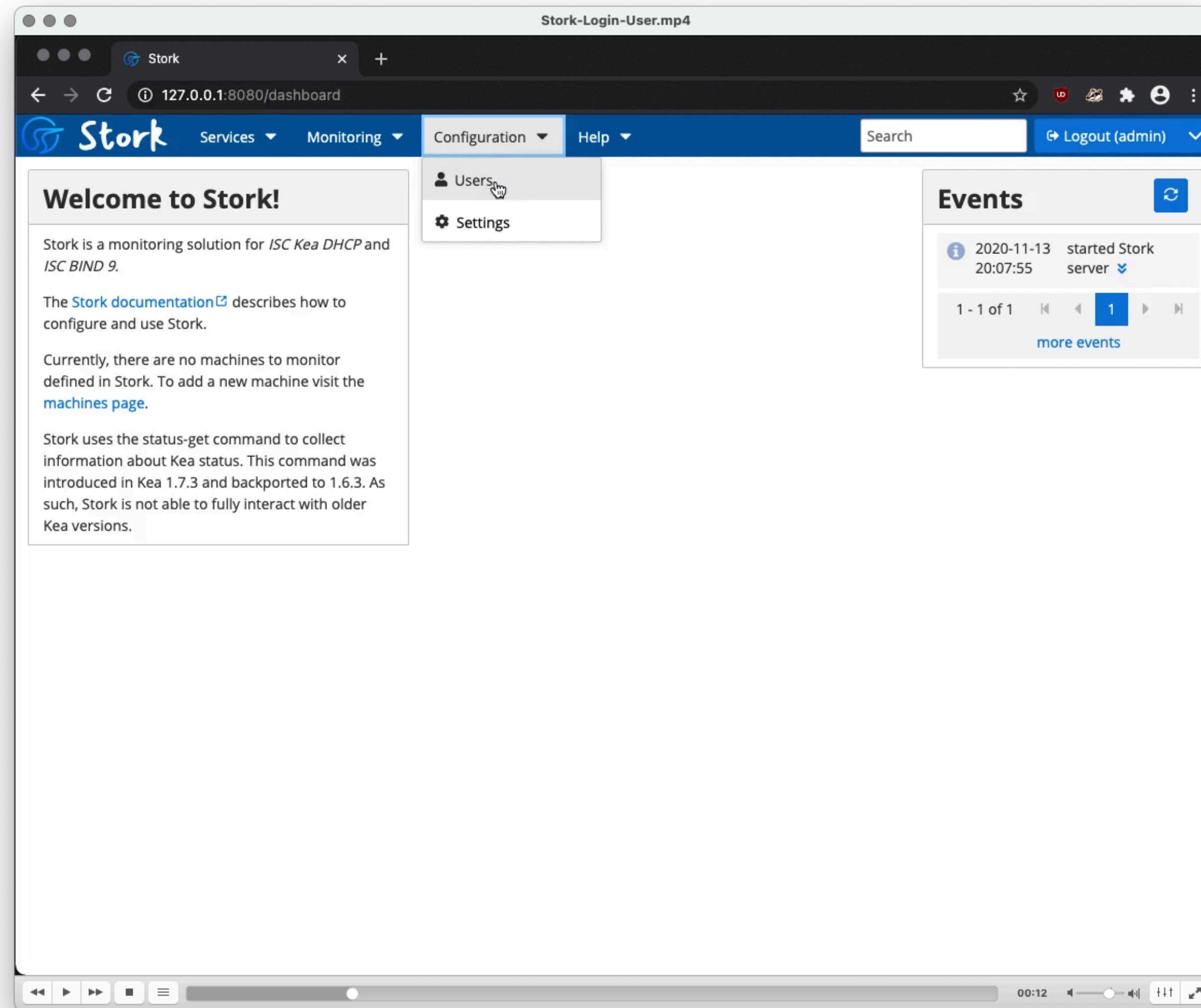


User Management





User Management





User Management

The screenshot shows a web browser window titled "Stork-Login-User.mp4" displaying the "Users" management page. The page has a blue header with the "Stork" logo and navigation menus for "Services", "Monitoring", "Configuration", and "Help". A search bar and a "Logout (admin)" button are also present. The breadcrumb trail indicates the current location: "Configuration > Users". Below the breadcrumb is a "Users" tab and a "+ Create User Account" button. A table lists the existing users:

Login	Email	First name	Last name	Group
admin <small>It's you</small>		admin	admin	super-admin

At the bottom of the table, there is a pagination control showing "1 of 1 pages", a page number "1", a dropdown menu set to "10", and the text "Total: 1 user".



User Management

Stork-Login-User.mp4

Configuration / Users - Stork

127.0.0.1:8080/users/new

Stork Services Monitoring Configuration Help Search Logout (admin)

Configuration > Users

Users new account

Creating new account

To create a new user account, please specify user login, first name, and last name. An email address is optional but strongly recommended. If an email is provided, the user can sign in either using the login or the email address. The password is mandatory and must be at least 8 characters long.

The user must be associated with an existing system group. Currently there are two groups available: super-admin and admin. Users belonging to the super-admin group have full control over the system, including creating and modifying user accounts. Users belonging to the admin group have similar permissions, with the exception that they are not allowed to create and/or modify user accounts. However, they are allowed to update their own passwords.

User account

Login*:

Email:

First name*:

Last name*:

Group*:

Password*:

Repeat password*:

Strong



User Management

Stork-Login-User.mp4

Configuration / Users - Stork

127.0.0.1:8080/users/list

Stork Services Monitoring Configuration Help

Configuration > Users

Users

+ Create User Account

Login	Email	First name	Last name	Group
admin <small>It's you</small>		admin	admin	super-admin
stork	stork-user@example.com	Stork	User	admin

1 of 1 pages | 1 | 10 | Total: 2 users

New user account created
Adding new user account succeeded



Adding Machines

Stork-Add-Machine.mp4

Services / Machines - Stork

127.0.0.1:8080/machines/all

Stork Services Monitoring Configuration Help

Search Logout (admin)

Services > Machines

Machines

Filter machines: name or any other field

+ Add New Machine Refresh

Hostname	Location ?	Agent Version	Daemons	CPUs	CPU Load ?	Total Memory [GB]	Memory Usage [%]	Error	Action
No machines found.									
Machines can be added by clicking the Add New Machine button at the top.									

1 of 1 pages 1 10 Total: 0 machines



Adding Machines

The screenshot shows a web browser window titled "Stork-Add-Machine.mp4" displaying the Stork web interface. The browser address bar shows "127.0.0.1:8080/machines/all". The Stork logo and navigation menu are visible at the top. The main content area is titled "Services > Machines" and contains a search filter, a table of machines, and an "Add New Machine" button. A modal dialog box titled "New Machine" is open, providing instructions and input fields for adding a new machine.

Filter machines: name or any other field

+ Add New Machine Refresh

Hostname	Location ?	Agent Version	Daemons	CPUs	CPU Load ?	Total Memory	Memory [%]	Error	Action
No machines found.									
Machines can be added by clicking the Add New M									

New Machine

First, install Stork Agent on the machine with Kea or BIND 9. This is described in [the Stork Agent instructions](#).

Machine is located by address and port where 'address' is the IP address or FQDN of machine, and port is the Stork Agent listening port. Port can be omitted; default value is 8080.

Address:

Port:

Cancel Add



Adding Machines

Stork-Add-Machine.mp4

Services / Machines - Stork

127.0.0.1:8080/machines/1

Stork DHCP Services Monitoring Configuration Help

Services > Machines

Machines agent-kea

agent-kea:8080

System Information

Address	agent-kea:8080
Hostame	agent-kea
Agent Version	0.13.0
CPUs	3
CPUs Load	0.16 0.46 0.76
Memory	3 GiB
Used Memory	25 %
Uptime	? days
OS	linux
Platform Family	debian
Platform	ubuntu
Platform Version	18.04
Kernel Version	5.4.0-52-generic
Kernel Arch	x86_64
Virtualization Role	guest
Virtualization System	docker
Host ID	ec234c61-0c62-4188-aebd-4fe12211954b
Last Visited	2020-11-13 20:38:17

Get Latest State

Applications

Kea App

Version 1.8.0

✓ DHCPv4 ⊗ DHCPv6 ⊗ DDNS ✓ CA

Events

- 2020-11-13 20:38:17 added daemon [4] dhcp6 to app [1] kea 1.8.0
- 2020-11-13 20:38:17 added daemon [3] dhcp4 to app [1] kea 1.8.0
- 2020-11-13 20:38:17 added daemon [2] d2 to app [1] kea 1.8.0
- 2020-11-13 20:38:17 added daemon [1] ca to app [1] kea 1.8.0
- 2020-11-13 20:38:17 added app [1] kea 1.8.0 on machine [1] agent-kea
- 2020-11-13 20:38:17 added machine [1] agent-kea

1 - 6 of 6 1 more events



Adding Machines

The application is hosted on the machine: [agent-kea](#) Refresh App

DHCPv4 DHCPv6 DDNS CA

Monitoring Host Reservations Subnets Shared Networks

Overview

Version: 1.8.0
Version Ext: 1.8.0
tarball
linked with:
log4cplus 1.1.2
OpenSSL 1.1.1 11 Sep 2018
database:
MySQL backend 9.3, library 5.7.32
PostgreSQL backend 6.1, library 100014
Memfile backend 2.1
Hooks:
/usr/lib/x86_64-linux-gnu/kea/hooks/libdhcp_lease_cmds.so
/usr/lib/x86_64-linux-gnu/kea/hooks/libdhcp_stat_cmds.so
Uptime: 32 minutes 2 seconds
Last Reloaded At: 2020-11-13 20:07:53

High Availability

High Availability is not enabled on this server.

Loggers

Logger	Severity	Output Location
kea-dhcp4	debug	stdout
kea-dhcp4	debug	/tmp/kea-dhcp4.log

Events

2020-11-13 20:38:17 added daemon [3] dhcp4 to app [1] kea 1.8.0

1 - 1 of 1 1 [more events](#)



Status Information

DHCP Identifiers	IP Addresses	IPv6 Prefixes	Hostname	Global/Subnet	AppID @ Machine
duid=01:02:03:04:05	192.0.2.103			192.0.2.0/24	1 @ agent-kea config
client-id=01:0a:0b:0c:0d:0e:0f	192.0.2.105			192.0.2.0/24	1 @ agent-kea config
client-id=01:11:22:33:44:55:66	192.0.2.102		special-snowflake	192.0.2.0/24	1 @ agent-kea config
client-id=01:12:23:34:45:56:67	192.0.2.104			192.0.2.0/24	1 @ agent-kea config
hw-address=1a:1b:1c:1d:1e:1f	192.0.2.101			192.0.2.0/24	1 @ agent-kea config
flex-id=73:30:6d:45:56:61:4c:75:65	192.0.2.106			192.0.2.0/24	1 @ agent-kea config
client-id=aa:aa:aa:aa:aa:aa	10.0.0.222			global	2 @ agent-kea-many-subnets config 1 @ agent-kea config
hw-address=ee:ee:ee:ee:ee:ee	10.0.0.123			global	2 @ agent-kea-many-subnets config 1 @ agent-kea config



Monitoring Service Health

The screenshot shows the Stork DHCP Dashboard. The main content is divided into several sections:

- DHCPv4:** Shows 6922 subnets and 2 shared networks (frog and mouse). Statistics indicate 0% of addresses are used.
- DHCPv6:** Shows 0 subnets and 0 shared networks.
- Services Status:** A table listing the status of various Kea agents. The 'agent-kea-ha1' service is highlighted with a red box and is marked as 'unavailable'.
- Events:** A list of recent events, including daemon reachability and communication failures.

Host	[ID] App Version	Daemon	Status	RPS (15min)	RPS (24h)	HA State	Detected Failure w/HA	Uptime
agent-kea	[1] Kea 1.8.0	dhcp4	✓	1	1	⊘ not configured		55 minutes 4 seconds
agent-kea-many-subnets	[2] Kea 1.7.3	dhcp4	✓			⊘ not configured		16 minutes 15 seconds
agent-kea-ha1	[4] Kea 1.7.8	dhcp4	✗			✗ unavailable	never	7 minutes 36 seconds
agent-kea-ha2	[5] Kea 1.7.8	dhcp4	✓			✓ hot-standby	2020-11-13 20:55:55	6 minutes 52 seconds



Pool Utilization

The screenshot shows the Stork DHCP Dashboard with the following sections:

- DHCPv4:** Subnets: 6922. Shared Networks: 2. Statistics: Addresses 0 / 452951227 (0% used), Declined 0.
- DHCPv6:** Subnets: 0. Shared Networks: 0. Statistics: Addresses 0 / 0 (0% used), Prefixes 0 / 0 (0% used), Declined 0.
- Services Status:** Table with columns: Host, [ID] App Version, Daemon, Status, RPS (15min), RPS (24h), HA State, Detected Failure w/HA, Uptime.
- Events:** Log of system events including daemon reachability and communication failures.

Host	[ID] App Version	Daemon	Status	RPS (15min)	RPS (24h)	HA State	Detected Failure w/HA	Uptime
agent-kea	[1] Kea 1.8.0	dhcp4	✓	1	1	⊘ not configured		45 minutes 5 seconds
agent-kea-many-subnets	[2] Kea 1.7.3	dhcp4	✓			⊘ not configured		5 minutes 46 seconds
agent-kea-ha1	[4] Kea 1.7.8	dhcp4	✓			✓ hot-standby	never	3 minutes 33 seconds
agent-kea-ha2	[5] Kea 1.7.8	dhcp4	✓			✓ hot-standby	2020-11-13 20:50:25	2 minutes 1 seconds



Pool Utilization

Stork-Kea-HA-2.mp4

DHCP / Subnets - Stork

127.0.0.1:8080/dhcp/subnets?dhcpVersion=4

Stork DHCP Services Monitoring Configuration Help Search Logout (admin)

> DHCP > Subnets

Filter subnets: subnet or any other file Protocol: DHCPv4

Subnet ID	Subnet	Addresses			Pools	Shared Network	AppID @ Machine
		Total	Assigned	Used %			
1	192.0.5.0/24	50	6	12 %	192.0.5.1-192.0.5.50	frog	1 @ agent-kea
2	192.0.6.0/24	110	1	0.9 %	192.0.6.1-192.0.6.40 192.0.6.61-192.0.6.90 192.0.6.111-192.0.6.150	frog	1 @ agent-kea
3	192.0.7.0/24	50	5	4 %	192.0.7.1-192.0.7.50	frog	1 @ agent-kea
4	192.0.8.0/24	50	50	2 %	192.0.8.1-192.0.8.50	frog	1 @ agent-kea
5	192.0.9.0/24	50	1	2 %	192.0.9.1-192.0.9.50	frog	1 @ agent-kea
6	192.1.15.0/24	50	50	24 %	192.1.15.1-192.1.15.50	mouse	1 @ agent-kea
7	192.1.16.0/24	150	1	0.6 %	192.1.16.1-192.1.16.50 192.1.16.51-192.1.16.100 192.1.16.101-192.1.16.150	mouse	1 @ agent-kea
8	192.1.17.0/24	245	1	0.4 %	192.1.17.1-192.1.17.20 192.1.17.21-192.1.17.40 192.1.17.41-192.1.17.60 192.1.17.66-192.1.17.80 192.1.17.81-192.1.17.100 192.1.17.101-192.1.17.120 192.1.17.121-192.1.17.140 192.1.17.141-192.1.17.160 192.1.17.161-192.1.17.180 192.1.17.181-192.1.17.200 192.1.17.201-192.1.17.220 192.1.17.221-192.1.17.240 192.1.17.241-192.1.17.243 192.1.17.244-192.1.17.246	mouse	1 @ agent-kea



Pool Utilization

The screenshot shows the Stork DHCP Shared Networks interface. The browser address bar indicates the URL is 127.0.0.1:8080/dhcp/shared-networks?dhcpVersion=4. The page title is 'Stork-Kea-HA-2.mp4'. The navigation menu includes 'DHCP', 'Services', 'Monitoring', 'Configuration', and 'Help'. The current page is 'Shared Networks'. A search filter is set to 'network or any other f' and the protocol is 'DHCPv4'. The table below shows the utilization of shared networks for two hosts: 'frog' and 'mouse'.

Name	Addresses			Subnets	AppID @ Machine
	Total	Assigned	Used %		
frog	310	63	20.3 %	192.0.6.0/24 192.0.7.0/24 192.0.8.0/24 192.0.5.0/24 192.0.9.0/24	1 @ agent-kea
mouse	445	52	11.6 %	192.1.17.0/24 192.1.16.0/24 192.1.15.0/24	1 @ agent-kea

1 of 1 pages | 10 | Total: 2 shared networks



Pool Utilization

The screenshot shows the Stork DHCP Dashboard. The main content is divided into several sections:

- DHCPv4:** Shows 6922 subnets. A table lists subnets with their utilization percentages. A red box highlights the first five subnets: 192.0.8.0/24 (100% used), 192.1.15.0/24 (100% used), 192.0.2.0/24 (97% used), 192.0.5.0/24 (12% used), and 192.0.7.0/24 (10% used). It also shows shared networks for 'frog' (5 subnets, 20.3% used) and 'mouse' (3 subnets, 11.6% used).
- DHCPv6:** Shows 0 subnets and 0 shared networks.
- Statistics:** Shows addresses used (309 / 452951227, 0% used) and declined addresses (0).
- Services Status:** A table showing the status of various Kea agents.
- Events:** A list of system events, including daemon reachability and communication failures.

Host	[ID] App Version	Daemon	Status	RPS (15min)	RPS (24h)	HA State	Detected Failure w/HA	Uptime
agent-kea	[1] Kea 1.8.0	dhcp4	✓	1	1	⊘ not configured		46 minutes 8 seconds
agent-kea-many-subnets	[2] Kea 1.7.3	dhcp4	✓			⊘ not configured		6 minutes 47 seconds
agent-kea-ha1	[4] Kea 1.7.8	dhcp4	✓			✓ hot-standby	never	4 minutes 36 seconds
agent-kea-ha2	[5] Kea 1.7.8	dhcp4	✓			✓ hot-standby	2020-11-13 20:50:25	3 minutes 5 seconds



HA-Health Status

The screenshot shows the Kea Admin interface for a high-availability setup. The 'Overview' section displays version 1.7.8 and uptime of 5 minutes 38 seconds. The 'Events' section shows several messages, including daemon unreachability and communication failures. The 'High Availability' section compares the local server (primary, online) with the remote server (standby, offline). A tooltip for the local server's 'partner-down' state explains that it responds to DHCP queries because the partner server is not functional.

Local server	Remote server Kea@127.0.0.1
Status time: 2020-11-13 20:55:25	Status time: 2020-11-13 20:54:56
Status checked: 11 seconds ago	Status checked: 40 seconds ago
Role: primary	Role: standby
Control status: ✓ online	Control status: ✗ offline
Heartbeat status: ✓ ok	Heartbeat status: ✗ failed
State: ✗ partner-down	State: ✗ unavailable
Scopes served: server1	
Last in partner-down: 2020-11-13 20:55:25	
Unacked clients: n/a	
Connecting clients: n/a	
Analyzed packets: n/a	

Help for state
This server now responds to all DHCP queries because it detected that partner server is not functional!



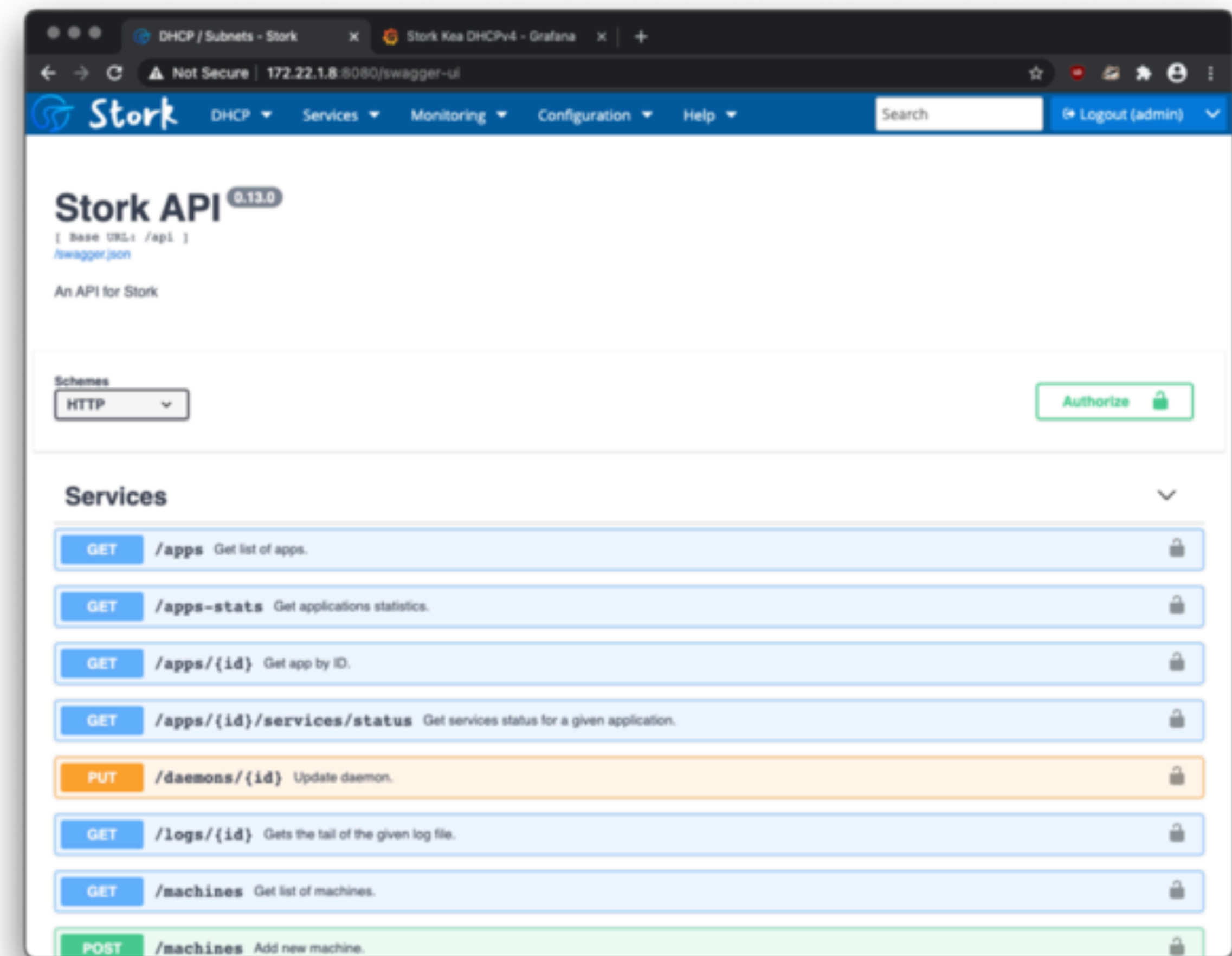
Stork REST API

- Like all parts of Kea, Stork has an extensive REST/JSON API to automate management
 - add/delete machines
 - manage users
 - fetch log files
 - fetch events
 - get reservations
 - get DHCP statistics on subnets



Stork REST API

- The API documentation can be found in the **Help** menu





Stork REST API

- The API documentation can be found in the **Help** menu

The screenshot shows a web browser window displaying the Stork REST API documentation. The browser tabs include 'DHCP / Subnets - Stork' and 'Stork API'. The address bar shows '172.22.1.8:8080/api/docs#operation/getSubnets'. The page content is as follows:

- Search** bar at the top left.
- Navigation Menu** on the left with categories: AUTHENTICATION, SERVICES, EVENTS, USERS, DHCP (selected), SEARCH, SETTINGS, and GENERAL.
- Endpoint List** under DHCP:
 - Get list of DHCP host reservations.
 - Get overview of whole DHCP state.
 - Get list of DHCP shared networks.
 - Get list of DHCP subnets.** (highlighted)
- Endpoint Details** for 'Get list of DHCP subnets':
 - Method:** GET /subnets
 - Description:** A list of subnets is returned in items field accompanied by total count which indicates total available number of records for given filtering parameters.
 - Parameters:** Query Parameters
 - start:** integer, First record to retrieve.
 - limit:** integer, Number of records to retrieve.
 - appid:** integer, Limit returned list of subnets to these which are served by given app ID.
 - dhcpVersion:** integer, Limit returned list of subnets to either DHCPv4 (4) or DHCPv6 (6).
 - text:** string, Limit returned list of subnets to the ones containing indicated text.
 - Responses:**
 - 200 List of subnets (highlighted)
 - default generic error response
 - Response Samples:** A JSON snippet showing the structure of the response:

```
{  "items": [    + ( - )  ],  "total": 0}
```



Other Monitoring



Leases from a memfile

- Mike Miller has created two shell scripts that list the DHCPv4 and DHCPv6 leases from a memfile database
- Homepage:
<https://archive.mgm51.com/sources/kea-scripts.html>

```
% kea-show-leases4.sh
```

IPAddr	HWAddr	Lease	Start	Renew	Expire	Hostname	State
10.20.2.7	z0:z1:d9:z5:7c:36	14400	20150905T113158	20150905T133158	20150905T153158	host1.	0
10.20.2.6	0z:1z:d9:z5:7c:35	14400	20150905T112931	20150905T132931	20150905T152931	.	0
10.20.2.234	zz:75:0z:1a:a0:98	14400	20150905T112029	20150905T132029	20150905T152029	.	0
172.20.2.222	az:z3:cz:c4:4b:00	14400	20150905T110758	20150905T130758	20150905T150758	.	0



Leases from a SQL database

- The presenter of this webinar has created a simple python3 script that lists the leases from a PostgreSQL Kea lease database

- Source:

`https://git.sr.ht/~cstrotm/kea-list-leases`

```
% kea-list-leases.py
```

```
DHCPv4 leases: 6
```

IP Address	Hostname	HW Addr	Client-ID	Subnet ID	lifetime	expire
192.0.2.23	macbookair	14:c2:33:fd:ba:fb	01:14:c2:33:fd:ba:fb	1	14400	2020-11-18T14:11:17+01:00
192.0.2.80	phone	00:02:13:55:5e:23		1	14400	2020-11-18T14:33:32+01:00
192.0.2.120	linux-fedora	3c:09:14:7a:6a:67	01:3c:09:14:7a:6a:67	1	14400	2020-11-18T13:24:08+01:00
192.0.2.121		80:47:23:e6:38:32		1	14400	2020-11-18T14:48:28+01:00
192.0.2.122	openbsd	a8:61:b6:d1:ee:6e	01:a8:61:b6:d1:ee:6e	1	14400	2020-11-18T14:48:42+01:00
192.0.2.242	nas	00:12:47:30:c4:de	01:00:12:47:30:b4:de	1	14400	2020-11-18T14:47:31+01:00



Process Monitoring - keactrl

- on the local machine, the command keactrl can be used to check the status of the Kea processes

```
$ keactrl status
DHCPv4 server: active
DHCPv6 server: inactive
DHCP DDNS: active
Control Agent: active
Netconf agent: inactive
Kea configuration file: /usr/local/etc/kea/kea.conf
Kea DHCPv4 configuration file: /usr/local/etc/kea/kea-dhcp4.conf
Kea DHCPv6 configuration file: /usr/local/etc/kea/kea-dhcp6.conf
Kea DHCP DDNS configuration file: /usr/local/etc/kea/kea-dhcp-ddns.conf
Kea Control Agent configuration file: /usr/local/etc/kea/kea-ctrl-agent.conf
Kea Netconf configuration file: /usr/local/etc/kea/kea-netconf.conf
keactrl configuration file: /usr/local/etc/kea/keactrl.conf
```



Process Monitoring - systemd

- On a Linux machine with systemd, the status of the Kea processes can be read from the systemd process

```
# systemctl status kea-dhcp6
● kea-dhcp6.service - Kea DHCPv6 Service
   Loaded: loaded (/etc/systemd/system/kea-dhcp6.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2020-11-12 22:50:14 CET; 1 day 10h ago
     Docs: man:kea-dhcp6(8)
  Main PID: 244200 (kea-dhcp6)
    Tasks: 1 (limit: 11784)
   Memory: 5.6M
      CPU: 22.572s
   CGroup: /system.slice/kea-dhcp6.service
           └─244200 /opt/kea/sbin/kea-dhcp6 -c /opt/kea/etc/kea/kea-dhcp6.conf
```

```
Nov 12 22:50:14 home01 systemd[1]: Started Kea DHCPv6 Service.
```

```
Nov 12 22:50:14 home01 kea-dhcp6[244200]: 2020-11-12 22:50:14.813 INFO [kea-dhcp6.dhcp6/244200.140267216668800] DHCP6_STARTING Kea DHCPv6 server version 1.9.1
```



Process Monitoring via Systemd API

- systemd exposes the state of managed services via the DBUS API
 - a monitoring system can read the DBUS API information
 - Example: Monitoring systemd services in realtime with Chronograf
<https://devconnected.com/monitoring-systemd-services-in-realtime-with-chronograf/>
 - Example: Prometheus exporter for systemd services
https://github.com/povilasv/systemd_exporter



Monitoring via Kea API

- Kea exposes a REST/JSON API
 - This API can be used to monitor the health and function of the Kea services (independent from Stork)
 - Python Kea exporter for Prometheus
<https://pypi.org/project/kea-exporter/>
 - Source code of the Prometheus Kea exporter:
<https://github.com/mweineit/kea-exporter>



DHCP Function Monitoring

- dhcping is a simple tool to test if a DHCP server responds to DHCP requests and returns a lease
 - it requests a lease (DHCPREQUEST) or DHCP option information (DHCPINFORM) from a DHCP Server
 - after obtaining a lease, it will release the lease immediately
 - Original Homepage:
`http://www.mavetju.org/unix/general.php`
 - Updated source:
`https://github.com/nean-and-i/dhcping`



DHCPing

```
% sudo ./dhcping -v -s 192.0.2.1 -h 01:02:03:04:05:05 -c 192.0.2.145
```

```
DHCP REQUEST  
packet 250 bytes
```

```
nop: 1  
htype: 1  
hlen: 6  
hops: 0  
xid: ef0aaf5f  
secs: 0  
flags: 0  
ciaddr: 192.0.2.145  
yiaddr: 0.0.0.0  
siaddr: 0.0.0.0  
giaddr: 0.0.0.0  
chaddr: 01:02:03:04:05:05  
sname :  
fname :  
option 53 DHCP message type  
    DHCP message type: 3 (DHCPREQUEST)  
option 50 Request IP address  
    Requested IP address: 192.0.2.145
```



DHCping

```
Got answer from: 192.0.2.1
packet 300 bytes

nop: 2
htype: 1
hlen: 6
hops: 0
xid: ef0aaf5f
secs: 0
flags: 7f80
ciaddr: no entry found
yiaddr: 0.0.0.0
siaddr: 0.0.0.0
giaddr: 0.0.0.0
chaddr: 01:02:03:04:05:05
sname :
fname :
option 53 DHCP message type
    DHCP message type: 6 (DHCPNAK)
option 54 DHCP Server identifier
    Server identifier: 192.0.2.1
option 56 Message
```



DHCping

```
DHCP RELEASE
packet 250 bytes

nop: 1
htype: 1
hlen: 6
hops: 0
xid: ef0aaf5f
secs: 0
flags: 0
ciaddr: 192.0.2.145
yiaddr: 0.0.0.0
siaddr: 0.0.0.0
giaddr: 0.0.0.0
chaddr: 01:02:03:04:05:05
sname :
fname :
option 53 DHCP message type
    DHCP message type: 7 (DHCPRELEASE)
option 54 DHCP Server identifier
    Server identifier: 192.0.2.1
```




DHCPtest

- another DHCP test tool

- written in D

- Source:

<https://github.com/CyberShadow/dhcptest>

```
% ./dhcptest --query
dhcptest v0.7 - Created by Vladimir Panteleev
https://github.com/CyberShadow/dhcptest
Run with --help for a list of command-line options.

Listening for DHCP replies on port 68.
Sending packet:
  op=BOOTREQUEST chaddr=2E:78:71:CA:DA:26 hops=0 xid=8DDD0A71 secs=0 flags=8000
  ciaddr=0.0.0.0 yiaddr=0.0.0.0 siaddr=0.0.0.0 giaddr=0.0.0.0 sname= file=
  1 options:
    53 (DHCP Message Type): discover
Received packet from 192.0.2.8:67:
  op=BOOTREPLY chaddr=2E:78:71:CA:DA:26 hops=0 xid=8DDD0A71 secs=0 flags=8000
  ciaddr=0.0.0.0 yiaddr=192.0.2.115 siaddr=0.0.0.0 giaddr=0.0.0.0 sname= file=
  9 options:
    53 (DHCP Message Type): offer
    1 (Subnet Mask): 255.255.255.0
    3 (Router Option): 192.0.2.1
    6 (Domain Name Server Option): 192.0.2.8, 172.16.1.105
    15 (Domain Name): home.example.com
    51 (IP Address Lease Time): 14400 (4 hours)
    54 (Server Identifier): 192.0.2.8
    58 (Renewal (T1) Time Value): 3600 (1 hour)
    59 (Rebinding (T2) Time Value): 7200 (2 hours)
```



Monitoring for Pool depletion

- Performance suffers at very high pool utilization, because Kea is checking every address in order to see if it is available
- If an DHCP pool runs full, there is a risk that DHCP clients will not get an IP address lease and cannot join the network

How to deal with pool depletion



- if you encounter address pool depletion, check for the reasons
 - lease time too high for the number of DHCP clients in the network
 - machines are not releasing their lease on shutdown
 - malicious/buggy DHCP client software



Countermeasures to address pool depletion

- configure the Microsoft DHCP clients to release their leases on shutdown
 - can be done via DHCP option:
https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-dhcpe/4cde5ceb-4fc1-4f9a-82e9-13f6b38d930c
 - useful in public places where clients are not coming back
- consider switching to IPv6
 - make the pool as large as the current IPv4 Internet ;)



Logging



Kea logging configuration

- All Kea services provide flexible logging:
<https://kea.readthedocs.io/en/latest/arm/logging.html>
- Log output can be written to one or more targets
 - to syslog
 - to a file
 - to stdout or stderr



Kea logging configuration

- Example: Logging to stdout and into a file

```
"loggers": [{
  "name": "kea-dhcp4",
  "output_options": [
    {
      "output": "stdout",
      "pattern": "%-5p %m\n"
    }, {
      "output": "/var/log/kea/kea-dhcp4.log",
      "maxsize": 1048576,
      "maxver": 10
    }
  ],
  "severity": "INFO",
  "debuglevel": 0
}]
[...]
```



Kea Logger

- The Kea Log-Messages are sent from different logging modules
 - the logging modules create a logging hierarchy
 - The Root-Logger is named after the Kea service process
 - Below the Root-Logger are one or more logging modules that can be used to sent specific logging information to other log-targets, or change other logging parameters such as the severity
- a list of Loggers supported by Kea servers and hook-libraries can be found in the Kea documentation

<https://kea.readthedocs.io/en/latest/arm/logging.html#the-name-string-logger>



Kea Logger

- The name of the logging module that created a log message can be found in the log output (when using the default log pattern for files)

```
kealog-output-example.txt
Root-Logger  Logger-Module
2020-11-15 09:57:42.166 INFO [kea-dhcp4.hooks/269605.139863672895616] HOOKS_LIBRARY_LOADED hooks library libdhcp_stat_cmds.so successfully loaded
2020-11-15 09:57:42.166 INFO [kea-dhcp4.dhcp4/269605.139863672895616] DHCP4_CONFIG_COMPLETE DHCPv4 server has completed configuration: ...
2020-11-15 09:57:42.166 INFO [kea-dhcp4.dhcpsrv/269605.139863672895616] DHCP4_PGSQLE_DB opening ...
2020-11-15 09:57:42.842 WARN [kea-dhcp4.dhcp4/269605.139863672895616] DHCP4_MULTI_THREADING_INFO enabled: no, number of threads: 0, queue size: 0
2020-11-15 09:57:42.843 INFO [kea-dhcp4.dhcp4/269605.139863672895616] DHCP4_STARTED Kea DHCPv4 server version 1.9.1 started
2020-11-15 09:57:43.059 INFO [kea-dhcp4.commands/269605.139863672895616] COMMAND_RECEIVED Received command 'version-get'
2020-11-15 09:57:43.061 INFO [kea-dhcp4.commands/269605.139863672895616] COMMAND_RECEIVED Received command 'status-get'
2020-11-15 09:57:43.063 INFO [kea-dhcp4.commands/269605.139863672895616] COMMAND_RECEIVED Received command 'config-get'
2020-11-15 09:57:50.402 INFO [kea-dhcp4.commands/269605.139863672895616] COMMAND_RECEIVED Received command 'statistic-get-all'
```



Logging to *syslog*

- Using the output parameter of `syslog` will send the log messages of the chosen logger to the `syslog` daemon
- If a different service name should be used for the `syslog` messages, the service name can be specified in the format `syslog:name`

```
[...]
  "loggers": [{
    "name": "kea-dhcp4",
    "output_options": [
      { "output": "syslog:dhcp4" }
    ],
    "severity": "WARN", "debuglevel": 0
  }]
[...]
```



Logging to a file

- When logging to a file, the parameter `output` specifies the file name
- file rollover can be specified with the **maxsize** (size of log-file in bytes) and **maxver** (number of log-file generations)



Logging Message Format

- The content of the log messages can be controlled with the `pattern` option
 - The pattern used for each message is described by a string containing one or more format components as part of a text string
 - In addition to the components the string may contain any other arbitrary text you find useful.
 - The Log4Cplus documentation provides information on the pattern format string:
<https://log4cp1us.sourceforge.io/>



Logging Message Format

- Example: the pattern definition below ...

```
{  
  "output": ".....",  
  "pattern": "%D{%Y-%m-%d %H:%M:%S.%q} %-5p [%c/%i.%t] %m\n"  
},
```

- ... will create a log entry similar to this one:

```
2019-08-05 14:27:45.871 DEBUG [kea-dhcp4.dhcpsrv/8475.12345] DHCPDRV_TIMERMGR_START_TIMER starting timer: reclaim-expired-leases
```



Kea and Systemd Journal

- when a Kea service is running under control of systemd, the logging output written to stdout will be stored in the systemd journal

```
[...]
  "loggers": [{
    "name": "kea-dhcp4",
    "output_options": [
      {
        "output": "stdout",
        "pattern": "%-5p %m\n"
      }
    ],
    "severity": "INFO",
    "debuglevel": 0
  }]
[...]
```



Kea and Systemd Journal

- Systemd-Journal entries can be queried with a filter language
 - easier than filtering through log files (if you don't know awk and perl)
 - systemd-journald data can be sent via an encrypted and authenticated connection to a central systemd-journald log host
 - see the journalctl documentation for details

```
# journalctl --since today -u kea-dhcp4 --grep DHCP4_LEASE_ADVERT
-- Logs begin at Fri 2020-09-18 11:20:45 CEST, end at Sat 2020-11-14 09:24:50 CET. --
Nov 14 00:00:00 home01 kea-dhcp4[244218]: INFO DHCP4_LEASE_ADVERT [hwtype=1 00:0d:93:29:2d:30], cid=[01:00:0d:93:29:2d:30], tid=0xfa7d9468: lease 192.0.2.114 will be a>
Nov 14 00:00:04 home01 kea-dhcp4[244218]: INFO DHCP4_LEASE_ADVERT [hwtype=1 00:0d:93:29:2d:30], cid=[01:00:0d:93:29:2d:30], tid=0xe998dcab: lease 192.0.2.114 will be a>
Nov 14 00:05:13 home01 kea-dhcp4[244218]: INFO DHCP4_LEASE_ADVERT [hwtype=1 2e:78:71:ca:da:26], cid=[no info], tid=0x8ddd0a71: lease 192.0.2.115 will be advertised
Nov 14 02:15:06 home01 kea-dhcp4[244218]: INFO DHCP4_LEASE_ADVERT [hwtype=1 14:c2:13:ed:ba:fb], cid=[01:14:c2:13:ed:ba:fb], tid=0xda0e88bc: lease 192.0.2.23 will be ad>
Nov 14 04:16:09 home01 kea-dhcp4[244218]: INFO DHCP4_LEASE_ADVERT [hwtype=1 14:c2:13:ed:ba:fb], cid=[01:14:c2:13:ed:ba:fb], tid=0xda0e88be: lease 192.0.2.23 will be ad>
Nov 14 06:01:03 home01 kea-dhcp4[244218]: INFO DHCP4_LEASE_ADVERT [hwtype=1 14:c2:13:ed:ba:fb], cid=[01:14:c2:13:ed:ba:fb], tid=0xda0e88c0: lease 192.0.2.23 will be ad>
Nov 14 08:04:24 home01 kea-dhcp4[244218]: INFO DHCP4_LEASE_ADVERT [hwtype=1 14:c2:13:ed:ba:fb], cid=[01:14:c2:13:ed:ba:fb], tid=0xda0e88c3: lease 192.0.2.23 will be ad>
```



Kea API authorization logging

- Starting with Kea 1.9.1, it is possible to restrict the Kea API commands to authorized users
- the authorization information will be logged with the `kea-ctrl-agent.http` logger:

```
# ./kea-ctrl-agent -c simple.json
20.10.15 14:05:16.550 INFO [kea-ctrl-agent.http/174909] HTTP_CLIENT_REQUEST_AUTHORIZED received HTTP request authorized for 'admin'
20.10.15 14:05:16.550 INFO [kea-ctrl-agent.commands/174909] COMMAND_RECEIVED Received command 'list-commands'
```




Debug-Logging

- Quick option: start KEA DHCP4 in debug mode from the command line. This will automatically enable the highest debugging level
 - On a busy server, this will create too much debug information (see next slide for an alternative)

```
[kea-server]# systemctl stop kea-dhcp4  
[kea-server]# kea-dhcp4 -d -c /etc/kea/kea-dhcp4.conf
```



Debug-Logging

- Alternative: enable debug logging on a specific logger only

```
"loggers": [{
  "name": "kea-dhcp4",
  "output_options": [
    { "output": "syslog:dhcp4" }
  ],
  "severity": "WARN", "debuglevel": 0
}, {
  "name": "kea-dhcp4.flex-id-hooks",
  "output_options": [ {
    "output": "/var/log/kea/kea-dhcp4-flex-id.log"
  } ],
  "severity": "DEBUG",
  "debuglevel": 55
} ]
[...]
```



Performance testing



Kea perfdhcp tool

- The Kea development team has published the performance measurement tool (called **perfdhcp**) that is used to do DHCP performance testing for Kea
- Documentation:
<https://kea.readthedocs.io/en/latest/man/perfdhcp.8.html?highlight=perfdhcp>
- Usage examples:
<https://users.isc.org/~tomasz/perfdhcp/dhcp-perf-guide.html#perfdhcp-commandline-examples>



Next Webinars

- 2nd December - Kea DHCP - Migrating to Kea from ISC DHCP



Resources

- Video: Stork Management Dashboard for Kea DHCP
<https://www.youtube.com/watch?v=5aF9NBiKhqQ>
- Stork Documentation
<https://kea.readthedocs.io/projects/Stork>
- Stork Project Page
<https://gitlab.isc.org/isc-projects/stork>
- Stork mailing list
<https://lists.isc.org/mailman/listinfo/stork-users>



Questions and Answers