Extending Kea with Hooks

29 June 2016
Logistics

- Webinar is scheduled for 1 hour
- Webinar will contain slides, code walkthrough and videos demonstrating hooks in action.
- This session will be recorded and posted at https://www.isc.org/mission/webinars/
- Participants are muted to improve audio quality for everyone.
- We want questions! Please enter into the WebEx Q&A tab
  - We will try to answer questions at the end of the presentation.
Presenters

Marcin Siodelski
DHCP Software Engineer
ISC
Previous Webinar

08 June 2016

Webinar: Getting Started with Kea
Eddy Winstead

Recording available at:
https://www.isc.org/mission/webinars/
What hooks are?

- Dynamically loaded libraries
- Customize your server to do anything you need in your deployment
- Have unrestricted access to the server data structures and functions
- Written by ISC, you or other users
- Written in C++
- Implement/use well defined and documented API
Terminology

- **Hook Point:**
  - A place within Kea code where specific callout is invoked.

- **Callout:**
  - A function within hook library, invoked by the server at specific hook point.

- **Hook Library:**
  - A dynamically loaded library implementing selected callouts.
Documentation

- Kea User’s Guide (How to use Hooks):
  - http://kea.isc.org/docs/kea-guide.html#hooks-libraries

- Kea Developer’s Guide (How to write hooks):
http://kea.isc.org/
Processing with Hooks

Kea Process

- Kea server
- Hook Pt. A
- Hook Pt. B
- Hook Pt. C
- Hook Pt. D
- Hook Pt. E

User Library 1

- Custom library1
- F1()
- F2()

User Library 2

- Custom library2
- G1()
- G2()
DHCPv6 Server Hooks

- `buffer6_receive` - after receiving data on interface
- `pkt6_receive` - after parsing DHCP message
- `subnet6_select` - after selecting a subnet
- `lease6_select` - when lease has been selected
- `lease6_renew` - during lease renewal
- `lease6_decline` - during declining a lease
- `lease6_release` - during releasing a lease
- `pkt6_send` - before creating wire format of response
- `buffer6_send` - after creating wire format of response
- `lease6_expire` - when lease is found expired
- `lease6_recover` - when declined lease is no longer declined
Access to Parameters in Callouts

- A reference to "CalloutHandle" object is passed to each callout.
- CalloutHandle provides access to "context" and arguments.
- At each hook point, the callouts receive well known set of arguments/objects.
- Objects to be retrieved or modified are accessed using well known names provided as strings.
- The Kea Developer’s Guide specifies arguments for each hook point.
Examples: Access Parameters

// Fetch the inbound packet.
Pkt6Ptr query;
handle.getArgument("query6", query);

// Fetch server response.
Pkt6Ptr response;
handle.getArgument("response6", response);

// Assign last configured subnet for a client.
Subnet6Ptr subnet = subnets->back();
handle.setArgument("subnet6", subnet);

// Store the id we search with so it is available down the road.
handle.setContext("query_user_id", duid);

// Get the user id saved from the query packet.
DuidPtr duid;
handle.getContext("query_user_id", duid);
The Next Step Status

- Callout returns status code to indicate what the server’s next step should be:
  - NEXT_STEP_CONTINUE - continue processing
  - NEXT_STEP_SKIP - skip next processing step (Developer’s Guide contains detailed explanation for each defined hook point)
  - NEXT_STEP_DROP - drop the packet
Hooks Supplied with Kea

- Standard hook libraries are shipped with Kea code: `<kea-sources>/src/hooks/dhcp/`
- There is currently one hook library shipped with Kea sources: `user_chk`
- We’ll use `user_chk` library to demonstrate how to implement hook library.
The „user_chk” Library

- Use external source of information to recognize registered and unregistered DHCP clients.
- Select different subnets for registered and unregistered clients.
- Create outcome file containing traces from subnet and address assignment.
The „user_chk” Library

Note 1: Both files must exist prior to loading the library!
Look into the code #1

- We will now look into the implementation of the pkt6_receive and the subnet6_select callouts.
Video 1

- Create „registry” and „outcome” files required by the user_chk library.
- Start the server.
- Client obtains address from the unprivileged subnet 3000::/64 because the client is not registered.
- Register the client. The Client should be assigned an address from the privileged subnet 2001:db8:1::/64.
Look into the code #2

- We will now look into the code of the pkt6_send callout.
Video 2

- Client is renewing its lease.
- Hook library overrides the bootfile option with the value specified in the registry file.
- The value in the registry is modified, the server should respond with a different value.
- The client is removed from the registry file and the server should respond with the default value specified in the configuration.
Look into the code #3

- We will see how easy it is to replace the „registration” with MySQL database.
Video 3

- Client renews and obtains the boot file option with the value of /tmp/v6bootfile.
- MySQL database contains a registration entry for the client and specifies the bootfile value /tmp/bootfile-from-mysql
- Introducing modification to the code to use MySQL database rather than the file.
- The bootfile value from MySQL is used.