Making Routing Decisions with OpenSIPS

Peter Kelly / pkelly@sourcevox.com

Who we are

- VoIP and OpenSIPs software development and consultancy
- Based in UK
- Some of larger customers are
  - Localphone
    - Retail ITSP offering (VoIP accounts, apps, DIDs in UK, US, Europe, Worldwide)
    - Over 1,000,000 users
  - Magic Telecom
    - US Faclities based CLEC
  - Voxbeam
    - Wholesale, A-Z Termination, VoIP reseller
    - US CLEC
    - Terminate ~20,000,000 mins/week internationally
- We use
  - OpenSIPS
  - Asterisk/FreeSWITCH
  - FreeSWITCH
  - RabbitMQ
  - Redis
  - Hadoop
  - Homer
  - Sangoma

Making Routing Decisions with OpenSIPS….

… what is a routing decision?

- Every business/user and type of business/user has their own definitions of routing decisions
- OpenSIPS gives you the flexibility to be able to make the decisions you need to make

Making Routing Decisions with OpenSIPS….

… overview of this talk.

- Introduction to some features and modules OpenSIPS provides.
- Examples of how all these features can come together in a powerful way.
- Some decision making!

Example Scenario

1. Authenticate calls by IP
2. Lookup user and find preferences
3. See if we need to make any external requests for the call.
4. Route the call to a carrier based on prefix
5. Deal with failures

1. Authenticate by IP … permissions module

- Check against known IP list

```
+---------------+------+--------------+
| ip            | port | context_info |
+---------------+------+--------------+
| 95.75.247.45  | 5060 | 434:1:5      |
+---------------+------+--------------+
```

- context_info gives us back any information we would like to retrieve against the match.

```
check_source_address("1" "$avp(info)");
```

1. Authenticate by IP … permissions module

- Use s.select string function to extract information from context string

```
$avp(user_id) = $(avp(info){s.select,0,:});
$avp(active) = $(avp(info){s.select,1,:});
$avp(max_spend) = $(avp(info){s.select,2,:});
```

- Now we have
  - validated the IP
  - used string manipulation features in OpenSIPS to get some useful information.
  - Part 1 complete!

2. Lookup user and find preferences

- We know
  - the User ID
  - The account is active so we want to allow the call to continue.
  - The max price the user will pay for a call

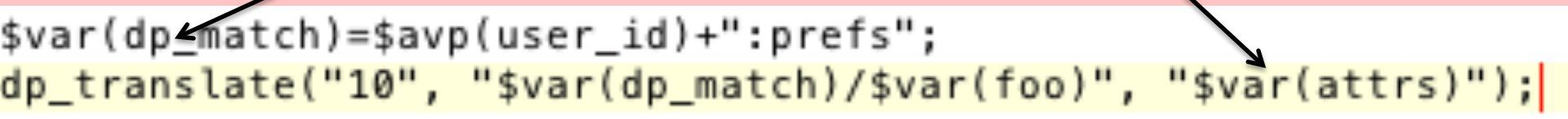
- Now we want to
  - Check if the user needs their calls to be "dipped"
  - Find out which "routing table" to use later on when sending the call onwards.

- Use dialplan module for this.

2. Lookup user and find preferences … dialplan module

- Use to translate strings into other strings
- In memory, very fast lookups

dpid	match_op	match_exp	attrs
10	0	454:prefs	dip=1;tariff=11



```
$var(dp_match)=$avp(user_id)+":prefs";
dp_translate("10", "$var(dp_match)/$var(foo)", "$var(attrs)");
```

2. Lookup user and find preferences … dialplan module

- Dialplan is more flexible than the simple example:

dpid	match_op	match_exp	subst_exp	repl_exp
11	1	^01226	^0([0-9]{1,})	44\1

- match_op = regex
- match_exp, match strings starting 01226
- subst_exp = extract 1226 part
- repl_exp = replace with 44 then 1226 part (441226)

2. Lookup user and find preferences … dialplan module

- Back to original dialplan result, we can once again extract information from attributes

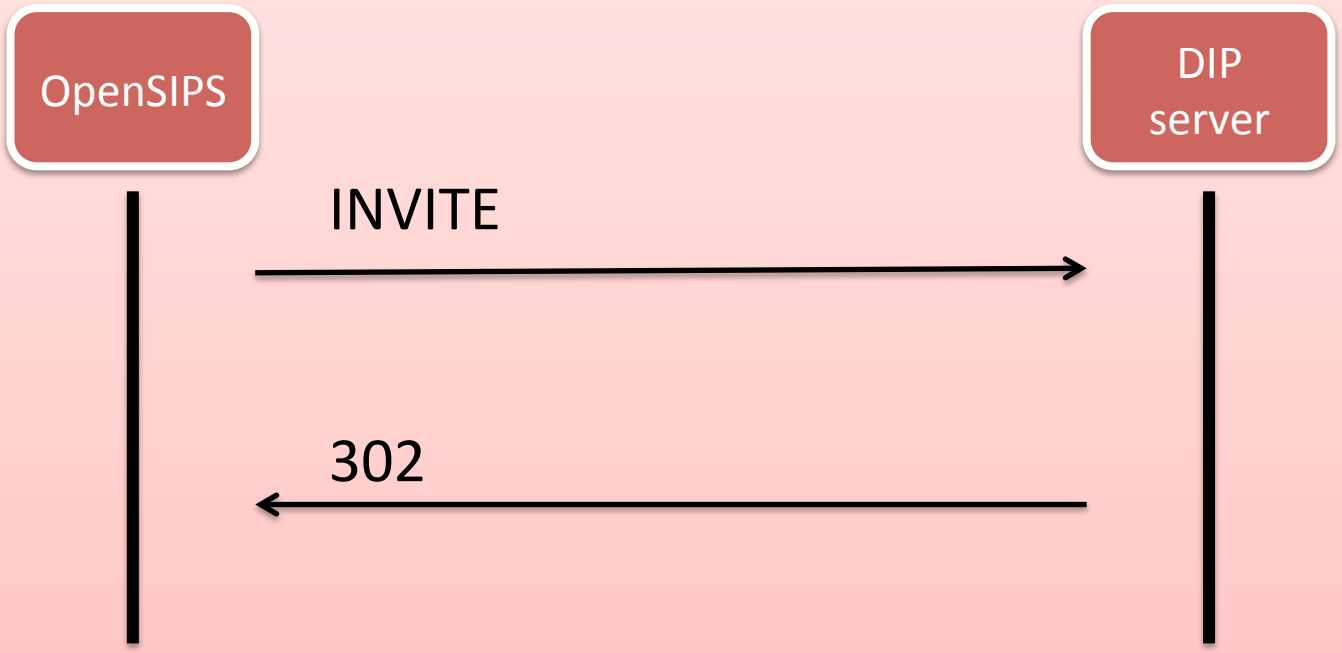
```
+-------------------+
| attrs             |
+-------------------+
| dip=1;tariff=11   |
+-------------------+
```

```
$avp(tariff) = $(var(attrs){param.value,tariff});
$avp(dip) = $(var(attrs){param.value,dip});
```

2. Lookup user and find preferences … dialplan module

- Now we know we need to do an external “dip” for information and the user uses tariff 11 (all will become clear later!)

3. Make external request (LRN dip)




```
Contact: <sip:7604314190@74.124.28.150;rn=8587649979;npdi>.
```

- Information embedded in Contact
- Extracted using string manipulation

```
$var(lrn_uri) = $(<reply>ct{param.value,rn});
```

3. Make external request

- Send dip

```
$rd = 'sip:dip.dipserver.com';
t_relay();
```

- Arm failure route

```
t_on_failure("FAIL_LRN");
```

- Handle failure

```
route[FAIL_LRN] {
    $var(lrn_uri) = $(<reply>ct{param.value,rn});
    route(RT_DROUTING);
}
```

3. Quick Review – we now know:

- User ID
- Account State
- Maximum spend per minute

```
$avp(user_id) = $(avp(info){s.select,0,:});
$avp(active) = $(avp(info){s.select,1,:});
$avp(max_spend) = $(avp(info){s.select,2,:});
```

- Preferred Tariff for the user
- Should the call be dipped?

```
$avp(tariff) = $(var(attrs){param.value,tariff});
$avp(dip) = $(var(attrs){param.value,dip});
```

- Dip Result

```
$avp(lrn_uri) = $(<reply>ct{param.value,rn});
```

3. Quick Question:

"How does the Failure route still now about the old variables?"

```
modparam("tm", "onreply_avp_mode", 1)
```

- **OpenSIPS can remember avp's for the duration of the transaction!**
- **Very useful to know and make use of.**

4. Route call to carrier … drouting module

- Drouting module is at its heart a longest match prefix module
- Resolves prefixes to gateways or carriers
- Once again has attributes which are very powerful

4. Route call to carrier … drouting module

Gateways

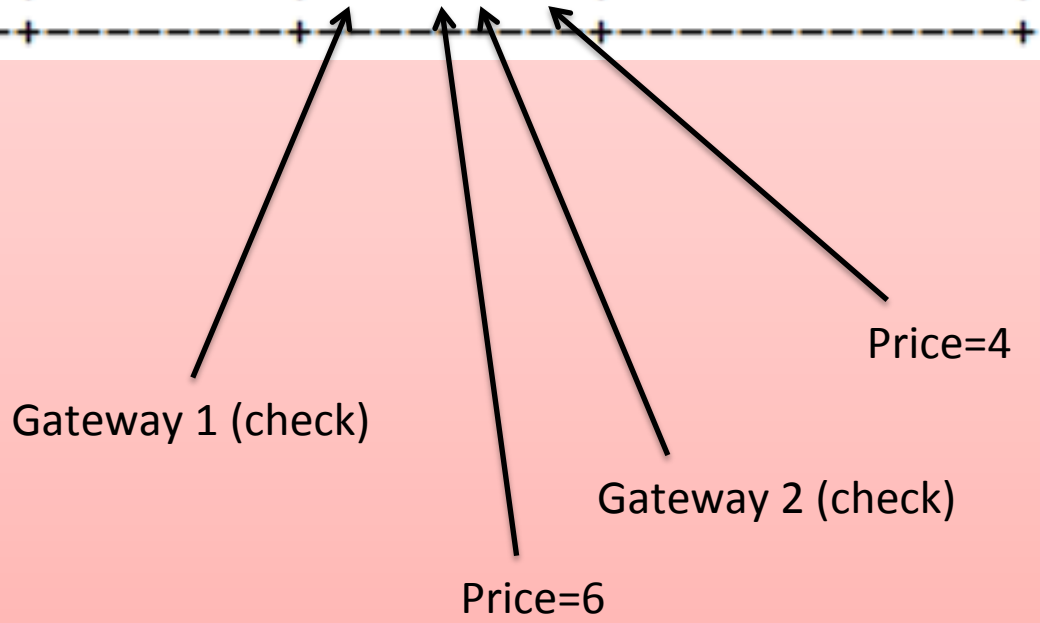
```
mysql> select gwid, address, description from dr_gateways;
+------+---------+-------------+
| gwid | address | description |
+------+---------+-------------+
| 1    | 1.2.3.4 | BT          |
| 2    | 5.6.7.8 | Vodafone    |
| 3    | 9.0.1.2 | AT&T        |
| 4    | 3.4.5.6 | Verizon     |
+------+---------+-------------+
4 rows in set (0.00 sec)
```

Rules

```
+---------+--------+--------+---------+-------------+
| groupid | prefix | gwlist | attrs   | description |
+---------+--------+--------+---------+-------------+
| 11      | 447    | 1,2    | 1,6:2,4 | UK Mobile   |
+---------+--------+--------+---------+-------------+
```

4. Route call to carrier … drouting module

groupid	prefix	gwlist	attrs	description
11	447	1,2	1,6:2,4	UK Mobile



4. Route call to carrier … drouting module

- We call do_routing
- drouting module does its job (selects a carrier)

```
if(!do_routing("$avp(tariff)", "", "", "$var(rule_attrs)")) {
    send_reply("503", "Service Unavailable");
}
```

Loaded earlier

We have our attributes!

- Now we do our job by using the attributes

4. Route call to carrier … drouting module

- Left to its own devices OpenSIPS will route the call to gateway 1.
- We can inspect the attributes and decide if we want to do this

```
$(avp(rule_attrs){s.select,0,:});
```

= 1,6

```
$(avp(rule_attrs){s.select,1,:});
```

= 2,4

```
route[RT_ROUTING] {

    #This gives us our first carrier
    do_routing(...);

    $var(cost_ok) = 0;
    $var(count) = 0;

    while($var(cost_ok) == 0) {
        route(RT_CHECK_COST);
        $var(count) = $var(count) + 1;
        use_next_gw();
    }

}

route[RT_CHECK_COST] {

    # Extract "$var(count)"th index, gives us 1,6 then 2,4
    $var(this_gw) = $(avp(rule_attrs){s.select,$var(count),:});

    # Extract second index, gives us 6 then 4
    $var(cost) = $(var(this_gw){s.select,1,,});

    if($var(cost) < $avp(max_spend)) {
        $var(cost_ok) = 1;
    }
}
```

Conclusion

- Used permissions module to retrieve account information
- Used dialplan module to retrieve
  - "DIP Information"
  - Tariff information
- Used attributes to our advantage
- Used persistent AVP's
- Made external requests and continued routing
- Routed a call to a carrier whilst making extra decisions

Conclusion

- Demonstrates how flexible you can be in your OpenSIPS usage.
- Could have used DB lookups
- May not naturally assume further complex routing can be done from a failure
- May not naturally assume complex actions in scripts
  - Looping
  - Recursion
- You can get seriously fast results using commodity hardware by using these tools (tens of thousands cps)

Questions?

pkelly@sourcevox.com
www.sourcevox.com