

VERSATILE SIP ROUTING

Kamailio Embedded Scripting Languages



ClueCon 2019

Daniel-Constantin Mierla
Co-Founder Kamailio Project
kamailio.org
asipto.com
[@miconda](https://twitter.com/miconda)



The Open Source SIP Server

www.kamailio.org

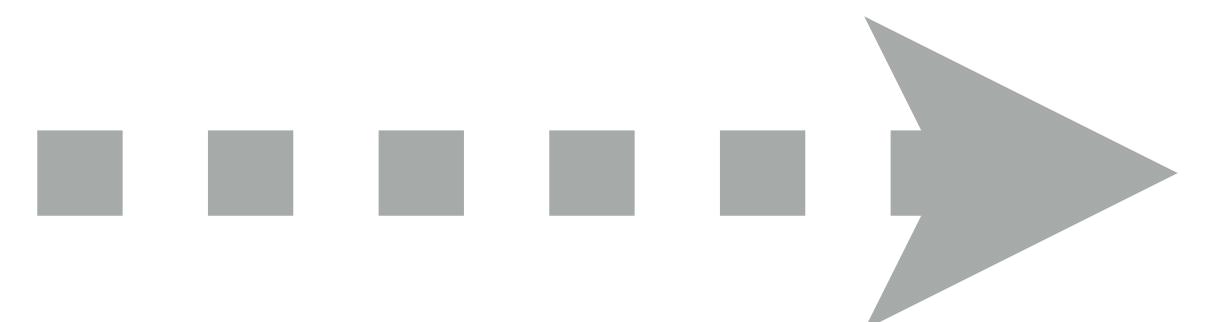


The **SRM** Of Your RTC Service



aka
the
SIP Reliability Manager

www.kamailio.org



KEMI Framework

Empowering Kamailio Since 2016

- introduced in Kamailio v5.0.0
 - use other scripting languages for writing SIP routing logic (route blocks)
 - exports a function one time and becomes available in embedded interpreters
 - ***allows reloading routing script without restart***
 - ***enables the use of an extensive number of extensions available in the scripting languages***
 - ***lua, python2, python3, javascript, ruby, squirrel***



Seeing Is Believing



```
324 -- Routing to foreign domains
325 function ksr_route_sipout()
326     if KSR.is_myself(KSR.pv.get("$ru")) then return 1; end
327
328     KSR.hdr.append_hf("P-Hint: outbound\r\n");
329     ksr_route_relay();
330     KSR.x.exit();
331 end
332 # Routing to foreign domains
333 def ksr_route_sipout(self, msg):
334     if KSR.is_myself(KSR.pv.get("$ru")) :
335         return 1;
336
337     KSR.hdr.append("P-Hint: outbound\r\n");
338     self.ksr_route_relay(msg);
339     return -255;
```

298 # Routing to foreign domains
299 route[SIPOUT] {
300 if (uri==myself) return;
301
302 append_hf("P-hint: outbound\r\n");
303 route(RELAY);
304 exit;
305 }



KEMI Performances For v5.2

- **Done in November 2018, just before the release of Kamailio v5.2**

- The focus was to compare the execution time of `request_route {}` from native `kamailio.cfg` with the equivalent KEMI callback functions for processing SIP REGISTER requests with user authentication
 - Done for Lua and Python

- **Testing Environment**

- Server: Intel NUC 7i7DNHE; CPU: I7-8650U @ 1.90GHz; CPU Cores: 4; CPU Threads: 8; Memory: 16GB
- Operating system: Debian Testing
- SIP traffic tool: sipp

- **Testing metrics**

- Each iteration was done with sending 20 000 requests
- For each iteration was collected
 - cnt - number of SIP messages processed (counter)
 - sum - the sum of execution times for `request_route` or `ksr_request_route()` (microseconds)
 - min - the minimum execution time for `request_route` or `ksr_request_route()` (microseconds)
 - max - the maximum execution time for `request_route` or `ksr_request_route()` (microseconds)
 - avg - the average execution time for `request_route` or `ksr_request_route()` (microseconds)



<https://www.kamailio.org/wiki/kemi/performance-tests/5.2.x>

KEMI Performances For v5.2

Lua

cnt: 61197
sum: 3844555
min: 22
max: 1852
avg: 62.8226



Native

cnt: 62752
sum: 4184315
min: 17
max: 1948
avg: 66.6802



Python

cnt: 60785
sum: 4400362
min: 20
max: 2093
avg: 72.3922



<https://www.kamailio.org/wiki/kemi/performance-tests/5.2.x>

KEMI Performances For v5.2

- **Remarks**

- the execution of **native scripting** routing blocks and **Lua scripting** functions **takes more or less same time**
- sometimes is native scripting slightly faster, other times is Lua scripting slightly faster
- the **average** (for both native and Lua scripts) is in the range of **60 to 80 microseconds** (1 / 1 000 000 of a second)
- the minimum reflects more what it takes for the first REGISTER request without authorization header
 - which is **quickly challenged** with 401 response
- the maximum reflect the processing of the REGISTER request **with valid authorization** header
 - which does **a query to database** to fetch the password
 - as well as **store/update the record** in the location hash table (writing to database on timer - usrlc db_mode 2).
- there are variable number of **retransmissions**, being the reason for having cnt higher than 60000
 - a matter of sipp being able to match the response with the request at this high pace of stress testing
 - **4 000 registrations per second**, with a limit of **10 000 transactions at the same time**
 - stopping after **20 000 sent messages**
- note that each registration is **challenged for authentication**, so it is resent with authorization header
- the execution times for the **Python script** were in the **same range** as for native and Lua routing scripts



<https://www.kamailio.org/wiki/kemi/performance-tests/5.2.x>

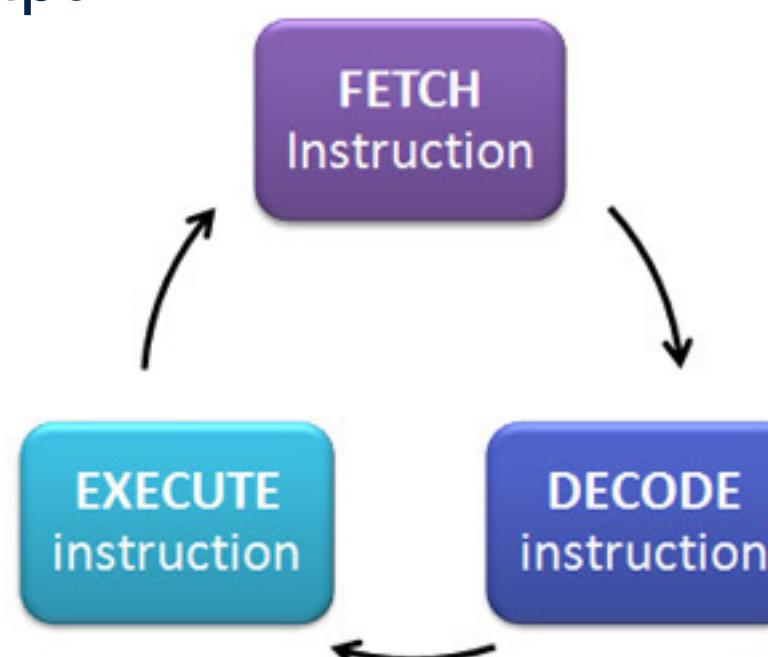


Optimize For Performance



Executing Module Functions

- No penalty for looking up what function to execute
 - Mapping the functions exports via a static array
 - Exported function to KEMI script has the index in the array of the functions to execute, which was set when loading the modules
- No performance difference for vast majority of them
 - Native execution process
 - Evaluate the parameters via fixup callbacks to int or string (replace variables with their value)
 - Execute the function exported to KEMI
 - KEMI execution process
 - Evaluation of parameters is done in KEMI script
 - Execute the function exported to KEMI



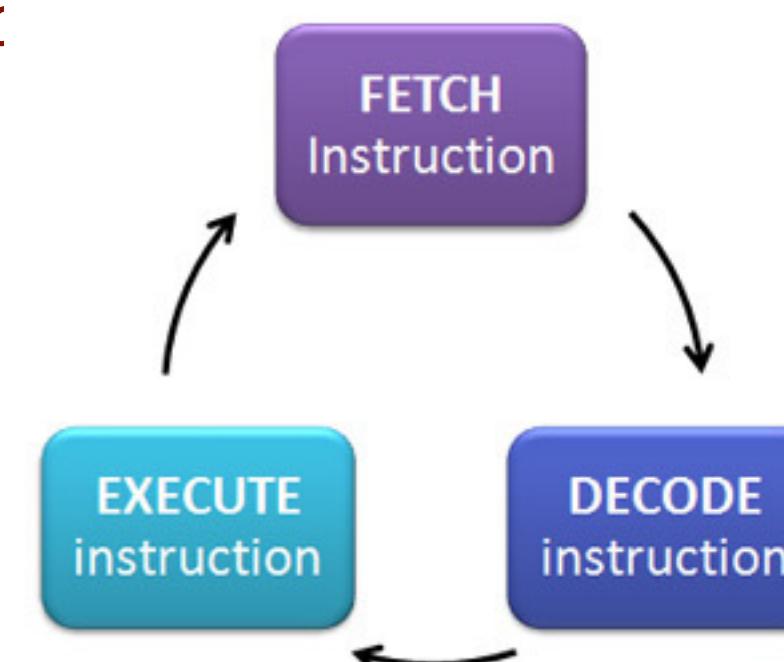
To Be Exported To Lua	Mapped When Loading Modules
<code>sr_kemi_lua_exec_func_0</code>	<code>sr_kemi_lua_export_t(t_relay)</code>
...	...
<code>sr_kemi_lua_exec_func_100</code>	<code>sr_kemi_lua_export_t(sl_send_reply)</code>
...	...
null	null
...	...

Executing Module Functions

– What is different

- Parameters that are regular expressions are parsed every time
 - Done also for native scripting when they contain variables
 - Applies to matching or substitution functions
 - Solutions:
 - Use regexp matching functions from that scripting language
 - Use dialplan module that loads regexp matching and substitution rules from database and caches them in memory
 - Use string based replacement functions
 - *Future plan:* analyze if some sort of caching can help

Note: it is not a big difference in execution time, all the operations are done in memory, so this is not a big concern in terms of performance



To Be Exported To Lua	Mapped When Loading Modules
<code>sr_kemi_lua_exec_func_0</code>	<code>sr_kemi_lua_export_t(t_relay)</code>
...	...
<code>sr_kemi_lua_exec_func_100</code>	<code>sr_kemi_lua_export_t(sl_send_reply)</code>
...	...
null	null
...	...

Variables

- Store value in a local KEMI script variable
 - Use KSR.kx exported by kemix module with functions that are equivalent of getting some (common) pseudo-variables
 - Get the value of Kamailio's (pseudo-)variable and store it locally
 - The scripting languages highly optimize the lookup of their native variables
 - Local variables in KEMI scripts are found faster even than pseudo-variables in native kamailio.cfg routing blocks
 - Lua precompiles in memory to native object code
 - Solutions:
 - Use regexp matching functions from that scripting language
- Be careful with:
 - Pseudo-variables that can change their value after executing some f
For example: \$ru changes after a successful lookup("location")
 - Do not define many (pseudo-)variable names (more to follow)



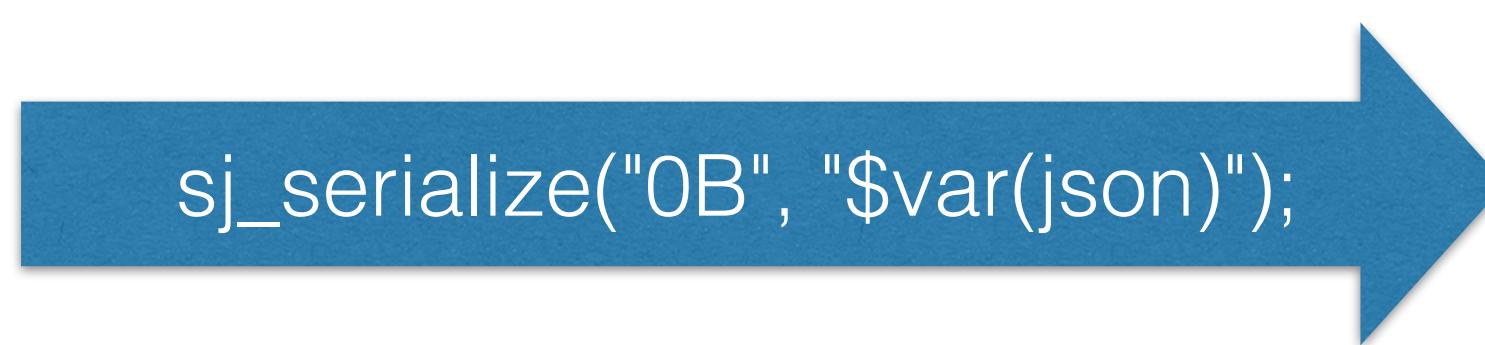
```
if KSR.corex.has_user_agent() then
    local K_ua = KSR.pv.get("$ua");
    if string.find(K_ua, "friendly-scanner")
        or string.find(K_ua, "sipcli") then
        KSR.sl.sl_send_reply(200, "OK");
        KSR.x.exit();
    end
end
```

SIPJSON Module

- Creates a JSON document with attributes from SIP message
 - Some KEMI scripting languages converts easily JSON to object or hash table
 - e.g., JavaScript, Lua, Python
 - Attribute names try to follow the pseudo-variable names

```
OPTIONS sip:alice@127.0.0.1 SIP/2.0
Via: SIP/2.0/UDP 127.0.0.1:65502;branch=z9hG4bK.79e78613;rport;alias
From: sip:sipsak@172.17.0.2;tag=74806173
To: sip:alice@127.0.0.1
Call-ID: 482215126@172.17.0.2
CSeq: 1 OPTIONS
Contact: sip:sipsak@127.0.0.1:65502
Content-Length: 0
Max-Forwards: 70
User-Agent: sipsak 0.9.7pre
Accept: text/plain
```

sj_serialize("OB", "\$var(json)");



```
{  
    "mt":2,  
    "rm":"OPTIONS",  
    "pr":"udp",  
    "si":"127.0.0.1",  
    "sp":36747,  
    "Ri":"127.0.0.1",  
    "Rp":5060,  
    "ru":"sip:alice@127.0.0.1",  
    "rU":"alice",  
    "rd":"127.0.0.1",  
    "rp":0,  
    "fU":"sipsak",  
    "fd":"172.17.0.2",  
    "ua":"sipsak 0.9.7pre",  
    "ci":"482215126@172.17.0.2",  
    "rb":""  
}
```

THE POWER OF
{JSON}

SIPJSON Module

```
$ sudo luarocks install lua-cjson
```



```
local json = require("cjson")
KSR.sipjson.sj_serialize("0B", "$var(json)")
local KV = json.decode(KSR.pv.get("var(json)"))
if KV["rm"] == "INVITE" then
    ...
end
```

Variables

- Have preference for dedicated functions with specific variables
 - htable module offers functions to:
 - set values and expire interval
 - remove items by name, with regexp or other matching rules
 - check if an item exists
 - pv module offers functions to:
 - do some XAVP operations (new - to be extended)
 - evaluate a string with pseudo-variables
 - corex module offers functions to:
 - test R-URI username and User-Agent header
 - set send socket
 - **kemix** module offers functions to get values of some variables
 - ...



```
KSR.pv.sets("$sht(t=>x)", "kamailio");  
KSR.pv.seti("$shtex(t=>x)", 300);
```

– is equivalent with:

```
KSRhtable.sht_setxs("t", "x", "kamailio", 300);
```

Variables

- Avoid declaring many (pseudo-)variables with dynamic name

- When a set or get operation is done over a pseudo-variable, it is automatically declared in Kamailio code — a structure is created for it
- If the same name is used, same associate structure is used
 - example: \$sht(t=>a), \$sht(t=>b) and \$sht(t=>c) are three variables, each with its own structure
 - example: \$var(x) = "a"; \$sht(t=>\$var(x)); \$var(x) = "b"; \$sht(t=>\$var(x)); \$var(x) = "c"; \$sht(t=>\$var(x)); — in this case only two variables are declared, respectively: \$var(x) and \$sht(t=>\$var(x))



```
// do not use  
KSR.pv.seti("$sht(ipban=>" + KSR.pv.gete("$si") + ")", 1);  
  
// use instead  
KSR.pv.seti("$sht(ipban=>$si)", 1);
```

- core parameters to control the pv cache size

- ***pv_cache_limit*** - the limit how many pv declarations in the cache after which an action is taken. Default value is 2048
- ***pv_cache_action*** - specify what action to be done when the size of pv cache is exceeded. If 0, print an warning log message when the limit is exceeded. If 1, warning log messages is printed and the cache systems tries to drop a \$sht(...) declaration. Default is 0

Core Keywords

- Those special tokens in kamailio.cfg routing blocks
 - uri, from_uri, to_uri, method, myself, af, UDP, TCP, TLS, src_ip, ...
 - The usual conditions can be done with functions
 - Variants to match methods and myself conditions

```
KSR.is_myself_ruri()  
KSR.is_myself_furi()  
KSR.is_myself_turi()  
KSR.is_myself_srcip()
```

– better than

```
KSR.is_myself(KSR.pv.get("$ru"))
```

```
if KSR.pv.get("$rm") == "CANCEL" then
```

– is equivalent with:

```
if KSR.is_CANCEL() then
```

```
if string.find("INVITE,BYE,SUBSCRIBE,UPDATE", KSR.pv.get("$rm")) then
```

– is equivalent with:

```
if KSR.is_method_in("IBSU") then
```



Concerned About Performances?



https://en.wikipedia.org/wiki/List_of_applications_using_Lua

It Is Not Only About Performances

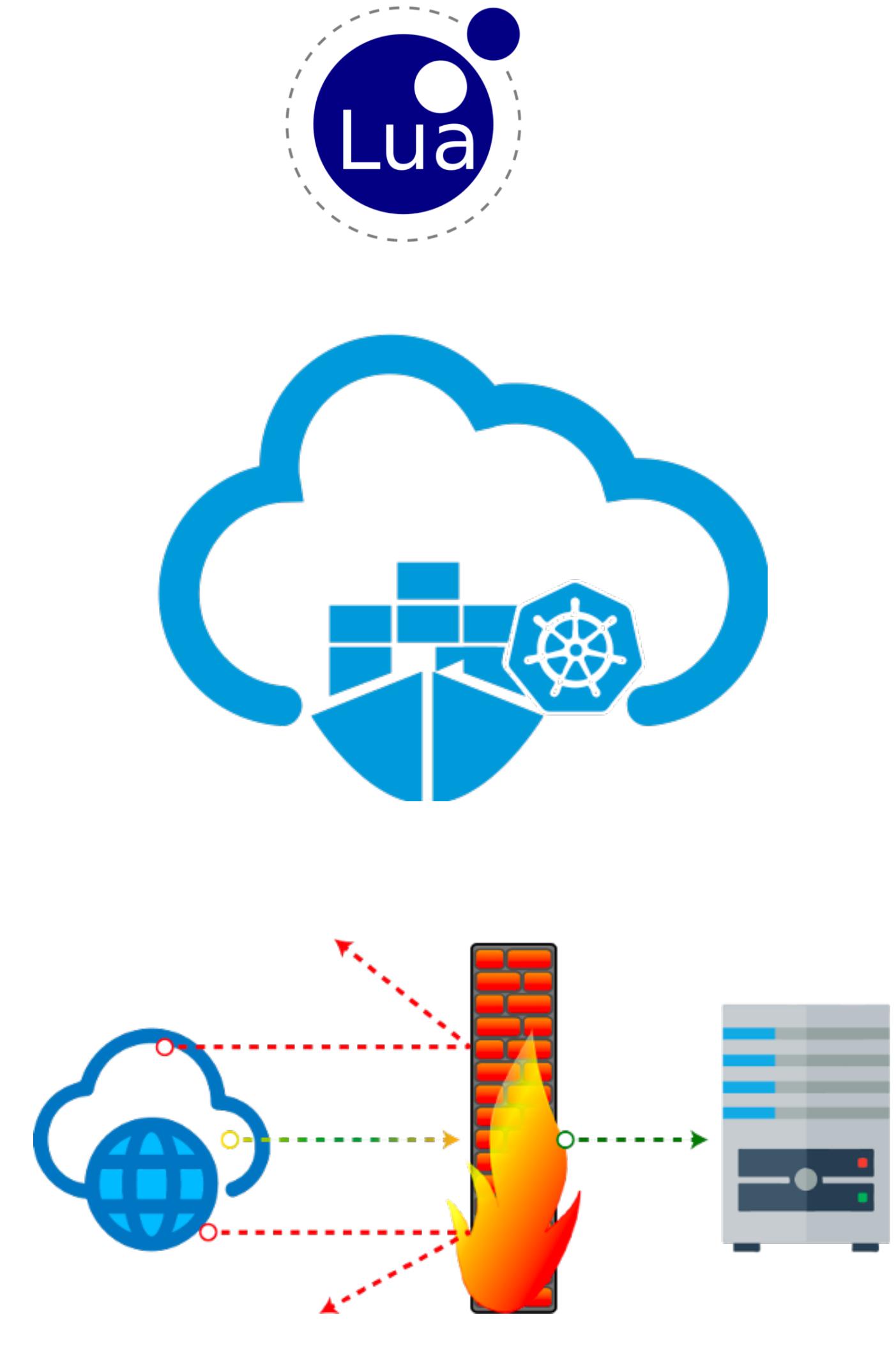
- Plenty of additional benefits
 - Documentation of the scripting language
 - More data types and operations, coherence in behaviour
 - Knowledge base and examples to learn the scripting language
 - Extensive testing of the scripting language
 - Development of the scripting language
 - Large set of extensions
 - Debugging tools
 - Specialized editors
 - Additional scalability mechanisms (threads, coroutines, ...)
 - **Reloading without restart**



Source IP Matching

- replace the permissions module -

```
ALLOWADDR={  
    "127.0.0.1/32",  
    "10.0.0.1/24"  
};  
  
-- match source ip against ALLOWADDR list  
function ksr_is_src_trusted()  
    local srcaddr = KSR.kx.get_srcip();  
    for idx, val in pairs(ALLOWADDR) do  
        if KSR.ipops.ip_is_in_subnet(srcaddr, val) > 0 then  
            return true;  
        end  
    end  
    return false;  
end  
  
-- IP authorization and user authentication  
function ksr_route_auth()  
    -- from trusted list of addresses  
    if ksr_is_src_trusted() then  
        return 1;  
    end  
    ... -- user authentication, etc...  
end
```



Just For Fun



CAUTION
FUN
AHEAD

```
import sys
import KSR as KSR
from art import *
from os import system

def mod_init():
    KSR.info("===== from Python mod init\n");
    # dumpObj(KSR);
    return kamailio();

class kamailio:
    def __init__(self):
        KSR.info('===== kamailio.__init__\n');

    def child_init(self, rank):
        KSR.info('===== kamailio.child_init(%d)\n' % rank);
        return 0;

    def ksr_request_route(self, msg):
        KSR.sl.sl_send_reply(200, "OK");
        Art=text2art("Hello " + KSR.kx.get_ruser() + "!")
        KSR.info("\n" + Art)
        system("say Hello " + KSR.kx.get_ruser() + "!")
        return 1;
```

```

import sys
import KSR as KSR
from http.server import HTTPServer, BaseHTTPRequestHandler

class SimpleHTTPRequestHandler(BaseHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200)
        self.end_headers()
        KSR.jsonrpcs.execx('{"jsonrpc": "2.0", "method": "stats.fetch", "params": ["all"], "id": 1}');
        self.wfile.write(b'Kamailio Statistics:\n' + KSR.pv.gete("$jsonrpl(body)").encode())

    def mod_init():
        return kamailio();

class kamailio:
    def __init__(self):
        KSR.info('===== kamailio.__init__\n');

    def child_init(self, rank):
        KSR.info('===== kamailio.child_init(%d)\n' % rank);
        return 0;

    def ksr_request_route(self, msg):
        KSR.info('===== kamailio sip routing block\n');
        KSR.sl.sl_send_reply(200, "OK");
        return 1;

    def ksr_webserver(self, msg, evdata):
        KSR.info('===== evroute start\n');
        httpd = HTTPServer(('localhost', 8000), SimpleHTTPRequestHandler)
        httpd.serve_forever()

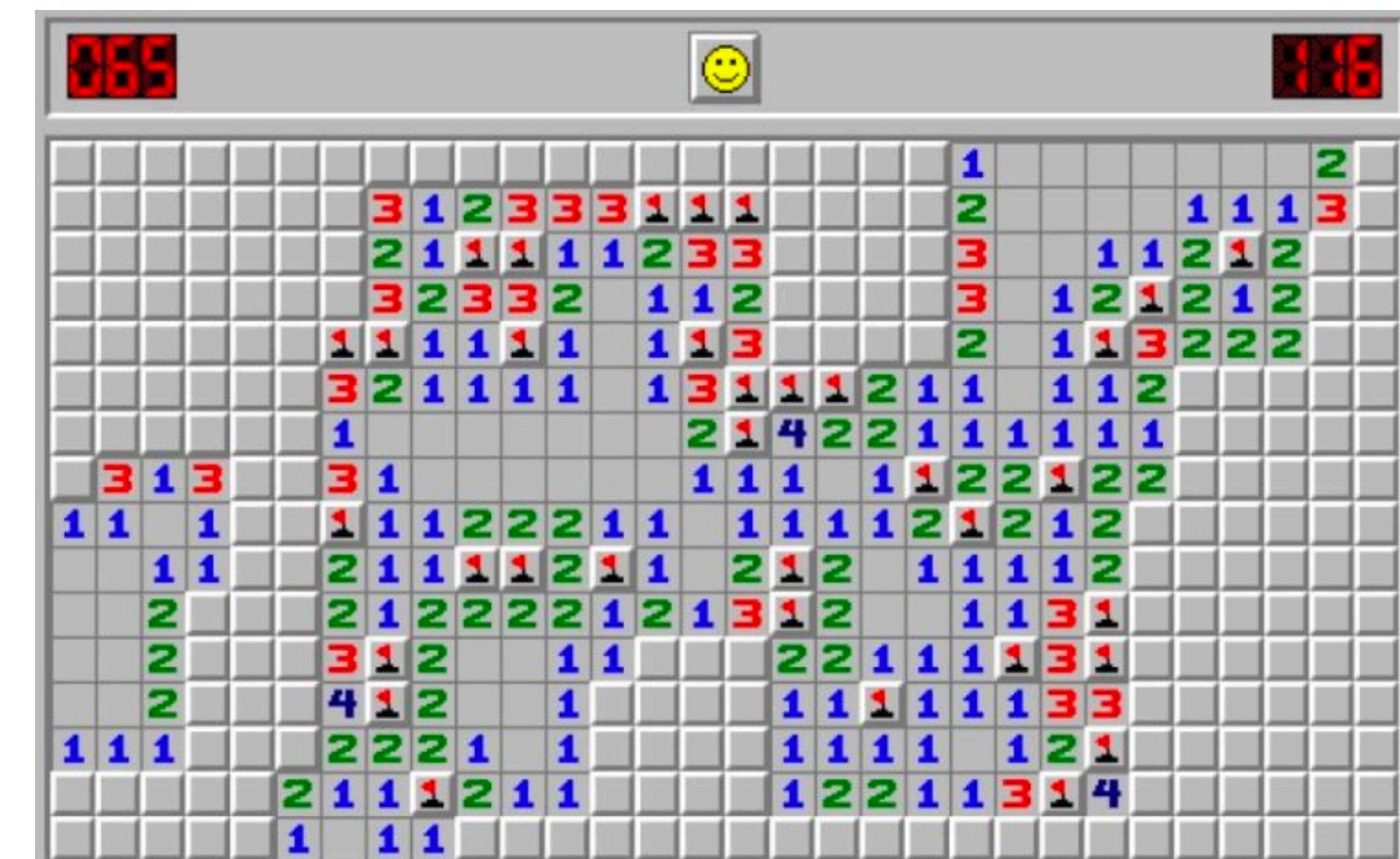
```



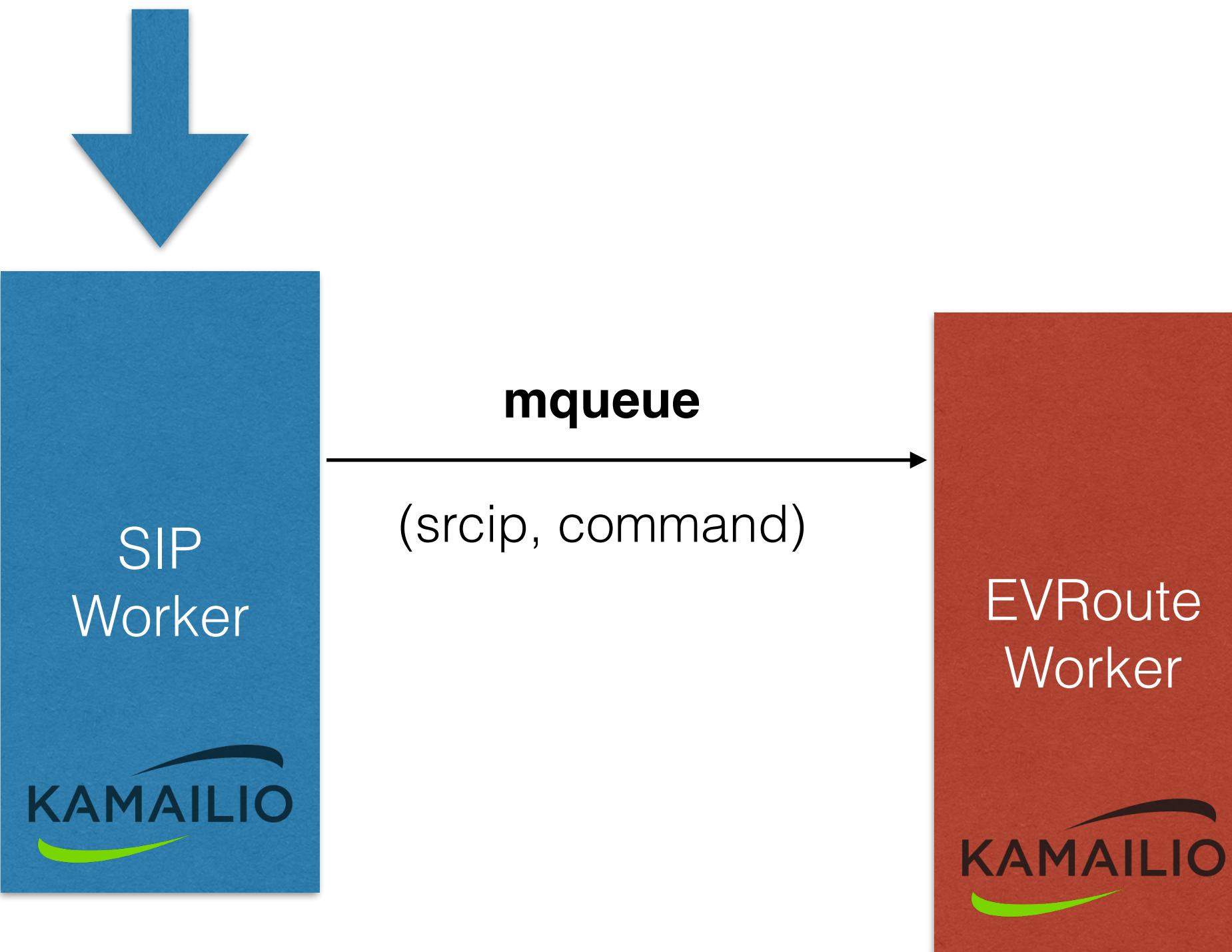


minesweeper

DANGEROUS OR NOT!!!



OPTIONS sip:**e-2-5**@127.0.0.1 SIP/2.0



Command: C-X-Y

E - explore

D - defuse

X - row

Y - column

* commands: e-x-y - explore; commands: d-x-y - defuse
* map: H - unexplored; W - defused; X - buuum!

①	H	H	H	H	H	H
②	H	H	H	H	H	H
③	H	H	H	H	H	H
④	H	H	H	H	H	H
⑤	H	H	H	H	H	H

① ② ③ ④ ⑤						

sipsak -vvv -s sip:**e-4-5**@127.0.0.1
curl http://127.0.0.1:5060/**e-1-2**
wget -O - http://127.0.0.1:5060/**e-1-2**

Use of
<https://github.com/fshahinfar1/MineSweeper>



THANK YOU!

Yours Truly,
The SRM Assistant!



THANK YOU!



8th Edition - Spring 2020
www.kamailioworld.com

Daniel-Constantin Mierla
Co-Founder Kamailio Project
[@miconda](https://twitter.com/miconda)
asipto.com