OpenSIPS - an event-driven SIP routing engine

FOSDEM'17

Liviu Chircu

- 4th Feb 2017 -

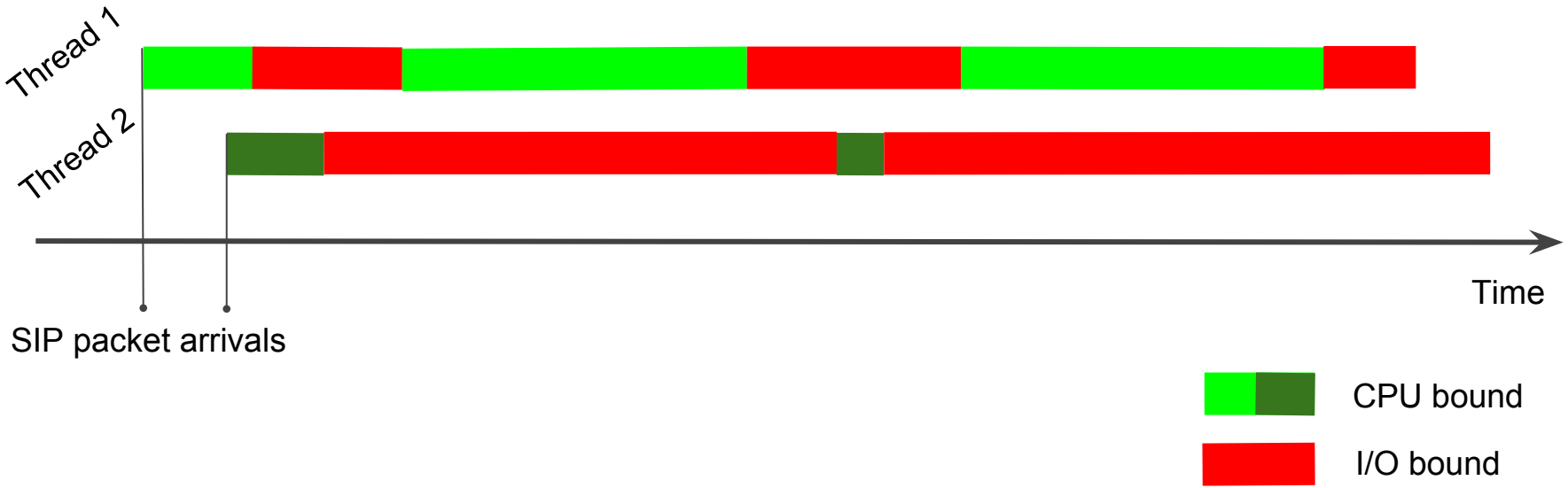
Outline

- Architecture timeline
- Event subscribe-notify
- Usage scenarios
- OpenSIPS scripts
- Conclusions

Architecture timeline

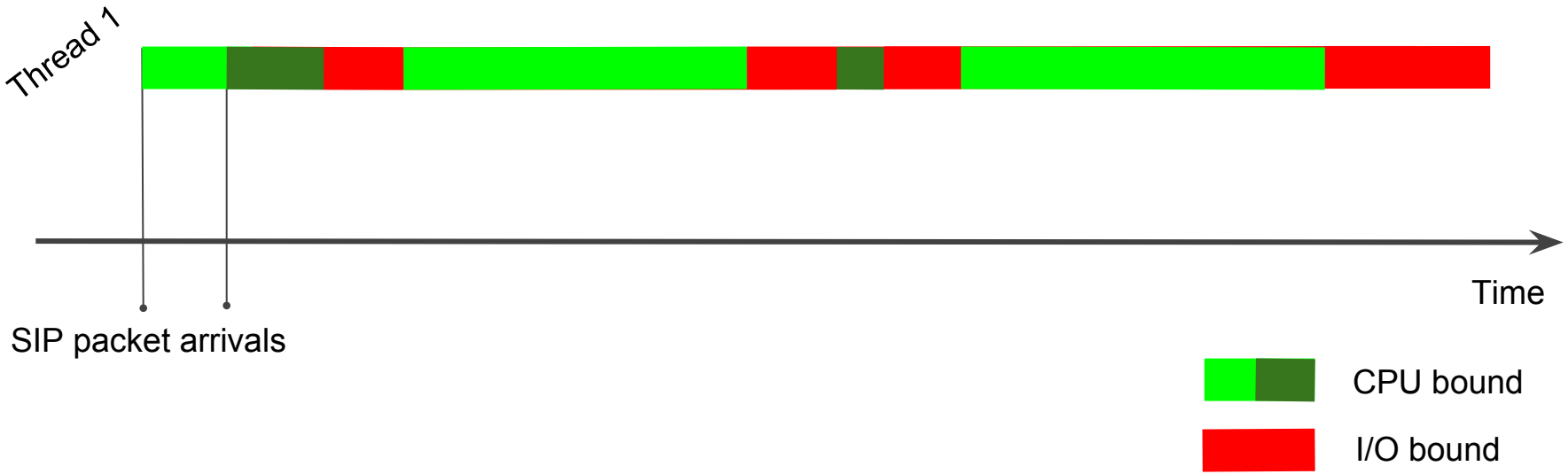
Step 1: “Linear” architecture

OpenSIPS 1.X



Step 2: “Async” architecture

OpenSIPS 2.1, 2.2



Limitation: Processing is still linear!

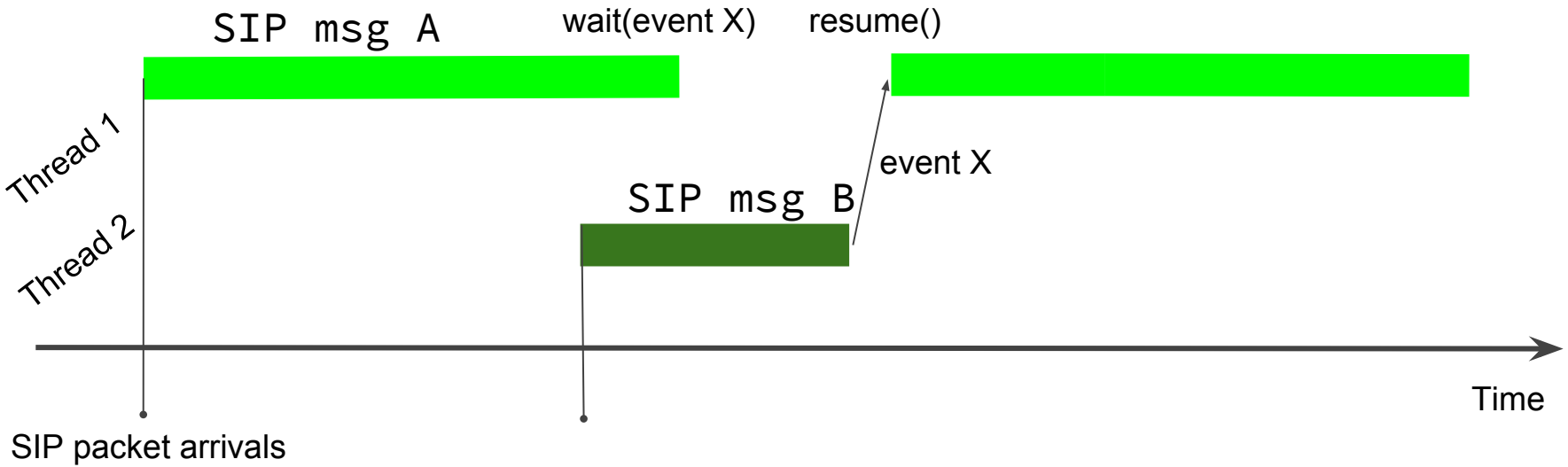
Advanced SIP scenarios:

- Push Notifications
- FreeSWITCH ESL Events (e.g. DTMF)
- Call Pick-up

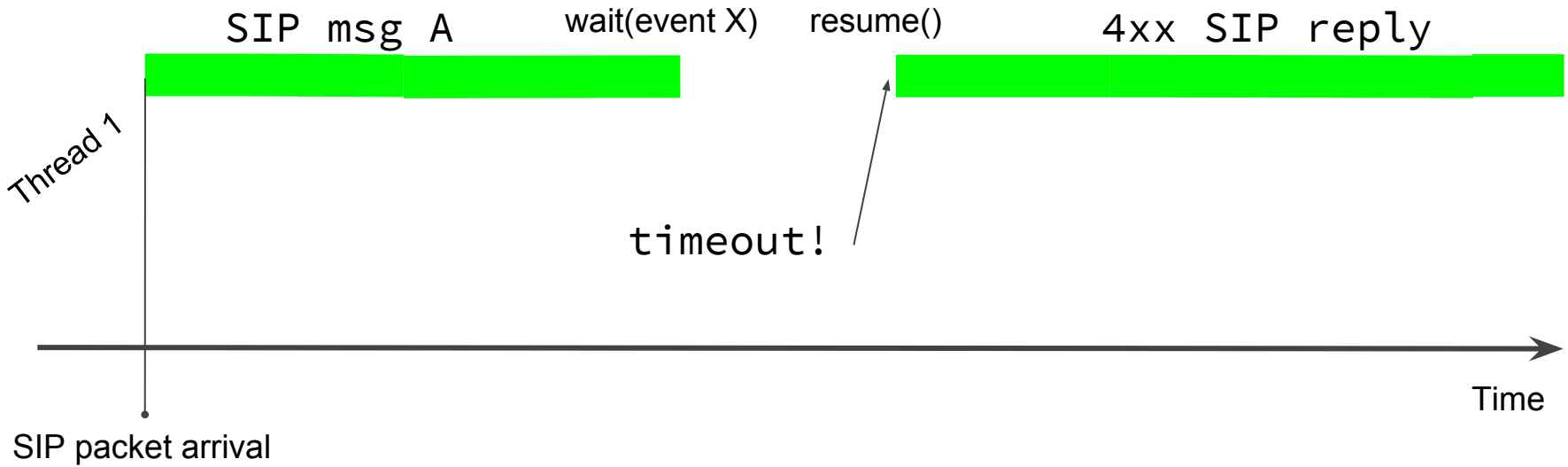
What's missing:

- Communication & data exchange between different processing contexts

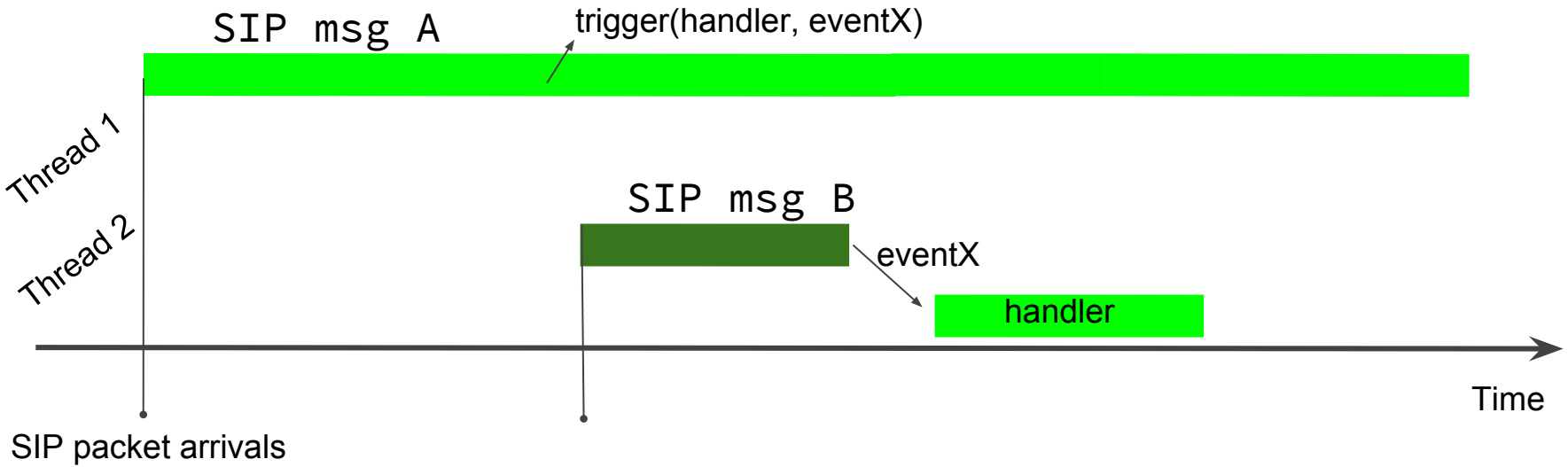
Step 3: “Event-driven” architecture



Step 3: “Event-driven” architecture



Step 3: “Event-driven” architecture



Event subscribe/notify

Event Definition

- Triggered by actions / data processing during runtime
- Events hold key/value attributes
- OpenSIPS has a list of predefined events

Subscribe

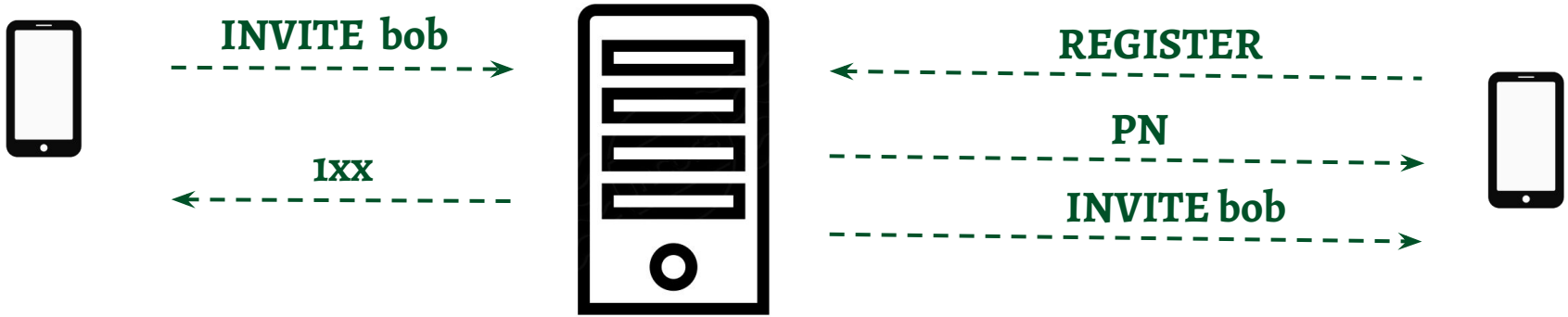
- interested OpenSIPS workers subscribe to events
- event subscriptions may contain filtering attributes

Notify

- Notification == event
- Events are generated during runtime
- They are dispatched to all relevant subscribers
- Events are parametrized (e.g. DTMF digit, REG Contact)

Usage scenarios

Push Notifications



Push Notifications

Current way

1. incoming call for “bob”
2. send PN to “bob”’s mobile device
3. async sleep (N)
4. call(“bob”) if registered(“bob”) else goto 3.

Push Notifications

Current way (limitations)

1. performance killer

2. inflexible, cannot handle complex scenarios

- parallel forking (desk + mobile devices)
- multiple gateways

Push Notifications

In 2.3:

1. incoming call for “bob”
2. subscribe(“REGISTER”, “aor=bob”, “reg_handler”)
3. send PN to “bob”’s mobile device
4. fork calls to existing registrations

…

```
route [reg_handler] { fork_call(“$event(contact)”); }
```

DTMF-based fax/voicemail detection

In 2.3:

1. incoming call for “bob”
2. subscribe(“DTMF”, “callid=$ci”, “dtmf_handler”)
3. send call to “bob”

...

```
route [dtmf_handler] { hangup() if $event(digit) != 2 }
```

OpenSIPS Script

Push Notifications

```
subscribe("REGISTER", "aor=bob", "reg_handler");

route(SEND_APN);

if (lookup("location"))

    t_relay();

halt();
```

Push Notifications

```
route [SEND_APN] {
    rest_append_hf("Authorization: key=CONSOLE_API_KEY");
    rest_append_hf("Content-Type: \"application/json\"");
    rest_post("https://android.googleapis.com/gcm/send",
            "{ \"data\" : {\"foo\": \"bar\"},
        \"registration_ids\":[\"REGISTRATION_ID\"] }"...);
}
```

Push Notifications

```
route [reg_handler] {
    route(CHECK_IF_MOBILE, "$event(contact)");
    t_relay();
}
```

Conclusions

Conclusions

Event-driven approach:

- powerful & easy to use
- complex scenarios with simple script
- lightning fast!

OpenSIPS 2.3 - “integration”

- SIP capturing - Homer/SIPCapture
- billing - CGRateS
- software PBX - FreeSWITCH
- middleware - RabbitMQ

Take-Away Message

Under development!

- Liviu Chircu
  - OpenSIPS Project: www.opensips.org
  - liviu@opensips.org