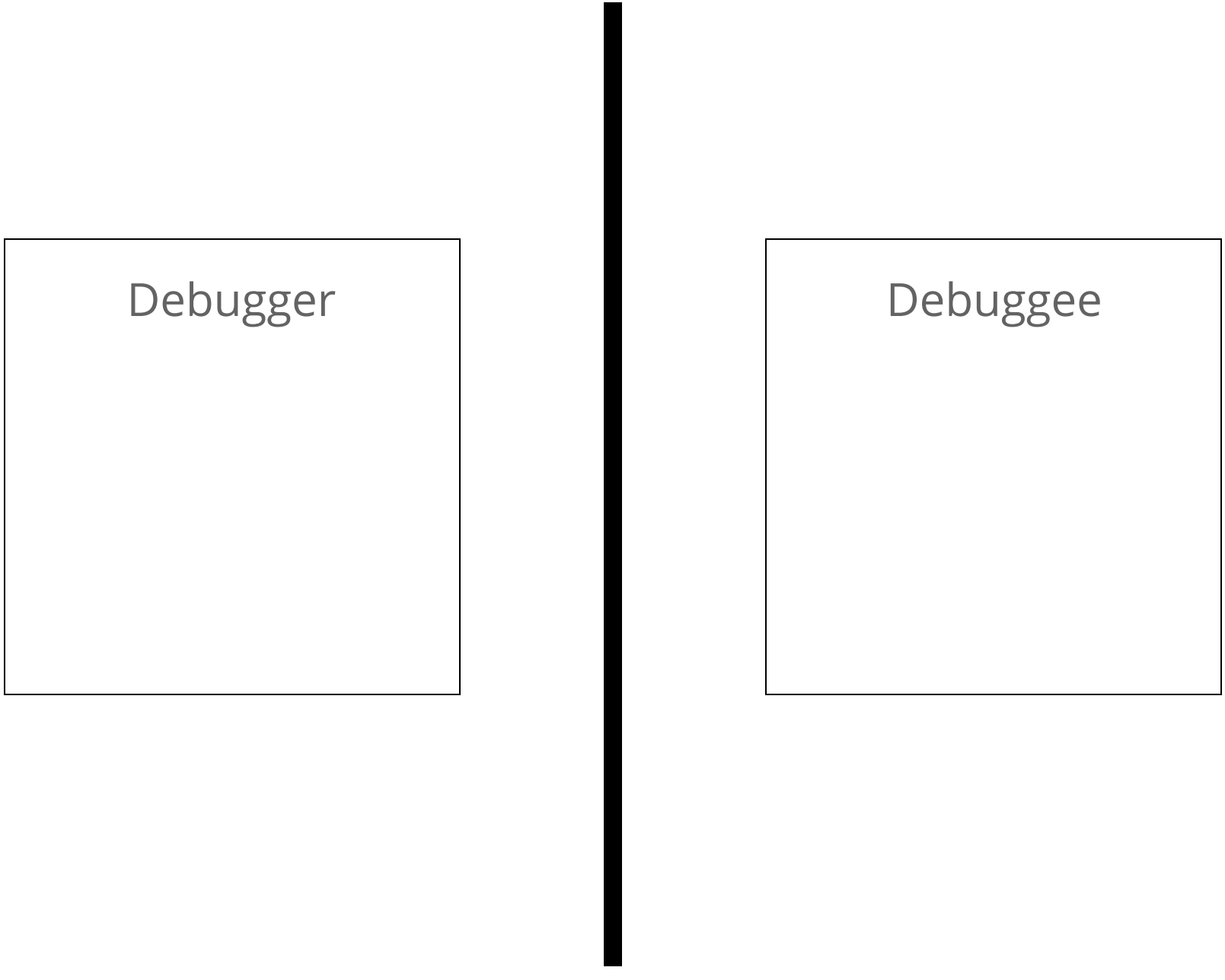
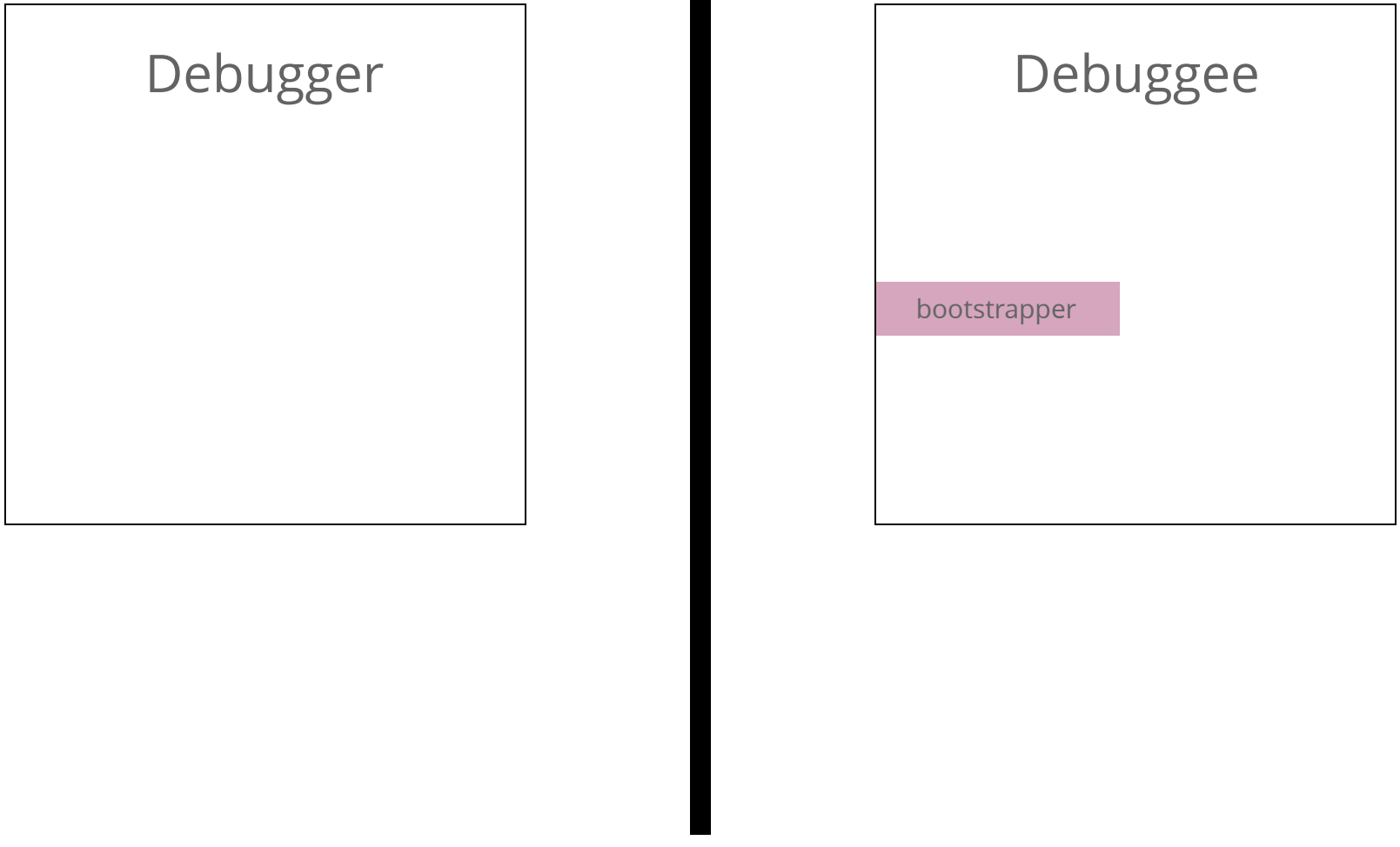
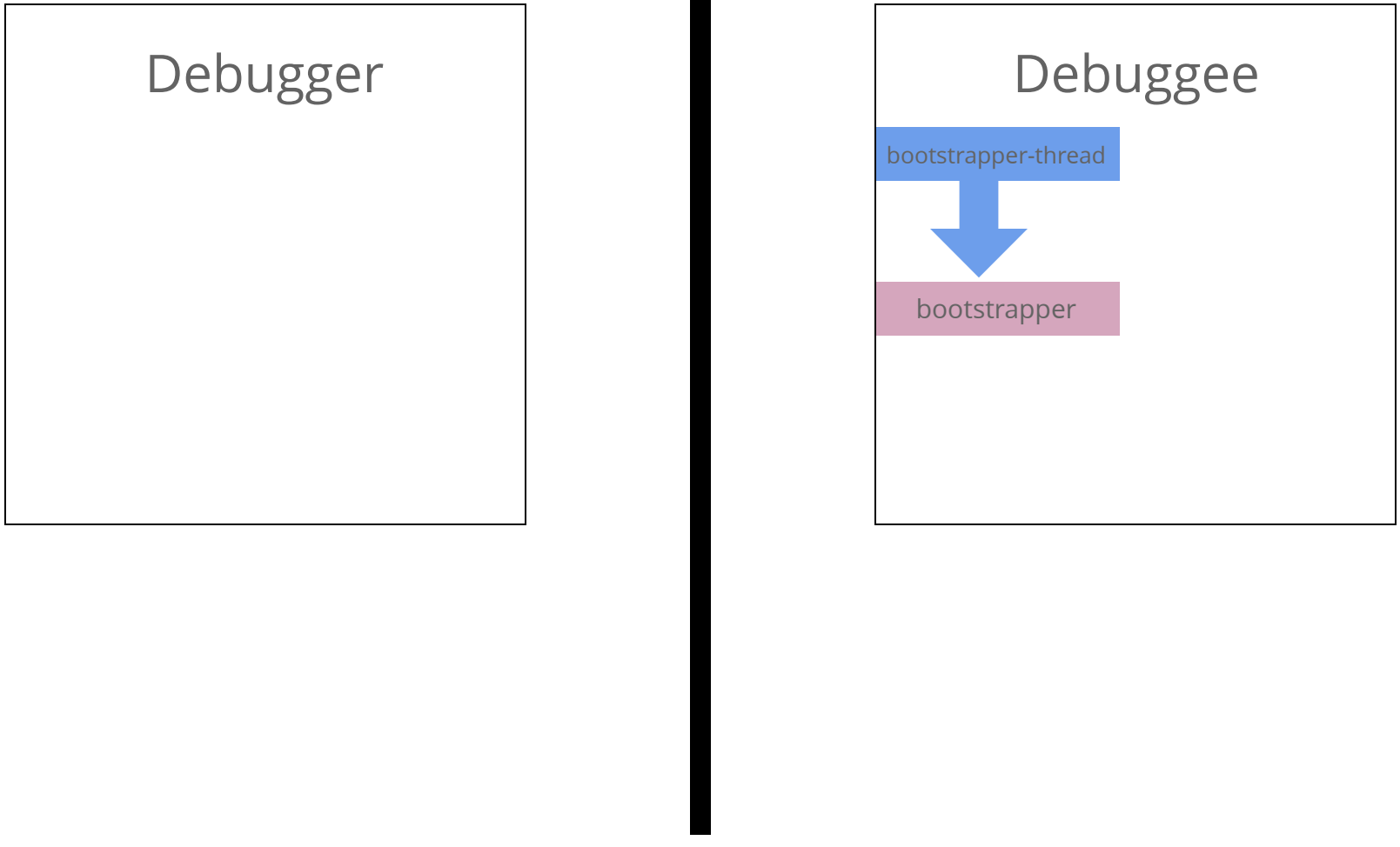
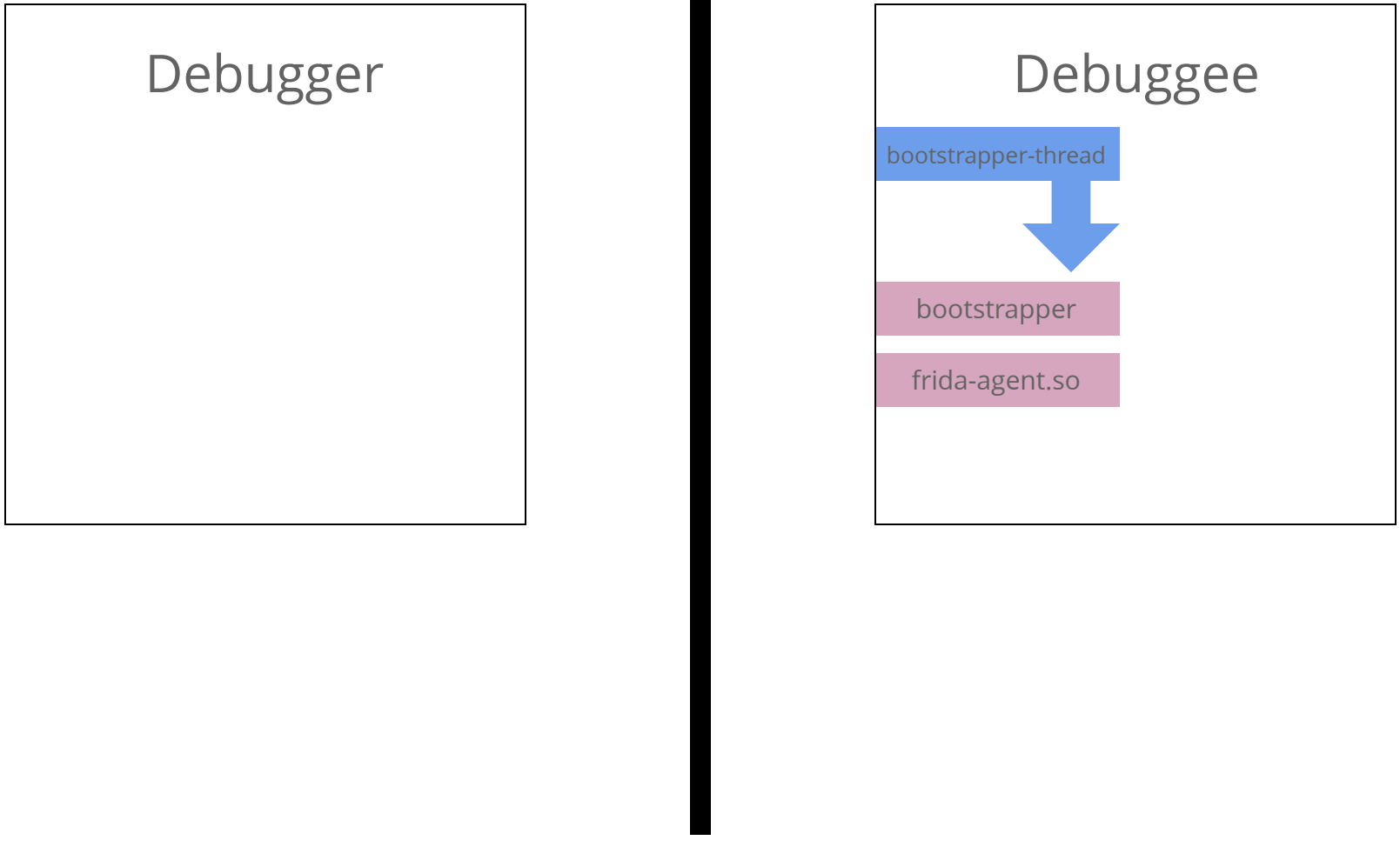
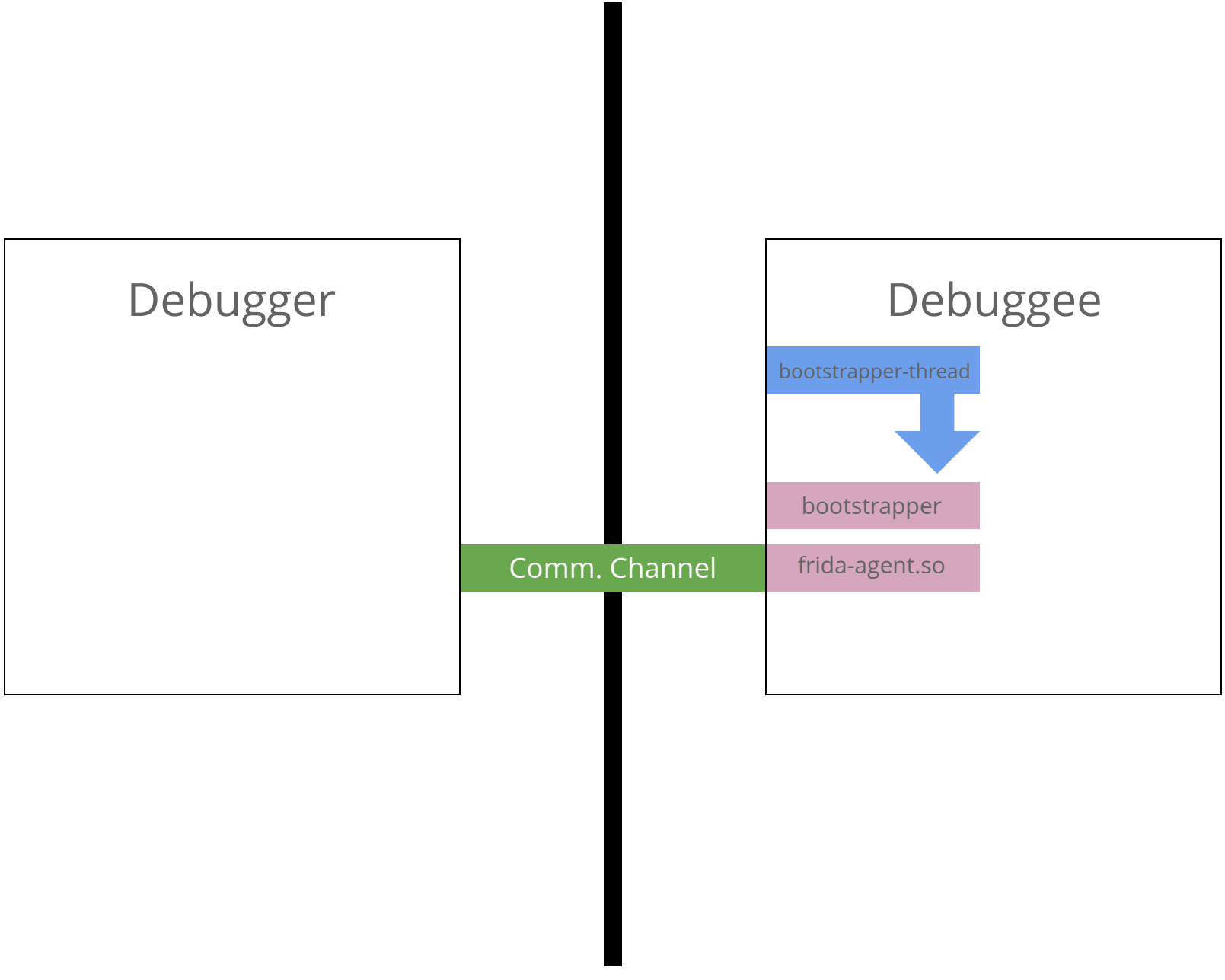
Dynamic Reverse Engineering with Frida

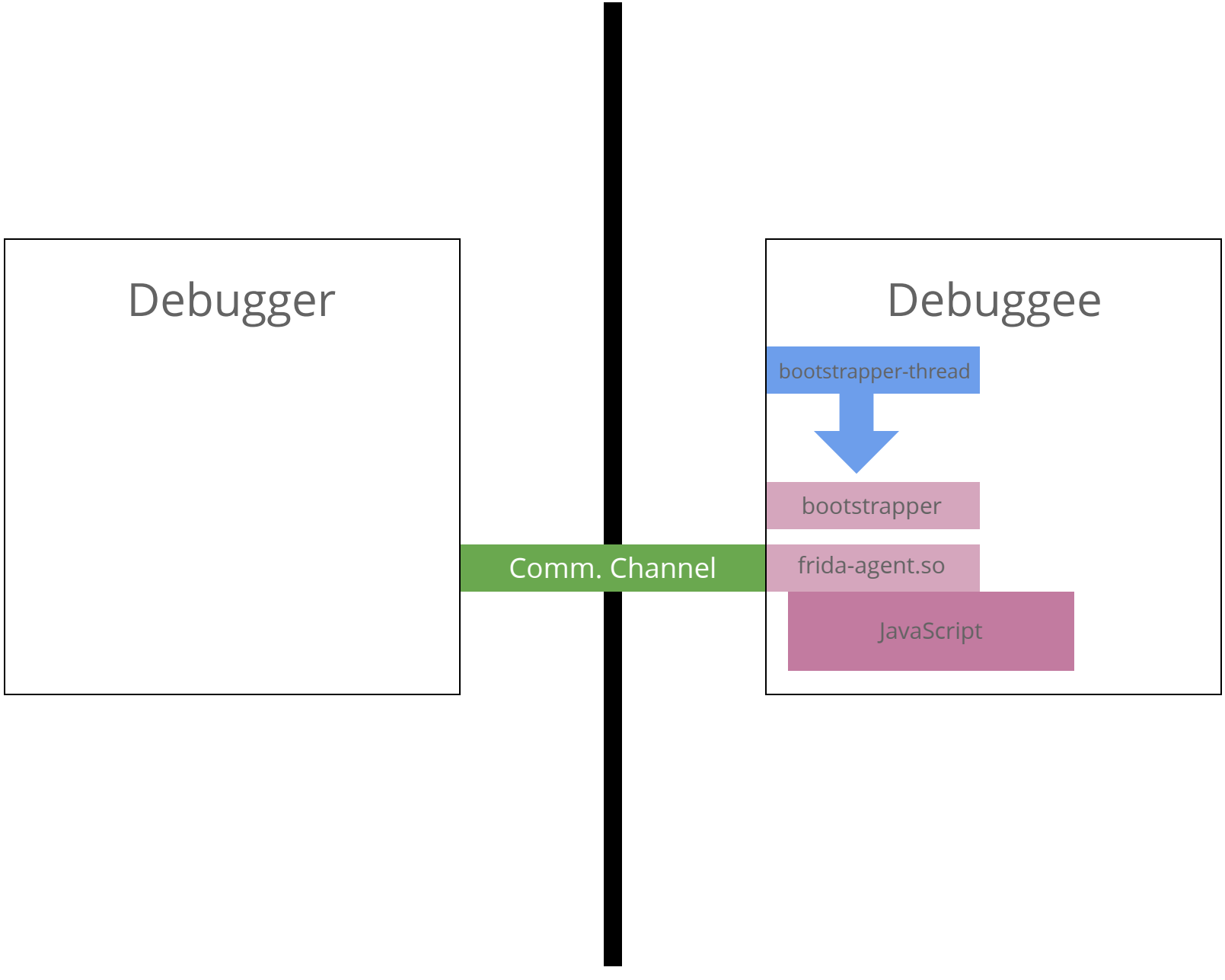

Debugger
Debuggee


Debugger
Debuggee
bootstrapper


Debugger
Debuggee
bootstrapper-thread
bootstrapper

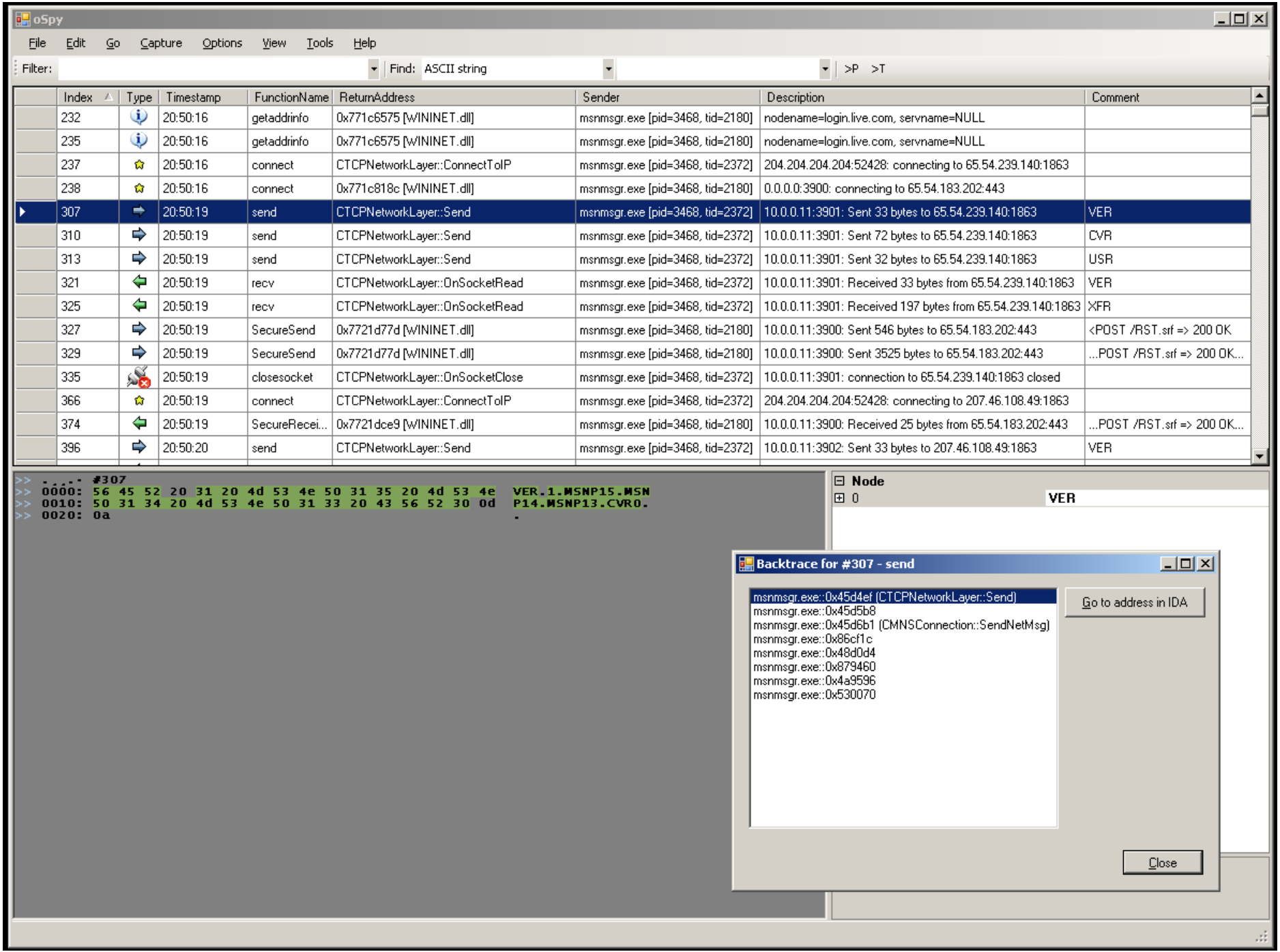

Debugger
Debuggee
bootstrapper-thread
bootstrapper
frida-agent.so

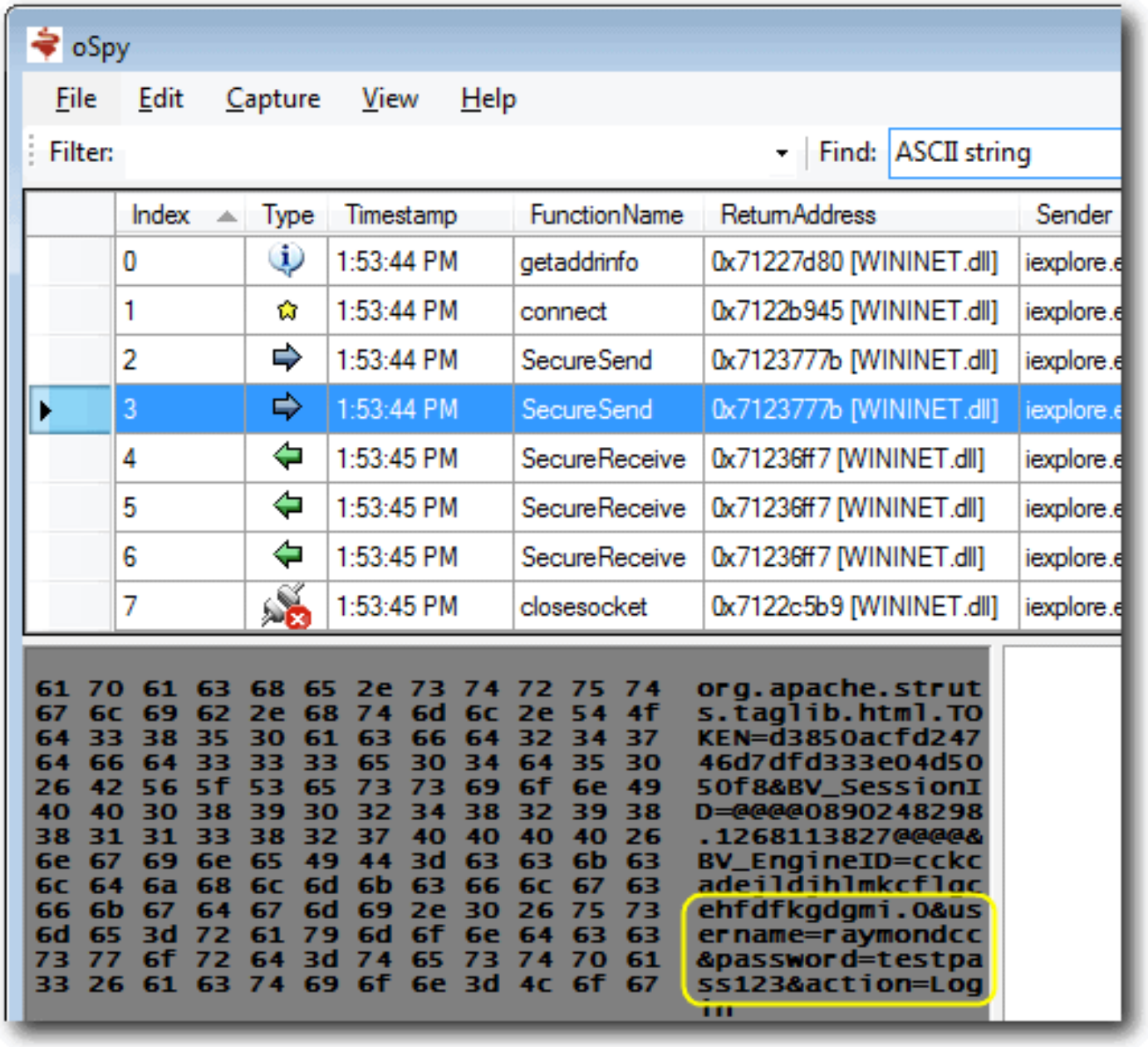

Debugger
Debuggee
bootstrapper-thread
bootstrapper
Comm. Channel
frida-agent.so


Debugger
Debuggee
bootstrapper-thread
bootstrapper
Comm. Channel
frida-agent.so
JavaScript

Motivation

- Existing tools often not a good fit for the task at hand
- Creating a new tool usually takes too much effort
- Short feedback loop: reversing is an iterative process
- Use one toolkit for multi-platform instrumentation
- Future remake of oSpy (see below)


oSpy
File Edit Go Capture Options View Tools Help
Filter:
Find: ASCII string
>P >T
Index Type Timestamp FunctionName ReturnAddress Sender Description Comment
232 20:50:16 getaddrinfo 0x771c6575 [WININET.dll] msnmsgr.exe [pid=3468, tid=2180] nodename=login.live.com, servname=NULL
235 20:50:16 getaddrinfo 0x771c6575 [WININET.dll] msnmsgr.exe [pid=3468, tid=2180] nodename=login.live.com, servname=NULL
237 20:50:16 connect CTCPNetworkLayer::ConnectToIP msnmsgr.exe [pid=3468, tid=2372] 204.204.204.204:52428: connecting to 65.54.239.140:1863
238 20:50:16 connect 0x771c818c [WININET.dll] msnmsgr.exe [pid=3468, tid=2180] 0.0.0.0:3900: connecting to 65.54.183.202:443
307 20:50:19 send CTCPNetworkLayer::Send msnmsgr.exe [pid=3468, tid=2372] 10.0.0.11:3901: Sent 33 bytes to 65.54.239.140:1863 VER
310 20:50:19 send CTCPNetworkLayer::Send msnmsgr.exe [pid=3468, tid=2372] 10.0.0.11:3901: Sent 72 bytes to 65.54.239.140:1863 CVR
313 20:50:19 send CTCPNetworkLayer::Send msnmsgr.exe [pid=3468, tid=2372] 10.0.0.11:3901: Sent 32 bytes to 65.54.239.140:1863 USR
321 20:50:19 recv CTCPNetworkLayer::OnSocketRead msnmsgr.exe [pid=3468, tid=2372] 10.0.0.11:3901: Received 33 bytes from 65.54.239.140:1863 VER
325 20:50:19 recv CTCPNetworkLayer::OnSocketRead msnmsgr.exe [pid=3468, tid=2372] 10.0.0.11:3901: Received 197 bytes from 65.54.239.140:1863 XFR
327 20:50:19 SecureSend 0x7721d77d [WININET.dll] msnmsgr.exe [pid=3468, tid=2180] 10.0.0.11:3900: Sent 546 bytes to 65.54.183.202:443 <POST /RST.srf => 200 OK
329 20:50:19 SecureSend 0x7721d77d [WININET.dll] msnmsgr.exe [pid=3468, tid=2180] 10.0.0.11:3900: Sent 3525 bytes to 65.54.183.202:443 ...POST /RST.srf => 200 OK...
335 20:50:19 closesocket CTCPNetworkLayer::OnSocketClose msnmsgr.exe [pid=3468, tid=2372] 10.0.0.11:3901: connection to 65.54.239.140:1863 closed
366 20:50:19 connect CTCPNetworkLayer::ConnectToIP msnmsgr.exe [pid=3468, tid=2372] 204.204.204.204:52428: connecting to 207.46.108.49:1863
374 20:50:19 SecureRecei... 0x7721dce9 [WININET.dll] msnmsgr.exe [pid=3468, tid=2180] 10.0.0.11:3900: Received 25 bytes from 65.54.183.202:443 ...POST /RST.srf => 200 OK...
396 20:50:20 send CTCPNetworkLayer::Send msnmsgr.exe [pid=3468, tid=2372] 10.0.0.11:3902: Sent 33 bytes to 207.46.108.49:1863 VER
>> ....: #307
>> 0000: 56 45 52 20 31 20 4d 53 4e 50 31 35 20 4d 53 4e VER.1.MSNP15.MSN
>> 0010: 50 31 34 20 4d 53 4e 50 31 33 20 43 56 52 30 0d P14.MSNP13.CVR0.
>> 0020: 0a .
Node
0 VER
Backtrace for #307 - send
msnmsgr.exe::0x45d4ef (CTCPNetworkLayer::Send)
msnmsgr.exe::0x45d5b8
msnmsgr.exe::0x45d6b1 (CMNSConnection::SendNetMsg)
msnmsgr.exe::0x86cf1c
msnmsgr.exe::0x48d0d4
msnmsgr.exe::0x879460
msnmsgr.exe::0x4a9596
msnmsgr.exe::0x530070
Go to address in IDA
Close


oSpy
File
Edit
Capture
View
Help
Filter:
Find:
ASCII string
Index
Type
Timestamp
FunctionName
ReturnAddress
Sender
0 1:53:44 PM getaddrinfo 0x71227d80 [WININET.dll] iexplore.e
1 1:53:44 PM connect 0x7122b945 [WININET.dll] iexplore.e
2 1:53:44 PM SecureSend 0x7123777b [WININET.dll] iexplore.e
3 1:53:44 PM SecureSend 0x7123777b [WININET.dll] iexplore.e
4 1:53:45 PM SecureReceive 0x71236ff7 [WININET.dll] iexplore.e
5 1:53:45 PM SecureReceive 0x71236ff7 [WININET.dll] iexplore.e
6 1:53:45 PM SecureReceive 0x71236ff7 [WININET.dll] iexplore.e
7 1:53:45 PM closesocket 0x7122c5b9 [WININET.dll] iexplore.e
61 70 61 63 68 65 2e 73 74 72 75 74 org.apache.strut
67 6c 69 62 2e 68 74 6d 6c 2e 54 4f s.taglib.html.TO
64 33 38 35 30 61 63 66 64 32 34 37 KEN=d3850acfd247
64 66 64 33 33 33 65 30 34 64 35 30 46d7dfd333e04d50
26 42 56 5f 53 65 73 73 69 6f 6e 49 50f8&BV_SessionI
40 40 30 38 39 30 32 34 38 32 39 38 D=@@@@0890248298
38 31 31 33 38 32 37 40 40 40 40 26 .1268113827@@@@&
6e 67 69 6e 65 49 44 3d 63 63 6b 63 BV_EngineID=cckc
6c 64 6a 68 6c 6d 6b 63 66 6c 67 63 adeildihlmkcflgc
66 6b 67 64 67 6d 69 2e 30 26 75 73 ehfdfkgdgmi.0&us
6d 65 3d 72 61 79 6d 6f 6e 64 63 63 ername=raymondcc
73 77 6f 72 64 3d 74 65 73 74 70 61 &password=testpa
33 26 61 63 74 69 6f 6e 3d 4c 6f 67 ss123&action=Log

Conversations

Visualization

10.0.0.11:1145 <-> 207.46.104.25:1863 | <UNKNOWN ENDPOINTS>

[14:20:40]

```
VER 1 MSNP15 MSNP14 MSNP13 CVR0
```

```
CVR 2 0x0409 winnt 5.1 i386 MSNMSGR 8.1.0178 msmsgs tryggve1@gmail.com
```

```
USR 3 SSO I tryggve1@gmail.com
```

[14:20:41]

```
VER 1 MSNP15 MSNP14 MSNP13 CVR0
```

```
CVR 2 8.1.0178 8.1.0178 8.0.0787 http://msgr.dlservice.microsoft.com/downl
```

```
GCF 0 3866

<Policies>
    <Policy type="SHIELDS">
        <config>
            <shield>
                <cli maj="7" min="0" minbld="0" maxbld="9999" deny=""/>
            </shield>
            <block>
                <hashes>
                </hashes>
                <regexp>
                    <imtext value="LipcLnBpZi4q"/>
                    <imtext value="LipcLnNjci4q"/>
                    <imtext value="LipncmV1cHBpY3R1cmVcLnBocC4q"/>
                    <imtext value="LipncmV1cGljdHVyZVwucGhwLio="/>
                    <imtext value="LipnYWxsZXJ5XC5waHAuKg=="/>
                    <imtext value="LipzdGFmZlwucGhwLio="/>
                    <imtext value="LipwaWNzXC5waHAuKg=="/>
                    <imtext value="Lipyb3R0ZWS0b21hdG91c1wud3MuKg=="/>
```

```
USR 3 SSO S MBI_KEY_OLD Bi3aMlrUpBoxHOPz30WhHLG0Qpd/G/dylet80WadPjKBbUoVjR
```

```
POST /RST.srf HTTP/1.1

Accept:          text/*
User-Agent:      Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1;
Host:            login.live.com
Content-Length: 3525
Connection:      Keep-Alive
Cache-Control:   no-cache

<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/" xmlns:wsse="
    <Header>
        <ps:AuthInfo xmlns:ps="http://schemas.microsoft.com/Passport/Soa
            <ps:HostingApp>
                {7108E71A-9926-4FCB-BCC9-9A9D3F32E423}
            </ps:HostingApp>
            <ps:BinaryVersion>
                4
            </ps:BinaryVersion>
            <ps:UIVersion>
                1
            </ps:UIVersion>
            <ps:Cookies>
            </ps:Cookies>
            <ps:RequestParams>
                AQAAAAIAAABsYwQAAAAxMDQ0
            </ps:RequestParams>
        </ps:AuthInfo>
```

[14:20:42]

```
HTTP/1.1 100 Continue
```

```
HTTP/1.1 200 OK

Connection:      close
Date:            Sun, 04 Feb 2007 13:20:37 GMT
Server:          Microsoft-IIS/6.0
PPServer:        PPV: 30 H: BAYPPLOGN2B29 V: 0
X-Powered-By:    ASP.NET
Content-Type:    text/html; charset=iso-8859-1
Expires:         Sun, 04 Feb 2007 13:19:37 GMT
Cache-Control:   no-cache
```

What is Frida?

- Dynamic instrumentation toolkit
  - Debug live processes
- Scriptable
  - **Execute your own debug scripts inside another process**
- Multi-platform
  - Windows, Mac, Linux, iOS, Android, QNX
- Highly modular, JavaScript is optional
- Open Source

Why would you need Frida?

- For reverse-engineering
- For programmable debugging
- For dynamic instrumentation
- But ultimately: To enable rapid development of new tools for the task at hand

Let's explore the basics

1) Build and run a simple program that calls f(**n**) every second with **n** increasing with each call.

2) Let's figure out what **n** is.

Frida has a REPL. Let's use it.

It live-reloads!

3) Let's modify what **n** is.
How about +9000?

4) Let's speed up time.

5) Let's call f() ourselves.

6) rpc, send() and recv().

Let's see what files Telegram open()s on macOS

Let's try interacting with Objective-C

Let's figure out who is calling open().

Let's inspect registers.

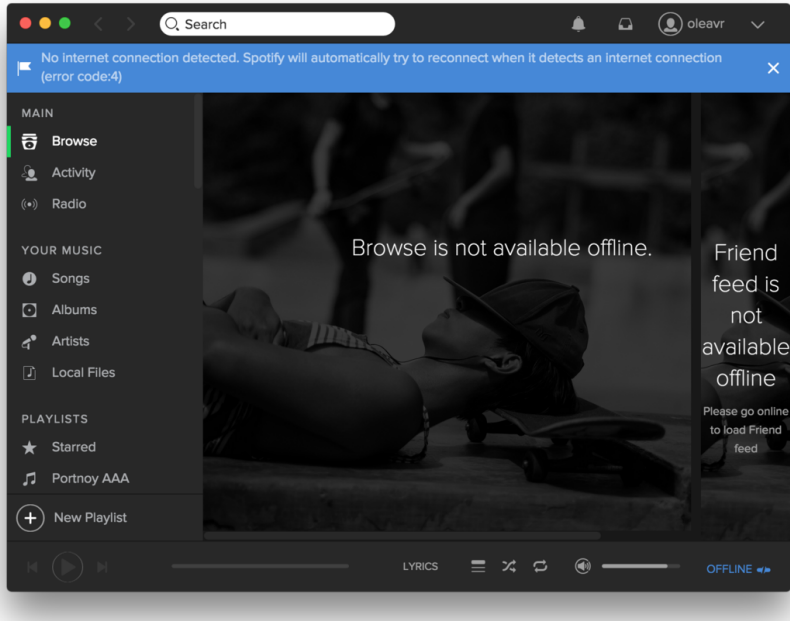
Let's try frida-gadget.

Let's explore early instrumentation.

Injecting errors

```
'use strict';

const AF_INET = 2;
const AF_INET6 = 30;
const ECONNREFUSED = 61;

['connect', 'connect$NOCANCEL'].forEach(funcName => {
  const connect = new NativeFunction(
    Module.findExportByName('libsystem_kernel.dylib', funcName),
    'int',
    ['int', 'pointer', 'int']);
  Interceptor.replace(connect, new NativeCallback((socket, address, addressLen) => {
    const family = Memory.readU8(address.add(1));
    if (family == AF_INET || family == AF_INET6) {
      const port = (Memory.readU8(address.add(2)) << 8) | Memory.readU8(address.add(3));

      let ip = '';
      if (family == AF_INET) {
        for (let offset = 4; offset != 8; offset++) {
          if (ip.length > 0)
            ip += '.';
          ip += Memory.readU8(address.add(offset));
        }
      } else {
        for (let offset = 8; offset !== 24; offset += 2) {
          if (ip.length > 0)
            ip += ':';
          ip += toHex(Memory.readU8(address.add(offset))) +
              toHex(Memory.readU8(address.add(offset + 1)));
        }
      }

      console.log('connect() family=' + family + ' ip=' + ip + ' port=' + port);
      if (port === 80 || port === 443 || port === 4070) {
        console.log('  blocking!');
        this.errno = ECONNREFUSED;
        return -1;
      } else {
        console.log('  accepting!');
        return connect(socket, address, addressLen);
      }
    } else {
      return connect(socket, address, addressLen);
    }
  }, 'int', ['int', 'pointer', 'int']));

  send('ready');
});

function toHex(v) {
  let result = v.toString(16);
  if (result.length === 1)
    result = '0' + result;
  return result;
}
```

```
$ node app.js Spotify
connect() family=2 ip=78.31.9.101 port=80
  blocking!
connect() family=2 ip=193.182.7.242 port=80
  blocking!
connect() family=2 ip=194.132.162.4 port=443
  blocking!
connect() family=2 ip=194.132.162.4 port=80
  blocking!
connect() family=2 ip=194.132.162.212 port=80
  blocking!
connect() family=2 ip=194.132.162.196 port=4070
  blocking!
connect() family=2 ip=193.182.7.226 port=443
  blocking!
```

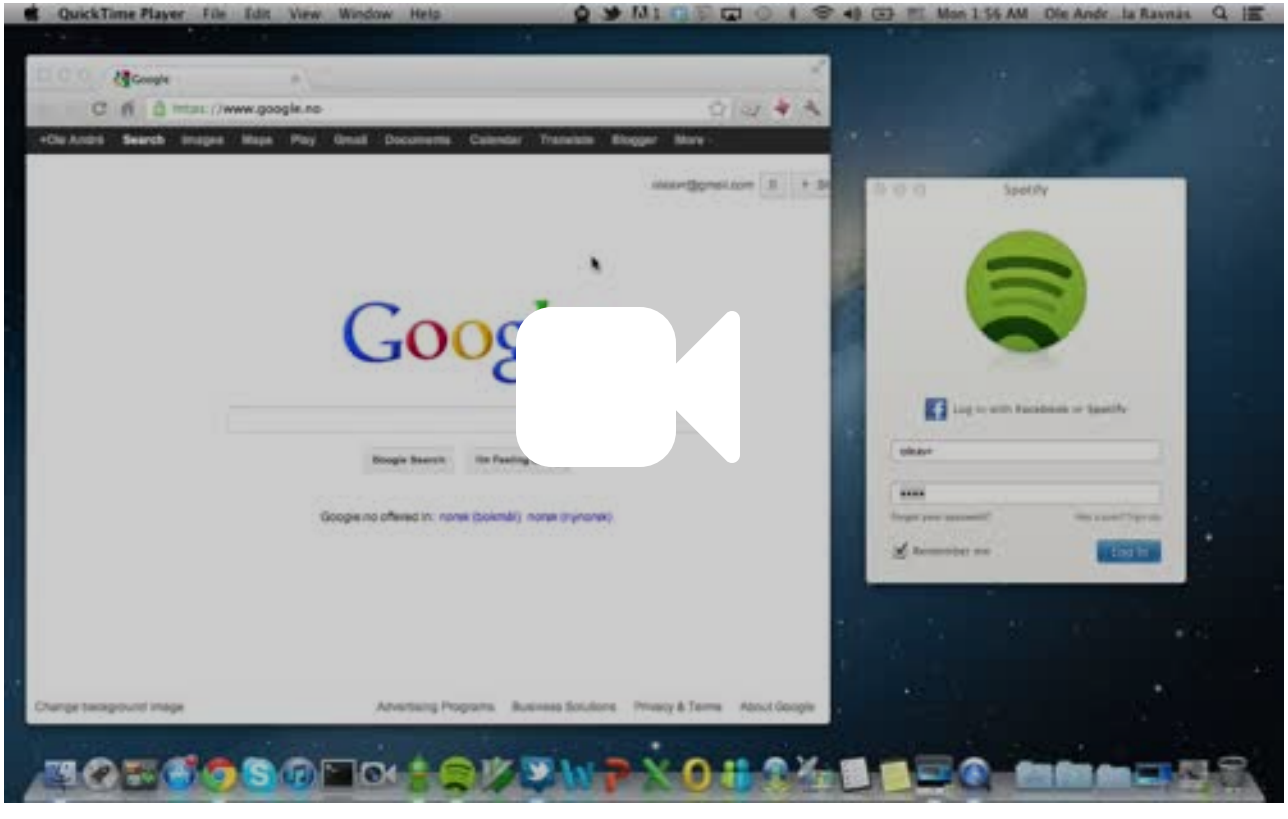


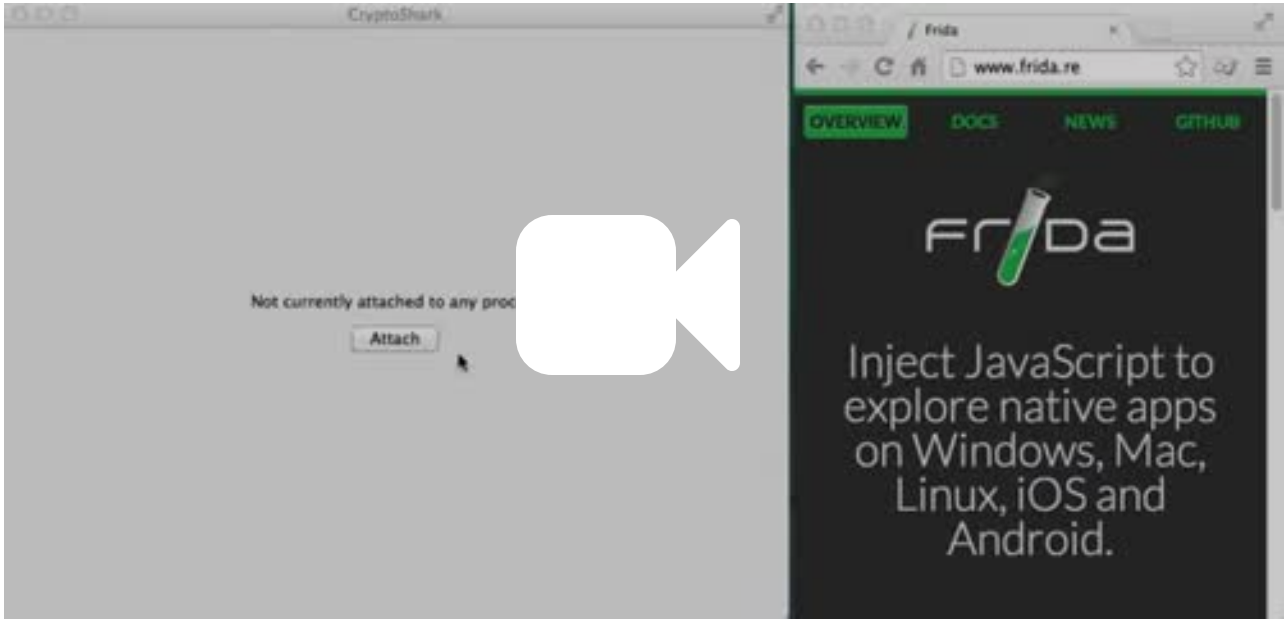
All calls between two recv() calls

```
'use strict';

const co = require('co');
const frida = require('frida');
const load = require('frida-load');

let session, script;
co(function *() {
  session = yield frida.attach(process.argv[2]);
  const source = yield load(
      require.resolve('./agent.js'));
  script = yield session.createScript(source);
  script.events.listen('message', message => {
    if (message.type === 'send') {
      const stanza = message.payload;
      switch (stanza.name) {
        case '+ready':
          console.log('Waiting for application to call recv()...');
          break;
        case '+result': {
          console.log('Results received:');
          const events = stanza.payload.events;
          events.forEach(ev => {
            const location = ev[0];
            const target = ev[1];
            const depth = ev[2];
            let indent = '';
            for (let i = 0; i !== depth; i++)
              indent += '   | ';
            console.log('\t' + indent + location + '\tCALL ' + target);
          });
          session.detach();
          break;
        }
      }
    } else {
      console.log(message);
    }
  });
  yield script.load();
});
```

```
'use strict';

const WAITING = 1;
const STALKING = 2;
const COLLECTING = 3;
const DONE = 4;

let state = WAITING;
let stalkedThreadId = null;
let blobs = [];

['recv', 'recvSOCKANCIL'].forEach(funcName => {
  Interceptor.attach(Module.findExportByName('libsystem_c.dylib', funcName), {
    onEnter: args => {
      if (state === STALKING && this.threadId === stalkedThreadId) {
        state = COLLECTING;
        Stalker.unfollow();
      }
    },
    onLeave: retval => {
      if (state === WAITING) {
        state = STALKING;
        stalkedThreadId = this.threadId;
        Stalker.follow({
          events: {
            call: true
          },
          onReceive: events => {
            blobs.push(events);
            if (state === COLLECTING) {
              sendResult();
              state = DONE;
            }
          }
        });
      }
    }
  });
});

send({
  name: '+ready'
});

function sendResult() {
  const events = blobs.reduce((result, blob) => {
    const cursor = {
      data: blob,
      offset: 0
    };
    let e;
    while ((e = nextEvent(cursor))) {
      result.push(e);
    }
    return result;
  }, []);
  send({
    name: '+result',
    payload: {
      events: events
    }
  });
}

function nextEvent(cursor) {
  // FIXME: 32-bit support
  const data = cursor.data;
  if (cursor.offset === data.length)
    return null;
  skipEventType(cursor);
  const location = readPointer(cursor);
  const target = readPointer(cursor);
  const depth = readDepth(cursor);
  return [location, target, depth];
}

function skipEventType(cursor) {
  cursor.offset += 8;
}
```

```
$ node app.js Spotify
Waiting for application to call recv()...
Results received:
        0x119875dc7     CALL 0x119887527
        0x119875e7      CALL 0x11989a1e6
        0x1197f4df      CALL 0x11992f934
        0x1197f4f3      CALL 0x1197edd7d
        0x7fff8acdf6ad  CALL 0x7fff8ace32dc
            | 0x7fff95355059     CALL 0x7fff9535c08b
        0x7fff937774be  CALL 0x7fff9375d5a0
            | 0x7fff9376e76      CALL 0x7fff93788d6e
            | 0x7fff9376e722     CALL 0x7fff93788d2c
            |     | 0x7fff8d1e9754        CALL 0x7fff8
            |     | 0x7fff8d1e9765        CALL 0x7fff8
            |     | 0x7fff8d1e9421        CALL 0x7fff8
            |     |     | 0x7fff8d1e95bf   CALL 0x7fff8
            |     |     |     | 0x7fff8d1e7417      CALL
            |     |     | 0x7fff8d1e95eb   CALL 0x7fff8
        0x7fff9377752c  CALL 0x7fff93788d9e
            | 0x7fff8d1ed7c8     CALL 0x7fff8d1e721
        0x7fff9377754e  CALL 0x7fff93788b5e
            | 0x7fff8acdfd10     CALL 0x7fff8acdec91
            |     | 0x7fff8acded53        CALL 0x7fff8
            |     |     | 0x7fff95352182   CALL 0x7fff9
            |     |     |     | 0x7fff95353663      CALL
            |     |     |     |     | 0x14ce858a     CALL
            |     |     |     |     |     | 0x7fff9535bb4e
            |     |     |     |     |     |     | 0x7fff953
            | 0x7fff8acdfd20     CALL 0x7fff8ace3348
            | 0x7fff8acdfd48     CALL 0x7fff8acde877
            |     | 0x7fff8acde8ce        CALL 0x7fff8
            |     |     | 0x7fff95352182   CALL 0x7fff9
            |     |     |     | 0x7fff95353663      CALL
            |     |     |     |     | 0x14ce858a     CALL
            |     |     |     |     |     | 0x7fff9535bb4e
            |     |     |     |     |     |     | 0x7fff953
            |     | 0x7fff8acde8e1        CALL 0x7fff8
            |     | 0x7fff8acde923        CALL 0x7fff8
            |     |     | 0x7fff8acdd6ad   CALL 0x7fff8
            |     |     |     | 0x7fff968e0ef       CALL
            |     |     | 0x7fff8acdd6b5   CALL 0x7fff8
```


QuickTime Player File Edit View Window Help
Mon 1:56 AM
https://www.google.no
Search
Spotify
Google Search
Google.no offered in:
Change background image
Advertising Programs
Business Solutions
Privacy & Terms
About Google


CryptoShark
Not currently attached to any proc
Attach
Frida
www.frida.re
OVERVIEW
DOCS
NEWS
GITHUB
Frida
Inject JavaScript to explore native apps on Windows, Mac, Linux, iOS and Android.

EOF!

Please drop by

https://t.me/fridadotre

(or **#frida** on FreeNode)