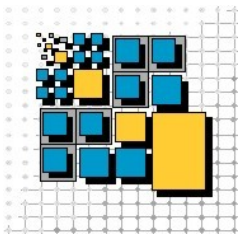
PuppeTor

Performing automatic tests and measurements in Tor networks

Karsten Loesing

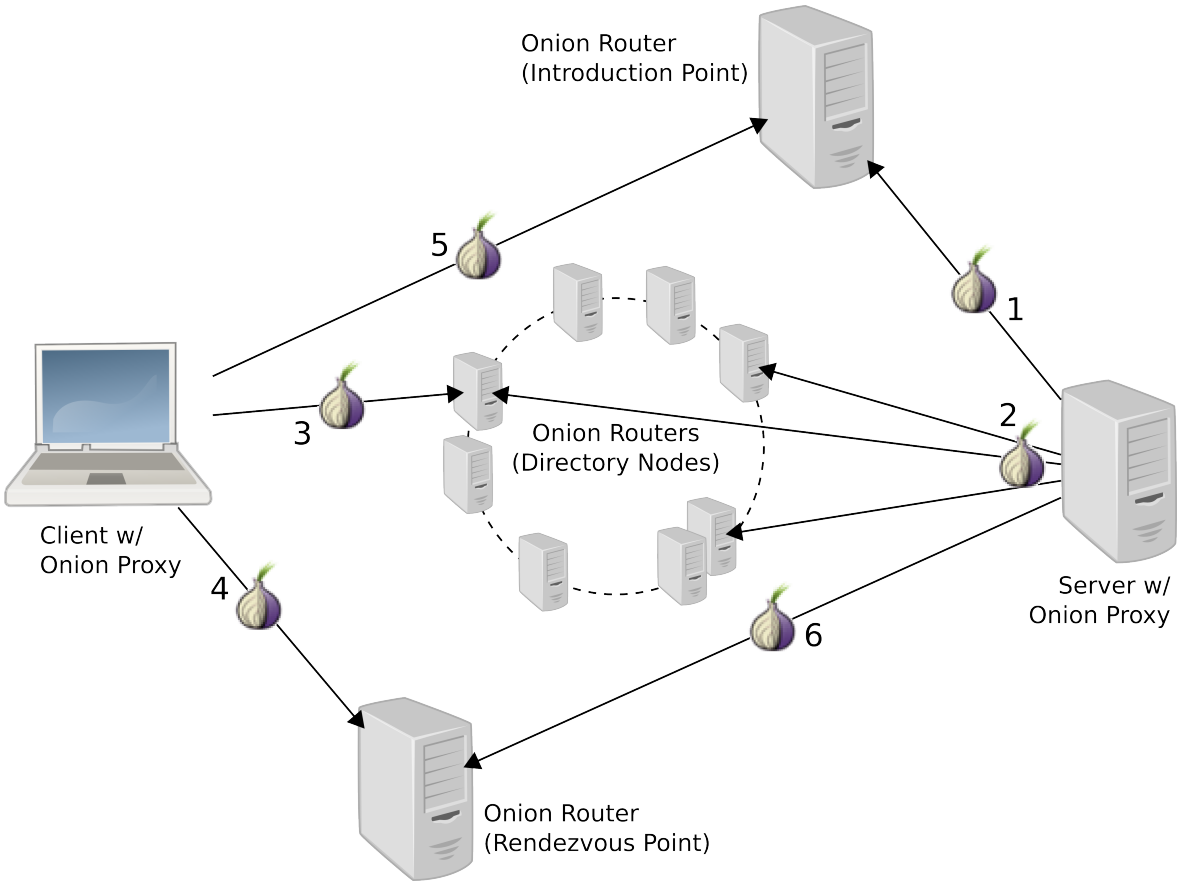
Distributed and Mobile Systems Group, University of Bamberg

PET-Con 2007, Frankfurt, 2007-08-16



A short note on Google Summer of Code 2007

- 3 directory nodes store and serve all hidden service descriptors
- Distribute among large subset of all onion routers ($\approx$ 1000)
- Improve availability and scalabilty, enable new security features



Today's focus is the by-product: PuppeTor

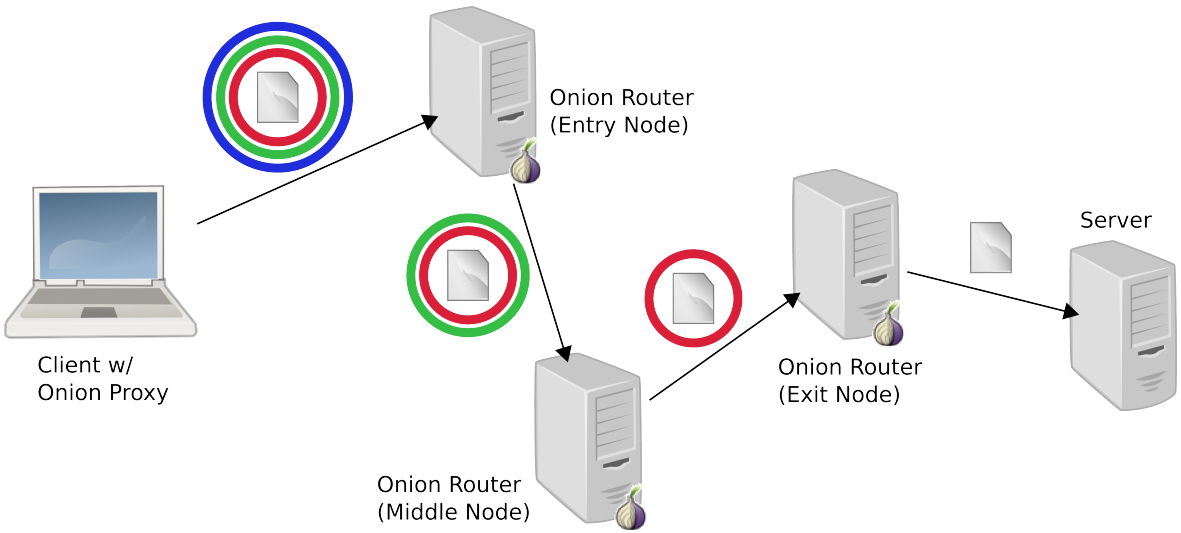
- How long do (what parts of) hidden service requests take? (cf. Christian's talk)
- Does propagation of new routing information and of descriptors work properly?
- The best way to find out: Configure and run one or more Tor nodes and parse their output/logs!
- Assumption: All observed Tor nodes run on a single physical node
- PuppeTor – A Java-based Tor Simulator

The marketing slide

- Tor node configuration with 3 adjustable standard configurations (proxy, router, directory)
- Ability to reconfigure nodes at runtime over control port (currently unused)
- Configuration as private network by approving routers to directories
- HTTP client and server for performing/answering requests
- Asynchronous event mechanism from parsing logs for synchronization of test application and for measurements
- More powerful/convenient than writing shell scripts and grepping logs
- Configuration, execution, and evaluation at a single place

PuppeTor at work

- A getting-started example for using PuppeTor
- Measuring the round-trip time of an anonymous access to Google



Configuring a test network

- Create initially empty network configuration:

```
Network network =
    NetworkFactory.createNetwork("example1");
```

- Create Tor proxy node to relay the request:

```
network.createProxy("proxy", 7001, 7002);
```

- Create client application to perform the request:

```
ClientApplication client =
    network.createClient("client",
    "www.google.de", 80, 7002);
```

- Write configurations to `torrc` file:

```
network.writeConfigurations();
```

Handling asynchronous events

- Obtain reference on test-global event manager:

```
EventManager manager =
    network.getEventManager();
```

- Create custom event listener:

```
EventListener clientEventListener =
    new EventListener() {
  public void handleEvent(Event event) {
    System.out.println(event);
} };
```

- Add event handler for client events:

```
manager.addEventListener(client,
    clientEventListener);
```

Test execution

- Start proxy node with timeout of 5 seconds:
  `network.startNodes(5000);`
- Restart nodes until they have created their first circuit:
  `network.hupUntilUp(5, 10000);`
- Perform request with at most 3 retries and a timeout of 20 seconds for each try:
  `client.performRequest(3, 20000, true);`
- Wait for request to be performed:
```
manager.waitForAnyOccurence(client,
    Event.CLIENT_REQUESTS_PERFORMED);
```
- Shut down proxy node:
  `network.shutdownNodes();`

More complex examples

- Advertising hidden service (1 proxy node)
- Advertising and accessing hidden service (2 proxy nodes)
- Advertising and accessing hidden service in private Tor network (3 router nodes, 2 directory nodes)
- Test of distributed storage of descriptors in private Tor network (9 periodically changing router nodes, 2 directory nodes)
- "Performance Evaluation of Tor Hidden Services", cf. Christian's talk (1 proxy node)

Idea for the workshop

- Perform some tests/measurements in Tor using PuppeTor
- Discuss and elaborate alternative approaches