Location: Bits of Freedom (Amsterdam, The Netherlands)
Friday: 10:30am–5pm GMT +0100

1. Meeting goals
   * Meet and greet
   * Accomplish coding tasks that could be done in a weekend
   * Get some of the people lined up who might want to be resident with us at GT.
   * What from BISmark+OONI might be doable in a weekend?

2. Attendees
   In-Person Participants
   * Georgia Tech/BISmark
     ** Nick
     ** Sam
     ** Giuseppe (from Napoli)
   * Measurement Lab
     ** Meredith Whittaker
     ** Dominic Hamon
   * Tor/OONI
     ** Roger Dingledine
     ** Arturo Filasto
     ** Jake Applebaum
     ** Aaron
     ** Isis Lovecruft
     ** George Kadianakis

   Remote Participants (via G+ Hangout on Friday)
   * Georgia Tech/UMD/BISmark
     ** Antonio Pescape
     ** Walter de Donato
     ** Wei Meng
     ** Hans Klein
     ** Ed Felten

3. Friday, 10:30am-12:00pm: Agenda Bashing

   * (How) could BISmark and OONI be merged?
     -- ooni proxying (privacy and control issues)
     -- how to make OONI more lightweight so that it can run on routers and smartphones
     -- how can OONI work for us?
     -- discuss development roadmaps
     -- Feedback on OONI development tasks
-- Organization of ooniprobe/BISmark workshop?

It would be great to inject a political perspective here. I realize this can be onerous, but there are a number of initiatives in their early stages that could substantially increase OONI's footprint (the European Commission’s No Disconnect, for one). If we can time OONI + M-Lab development with an eye here, I can help rally support in the right places.

* **Status updates:** Let’s decide if we want to keep paying attention to and how we want to use (if at all) the following technologies we discussed at the last meeting.
  * Tor router (seems like a likely yes)
  * Dream plug
  * Seattle
  * Hobbit

* **Mobile measurement.** Should/could we have a version of OONI that runs on smartphones? Would be nice to get an update on Orbot from Sathya and his team.

* **Data curation and presentation.** Propose posting data in a location where others can analyze. MLab background.

* **Setting up OONI collector/back-end at Georgia Tech.** Does the OONI collector actually exist yet? If not, might be a good thing to work on for the weekend. Is this something we should be coordinating with the measurement lab folks on? Boils down to setting up an MLab server at Georgia Tech.

* **OONI/Tutorial Ramp-Up.** ([http://ooni.torproject.org/docs/writing_tests.html](http://ooni.torproject.org/docs/writing_tests.html))

* **Policy discussion.** As a technical measurement project, there is a balance between choosing what to measure in terms of having the most political impact and never specifying the political impact that we have in mind.

---

**Starting Discussions**

How to write OONI tests:

[http://ooni.torproject.org/docs/writing_tests.html](http://ooni.torproject.org/docs/writing_tests.html)

Some points that came up in initial discussions:

* Need to measure all countries, not just the ones that we think are censoring.
* How to encourage users to run this tool in the first place.

Barriers to entry: Can running the measurement cause the authorities to come knocking at your door?
Do we want to ask someone who has served on an IRB with knowledge of international law about some of the nuances concerning informed consent (specifically, of running censorship measurements)?

In the US, the bottlenecks may in fact be the users themselves.

Deployment at Web caches, etc.—the further you are from the user, perhaps the better.

Deploying OONIprober from Tor relays and nodes to fill in the gaps concerning the CAIDA AS graphs. Traceroutes from Tor nodes and MLab nodes.
- Possible tool: Paris Traceroute

Another strategy: Passive measurements. Trigger an active measurement after the agent observes something in passive measurements. OONIprobe supports this already. Could be used to determine whether user is triggering anonymous redirects, etc.

MLab doesn’t reveal anything except the fact that a user asked that the test be run.

---

**High-Level Goals of OONI Project**

* Jake: Discussion of high-level goals
  Evidence-based policy and discussion
  Not to advocate for any particular agenda, but make sure that the discussions of human rights are based on data.
  Framework where everyone can participate in telling the world about their experience
  Open method and taxonomy (and data) (So thinks like “DNS censorship” means something.)
  Open measurement code. So that there are multiple independent ways of producing similar datasets.
  Common, non-partisan agreement about what types of activities are taking place.
  Human-rights observation is something that is very difficult to speak out against.

OONI mainly cares about developing a common taxonomy. Want to have data when the tool runs vs. others, that the output should be similar, and things should be cohesive, so that other tools can help complete the picture.

* Meredith:
  Need to have people who are actively in this space implementing, validating many tools, etc.
  Not centralized control, but a serious active community in this area.

Others: ONI (dissolving), Herdict, secdev weasel team, Netalyzr/Fathom
Failure modes for these projects in the past:
* We do tests, but we won’t tell you what they are (secdev)
* We’re good at talking, but we don’t really understand how the Internet works (Herdict)

Problems with ONI and Herdict projects: no open data, taxonomy, etc. e.g., can’t ask Herdict for all of the URLs that are censored. Many of the basic principles of openness are totally not a given. Anyone we want to bring into the community should be on board with evidence-based argumentation.

(MLab has a five-point manifesto for openness that could be adapted for this purpose.)

Should provide for measurements of the performance of a binary blob (e.g., UltraSurf).

If someone wants to provide a list, we should have tools to measure it. If someone produces a list of things to test, great, we will test it! The fact that ONI is shutting down might be a treasure trove, since they may ultimately release URLs.

Freedom House map of the world colored by censorship. One of our “win” conditions is if they switch to using our data rather than generating their own. The back-end that they use to make our color decisions could serve as one component for them generating their reports.

Another project: Chokepoint.

Vantage points: Current state and how do we get more?
Currently, people hop on IRC and ask to run OONI, and Arturo sets them up. Reports from Turkey, Russia, etc. People going to places where they expect they’re going to find interesting things. Some things can be done in a more centralized fashion (e.g., scanning open SOCKS proxies). Goal is to get more vantage points. Damon McCoy at UCSD was creating a gigantic list of SOCKS proxies. One of the problem with SOCKS proxies is that many of them are not long-lived. But Damon has paid to get an up-to-date list.

Giuseppe has done some work on scanning these proxies.

* OONI: no current vantage points. moving towards a daemon architecture. dozens have run it.
* BISmark: giuseppe tried DNS checks and captive portal checks, as well as TCP reachability nodes. from < 100 BISmark vantage points, mostly in the US. started with the Alexa top 100, as well as a list of 1,000 proxies. no real evidence of censorship thus far.

Continually running tests from a variety of places. How to give users a reason to install it? What immediate service can this provide?

Might be able to fire off OONI tests on BISmark routers under the current use policies.
Scale? BISmark could provide hundreds.

Roger not so concerned about deployment. Hobbyists could probably run it. Important to get one vantage point that basically runs the whole process. Do it for something really easy; like, “can you do an SSL handshake with a Tor relay?”

Jake has three different ideas for hardware:

- **RIPE Atlas Probe.** Not so good for running software, but might be useful as a GRE tunnel. Advantages: cost. Social incentives for making it work. Drop in and forget.
- **Raspberry Pi.** Runs Tor very well. Connect-back shell over Tor hidden service. $26 w/o a case. $29 w/case.
- **Tor Router.** Higher end. Three tiers: partner up with RIPE, etc. Open thing, can fab hardware yourself. $125-160. 4 GGB or DDR3, GigE, etc. No binary blobs. Possible replacement for NetGear. Currently runs Debian. 4 cores, multiple gigs of memory, etc. Audio, HDMI, etc. In theory, replaces a bunch of devices that you might want. Plus, OONI, Tor, BISmark, ...

Discovery problem: How to basically get the hardware deployed in various countries.

Want to make sure that the image of running OONI doesn’t automatically identify the user as an activist. What other incentives might there be for running this tool.

Part of the point of making this open is that other folks run these tests. Only risky tests are those involve child pornography and terrorism. Active HTTP probing with a list outside of the Alexa top 1000, etc. is the stuff we should potentially be worried about. Separate the user base from activists right from the beginning.

Perhaps worth adding some “safe” tests (e.g., “map the internet”). One test that OONI also already does is some kind of header inspection. Sell OONI as an interesting tool to run outside of the activism.

If lots of tools run the same set of tests, then this is good because it does not necessarily expose the measurement agent as an activist. Get many people from different incentive groups to try to solve this problem. The more people who are a part of this, the better.

- **Hardware platforms**
- **Software/languages**
- **Deployment**
- **Interoperability**
- **Users**

What do we want users to run?
Two types of tests:
- Content blocking
- Traffic manipulation

Inspiration for tests: lots of staring at packet traces (obtained by running tcpdump on traffic coming from various countries).

The following will (initially) run on MLab:

1. **HTTP Invalid Request Line** This tests does some basic fuzzing on the HTTP request line, generating a series of invalid HTTP requests between the OONI test client and the M-Lab server. The M-Lab server runs a TCP echo test helper, and if the response from the server doesn’t match with what was sent, the conclusion is that tampering is occurring. The assumption driving this methodology is that certain transparent HTTP proxies may not properly parse the HTTP request line.

2. **DNS Spoof** This test detects DNS spoofing by performing a query to a test resolver and to a validated control resolver. The query is considered tampered with if the two responses do not match.

3. **Header Field Manipulation** This test client sends HTTP requests request headers that vary capitalization toward an HTTPReturnJSONHeaders test helper backend running on an M-Lab server. If the headers received by the M-Lab server don’t match those sent, tampering is inferred.

4. **Multi-port Traceroute** This test performs a multi port, multi-protocol traceroute from an OONI client toward an M-Lab server. The goal of such is to determine biases in the paths based on destination port. Destination ports are 22, 23, 80, 123, 443. Note that if the user has opted not to include source IP in the report then source and destination IP will be eliminated from the collected data. Note that while a user may be able to opt not to eliminate IP address, we will need to provision an option for those who wish IP not to be collected.

5. **HTTP Host** This test detects the presence of a transparent HTTP proxy and enumerates the sites it is configured to censor. To do this the test places the hostname of a probable censored site inside of the Host header field, and communicates this stream of data between an OONI client and an M-Lab server. If the response from the server doesn’t match the data sent, the test determines the presence of a transparent HTTP proxy.

6. **DNS Tamper** This test performs A queries to a set of test resolvers and a validated control resolver. If the two results do not match it performs a reverse DNS lookup on the first A record address of both sets of queries, checking that they both resolve to the same name. NOTE: This test frequently results in false positives due to GeoIP-based load balancing on major global sites such as Google, Facebook, and Youtube, etc. This will need to be noted and accounted for.
Note: Need to make it easy for people who think of tests to write and deploy them.

Roger’s question: Framework or tests?
Answer: Both.

Point of framework is to integrate as much as possible of what other people think of as tests. For example, the framework will have a notion of an anonymous communications channel.

Data
Important: Outputting compatible data formats. (complete with application name and software version number).

BISmark data format? Something that approaches a column-based data format is OK. OONI data format currently has nested keys.

For a given test, each test should have a specification of the keys, what they mean, which ones are optional, etc.

Can we agree on a data format? Yes.

Data should be annotated with different version numbers
- Specification versions
- Software implementation version (OONI, BISmark)
- OONI test implementation version
- BISmark test implementation version

Initial idea for MLab: store different versions/implementations of the same test could reside in the same table.

Another research problem: How to trust the data? Can identifiers help? Current best solutions (removing all data within a certain time window, as was done with Herdict, tagging trusted users, as is done in ONI) are pretty poor.

We may have some issues to deal with in terms of exposing the user, even if we remove the IP address of the user. Should be very upfront about the fact that we are doing “best effort”.

How to tag users (so that we can identify a user who performs bad tests)? Do we even want that? Need to design the software so that we could augment the data/reports with this later should we decide that this is a good idea. Could provide a very useful handle for removing data from the corpus in case of malfunction or malice.

Testing
OONI has a list of URLs of lists of places to test. Current URLs on the OONI wiki has the Alexa top million. OONI also uses the “Russian blacklists”.

Interesting question: What about URLs that are illegal to query in a particular jurisdiction. What to do about questionable bytes and links? Perhaps tricky business to try to determine which URLs that might have been pointing to CP at some time in the past. These URLs eventually become stale, so including them in the data need not be considered criminalized information. In fact, demonstrating the staleness of these lists is an important service.

Being able to track blacklists and updates that a country is currently using.

---

**Hangout Policy Discussion**

Present remotely via hangout: Ed Felten, Hans Klein, Wei Meng, Walter de Donato, Antonio Pescape

- It would be nice to set up a "censorship lab" to test censorship devices
  - e.g., Bluecoat devices
  - This is very expensive because Bluecoat only sells to governments, but we might be able to team up with friendly folks that already own one. Even if we do buy them, we don't have configurations.
  - (We also don't want to give these companies more money.)
  - There also might be VM images, source code, etc floating around, but we don't have the human resources to do much with this.
- We really need a testbed. Emulation won't work because of configurations, bugs, etc.
  - Even better is a machine behind a real firewall. Raspberry Pis are cheap
- Once we have these probers in place, we can use the same infrastructure to test the effectiveness of various circumvention tools (e.g., Tor, Ultrasurf, etc.)
  - Don't just measure the tool; also measure protocols of the tools (e.g., the Tor handshake.)
  - We need to make tests easy to write. The Tor handshake is complicated; it'd be nice if OONI's test language didn't get in the way.

- It'd be nice if we could get data about all censorship tools
  - Examples:
    - Nick has Verisign TLD data; this could be useful for identifying and counting users of other censorship tools
    - Colin Anderson tracked Tor and Ultrasurf usage in Iran. His data might be useful.
- Jake has a Chinese report about Ultrasurf but can't read it. Wei will translate it for him.

*Comments from Hans:*
- There's an alternative, non-technical approach to preventing circumvention. Instead of blocking tools, we would dissuade policy makers from deploying censorship tools.
- This also works in reverse: censors could dissuade users of tools; tools would technically work, but noone would use them (e.g., out of fear.)
- For many governments, it would be good to understand the top-down decision processes. For example, we could increase the cost of implementing circumvention by making it politically inconvenient.
- wrt our group: we should promote these Hans' ideas, but indirectly; we promote others toward these policy goals, but don't engage them ourselves. Who do we engage for this? Reports Without Borders, Freedom House, etc. Lots of people want our data, but we still need to make the data available and clearly convey the limitations of the data.
- We should keep track and promote organization that do "responsible analysis" of our data, if that's even possible. (Important side not: we should promote organizations that do good analysis, not organizations that promote our world view.)
- There currently aren't any such organizations yet, mostly because we don't have data yet.
- Meredith and Hans might discuss this more offline.

- Question from Hans: is our group prepared to embarass developers of censorship tools? Answer: Yes, it's already happening, but it's not clear if this is affecting corporate policy.

Summary:
1. Evolution of blocking technologies and activities over time. We need a comprehensive test suite. How does blocking evolve over time, in response to new tests and new circumvention tools. Our goal is to predict what censors will do well in advance.
2. Can we correlate usage of circumvention tools in response to blocking?

Measuring censorship of the network vs. censorship of the Tor network specifically. Getting OONIprobe data into public data explorer would be pretty easy, since the data is already in tabular format.

Takeaways from Today / Stuff to talk about tomorrow

Jake: Wants to show off the Tor router tomorrow. Also Raspberry Pi as a secondary platform. Want to talk about technical details tomorrow. Split into smaller working groups. Make it easy to flit between groups. Possible groups: Output data format; make it possible to build the tools (e.g., can Jake build BISmark on his router by the time he leaves?).

George: Prioritize the next set of tests for OONI. Getting BISmark tests running whereever OONI is deployed.

Arturo: Tutorial on how to write tests.
Roger: How to figure out how he can be most useful. End tomorrow knowing that Arturo knows how to get a test deployed so that MLab can publish the results.

Nick: Is there a “Hello World?” example for the OONI probe -> MLab pipeline? Getting BISmark active into this pipeline could be part of the story. **How to bootstrap tests.**

Isis: Cross-compilation of OONI for Android. Possible? How to make it work?

Dominic: Specifics of building something that will work across platforms.

Giuseppe: OONI-like tools running on BISmark. The more similarities that exist between the harness and the data output. Getting OONI-probe or equivalent on the BISmark routers. How to get on the routers.

Sam: Wants to learn more about OONI

Meredith: Test prioritization

Aaron: Interested in writing some of these tests

OONI deployment, common data format

Working Groups/Themes:
- Hardware (funding, pet projects, etc.) - everyone
- Test Implementation/Hacking (+ backend for non-ooniprobe tests?)
- Data Formats
- Test Prioritization
  - (including aspects of BISmark active that could be part of OONI-default?)
- Platforms/Cross-compilation

**Saturday, January 19**

**11a: OONI Tutorial (Arturo)**

IRC help from Arturo: #ooni on oftc.net (Arturo is hellais.)

There is some setup to be done. [https://github.com/hellais/ooni-probe#getting-started](https://github.com/hellais/ooni-probe#getting-started)

Walkthrough of [https://ooni.torproject.org/docs/writing_tests.html](https://ooni.torproject.org/docs/writing_tests.html)

Base class is NetTestCase.
- inputFile class attribute determines the inputs that are used.
- UsageOptions to specify command-line options. (subclass of twisted's UsageOptions class). Same as the twisted's UsageOptions class. self.localOptions to retrieve the values of various options. Can set required options with local requiredOptions attribute.
- test methods have "test_" prefix. anything with test_ gets called. these then get appended to the report.
- self.report['foo'] = bar gets written into the report
- self.input refers to the current input

Test Templates
- Base templates that allow the programmer to have some utility class methods that perform operations that will then be included inside of the report.
- based on Scapy.
- familiarize oneself with the concept of deferred in the twisted framework. results in nested functions

Walkthrough of different types of tests
- scapy
- TCP
- HTTP
- DNS

Question: What if I want to run a DNS test and and HTTP test as part of the same test? Do we have to inherit from both? Can compose. Hopefully no clashing keys.

How to include a test:
- subclass
- drop in nettest
- workon ooni (virtual environment stuff for python)
- /bin/ooniprobe --help

12:30p: Router Hardware Discussion

Goal: Free hardware/software platform that is not dependent on vendors.

Goal: measurement, something that runs Tor, can be a bridge

Dream plug is not an option.

Three options:
- RIPE Atlas Probe. Disaster to rely on this kind of hardware. Locked down hardware.
- Raspberry Pi. Not actually free (can't build it yourself). Requires loading a proprietary HDMI driver for the device to actually turn on. Benefits: small, available, known piece of hardware. Familiar to hackers, bloggers. Unit cost: $30-35. Another $8 for a USB WiFi 802.11 a/b/g/n. So much proprietary stuff. Very easy to reprogram. 3D printed case costs a buck or two. 800 MHz, 1.2 GB RAM. 20-30 Mbps without maxing out the CPU. 10/100 RJ45 interface. (Alternative: Arduino.) Runs Debian. apt-get “just works”.

• Tor Router: $150 not including wireless interface or case. (assuming 1000 boards). Fully spec’d out here: http://www.bunniestudios.com/blog/?p=2686
  o UIM slot for plugging in a SIM card. SIM card is a generic device
  o SD Card slot
  o ESAT port for plugging in a hard drive (if desired)
  o Mini PCI port (e.g., for plugging in WiFi module). To Do: Send Jake Mini PCI Ath 9k chips that we want Jake to install.
  o 10/100 Ethernet port
  o GigE port
  o USB “on the go” port
  o 2 USB high-voltage ports (charging, etc. is possible)
  o Audio in/out port. (want people to have the ability to replace media servers, etc.). Costs $1, but could significantly increase adoption.
  o on-board microphone.
  o Form factor: currently about 6” x 6” (prototype board). Finally, the form factor will be about 6” x 3”.
  o Port for plugging in an LCD ribbon
  o port for battery charger (for charging)
  o three high-speed serial ports (could bolt-on GPS to this)
  o accelerometer (e.g., for clearing all of the keys if the box is moved)
  o pins for h/w intrusion detection
  o GPIO pin set (same as Raspberry Pi)
  o 2 GB of DDR3 RAM (can support as much as 4GB)
  o Spartan FPGA
  o 4 ARM9 cores. 1-1.3 GHz per core. RNG built in, as well as AES acceleration.

Want to have a reliable base OS (Debian).

Can ship direct from factory. Could have a “click to buy”. Can I buy it readily configured as a home router?

Funding questions.
How many units for first order to bring unit costs down?

5:00p Discussion of Tests

Categories of tests:
  • Interference/Blocking Tests
  • Manipulation Tests: Some transformation has been performed on the traffic that you are sending (extra headers, extra latency, etc.).
  • Circumvention tools

Properties of tests:
  • They have an input
● They detect packet mangling
● Requirement for back-end
● Incorporates outside/auxiliary information
● Needs client/server
● May return false positives
● Requires root
● Stateful/stateless

List of tests in the initial M-Lab deployment
● HTTP
  ○ requests
  ○ headers
● DNS tampering (should be called “DNS (in)consistency”)
● Traceroute
● Captive portal

Other implemented tests
● Man in the middle (SSL, SSH)
● Daphne: We open 'n' connections with the backend, and foreach connection 'i' we mutate the 'i'th byte of the conversation. When the conversation is no longer blocked it means that censor can no longer find the fingerprint in our packets, and that the last mutated byte is part of the DPI fingerprint. (requires server-side backend). Does bidirectional tests, etc.

“Want” (Other tests worth considering)
● Vern Paxson-style breaking of keywords across packets to see whether that still results in blocking
● Do Host fields with just IP addresses get blocked? Do host fields whose names do not match the IP address corresponding to the connection get blocked?
● Transcoding (JPEG,MPEG) tests. If I send an image or video, does the same thing show up on the other side?
● Generalized: do particular byte sequences trigger actions
  ○ for various types of connections: SCTP, TCP, UDP, etc.
  ○ regardless of state of connections
  ○ in other parts of the protocol (e.g., in an HTTP cookie)
● Testing for whether HTTP connections with various browser versions are blocked (or not)
● Timing measurements (basically, latency)
● {Packet loss, jitter, latency} detection (e.g., to different destinations)
● NAT
● IPv6 (perhaps all of the tests could be running concurrently/coupled with IPv4 tests). Hypothesis might be that censorship is less pervasive on IPv6 than on IPv4 because some middleboxes may not support IPv6, or the IPv6 paths may be entirely different and not even go through the same middleboxes.
Some discussion of the EFF Switzerland tool differential packet trace analysis tool to detect on-path packet mangling.

Where are the sources of information that could help us develop tests? (e.g., Citizen Lab report on BlueCoat).

Want multiple independent implementations of the same tests.

**Parameters for Prioritizing Tests**
Goal: Wide breadth of tests that are “good enough” to gather some data.

- Do we have other tests like it?
- Effort required
  - to make a specification for the test. (really important, particularly if there are multiple implementations of the same test. specification *is* the test, and the test *is* the specification. the specification is versioned according to, e.g., the date).
  - constraint: make no claims based on data that is gathered based on code that is written without a specification
  - to get it gathering data (really important) — goal being to iterate quickly to gather a lot of data.
    - Is MLab willing to store the data?
  - to get it stable
- Impact (how much censorship it will detect, based on data collected by other tools, anecdotal evidence, etc.)
- How many places is it likely to be run? How expensive is it to run this test: how much are we asking the user, in terms of bandwidth, how aggressive it is, does it require root, does the tool require downloading other stuff, etc.
- Likelihood of conclusive result as being censorship (vs. some other kind of innocuous behavior (likelihood of false positive)
- Necessary behavior for the test to work correctly vs. “this was simply implemented that way”

**How to prioritize tests?**
Multiprotocol parasitic traceroute
Timing analysis + differential packet loss, jitter, latency
Bridget

**What does it mean to create a test?**
Part of a well-specified production test is (1) specification; (2) implementation.

**What should go in a specification for a test?** (Jake has written all of this on the OONI project wiki in a more complete manner.)
- version number
- name
- inputs (format/syntax and version number)
- outputs (format/syntax, semantics and version number)
- preconditions
- postconditions
- where the data should be stored (i.e., what's public, MLab, private, etc.)
- Also include “parameters for prioritizing tests” (above; e.g., impact, who should run the tool, likelihood of false positives, etc.) so that it’s clear *why* this is a useful test to implement

Packet capture considerations

OONI Spec documentation will go here:
https://github.com/TheTorProject/ooni-spec

**Cross-platform implementation**

Dominic’s proposal: Do some of the low-level support stuff in C, but then have bindings in Java, Lua, Python, etc.

Reduces memory constraints on OONI.