Towards a Censorship Analyser for Tor

Philipp Winter
*The Tor Project & Karlstad University*

2013-08-13
Some countries, corporate firewalls, captive portals and ISPs block Tor.

Blocks become known through users and dropping usage statistics.

Incidents are then analysed to either modify Tor or motivate new censorship-resistant protocol (see obfsproxy et al.).
Example: pre-election censorship in Iran

![Graph showing the number of users in Iran over the months from January 1 to February 15. The data shows a sharp increase in January, followed by a decrease.](image-url)
Motivation for this paper

- Analysis of censorship incidents not always straightforward.

- Two typical analysis scenarios
  - Obtain shell inside censoring network and debug Tor handshake.
  - Obtain network trace of Tor bootstrapping and study it.

- Problems
  - No shells.
  - No network traces.
  - Dependence on technical volunteers.
Our approach to the problem

- How about (unskilled) users do the censorship analysis for us?
- Provide a small tool which automatically gathers analysis-relevant data.
- Comes with novel technical and ethical challenges.
- Important: respect user’s privacy and security.
What our analyser does
Analysis steps (1/3)

- Create a network trace of analysis
  - Should be optional.
  - Must only cover analysis.

- Obfuscate tests
  - Randomise order of executed tests.
  - Use random sleep periods between tests.

- Probe the website
  - Try to download the index page.
  - Resolve www.torproject.org and check A records.
  - Experiment with TLS SNI and perhaps HTTP Host header.
Analysis steps (2/3)

▶ Probe the directory authorities
  ▶ Authorities are a popular choke point.
  ▶ Try to download the consensus.
  ▶ If it fails, ping and traceroute the authorities.

▶ Test relay reachability
  ▶ Connect to relay found in consensus.
  ▶ Step through TLS handshake.
  ▶ Send Tor-specific TLS client hello to unrelated machine.

▶ Test bridge reachability
  ▶ Bridges are relays not listed in the consensus.
  ▶ See if pluggable transport protocols work.
Gather debug information

- What ISP does the user have?
- What is the autonomous system number?
- Is the user behind a captive portal?
- Is all traffic routed through an HTTP proxy?

Anonymising reports

- Network traces, IP addresses, ASNs, whois and traceroutes can be discarded.
- However, anonymous submission is hard → Tor unavailable.
Think about the users

- Analyser must be as easy to use as possible.
- Provide user-friendly output with little jargon.
- Cover our analyser’s tracks and delete reports after submission.
- Informed consent: analyser should inform users about analysis steps and make it easy to abort process.
Create usage diversity

- Based on analysis results, we can recommend further steps.

- Therefore, our tool’s only purpose is no longer to assist in censorship circumvention.

- Usage diversity should make having a copy of our tool less suspicious.
Report submission

- We end up with a text file containing YAML-like data.
- Report could be submitted using email or instant messaging.
- Hard-coded OpenPGP public key could be used to encrypt report.
- Report content can be anonymised but report submission hard to do anonymously.
No need to reinvent the software wheel!

- **OONI** is a modular framework for censorship analysis and network interference (see FOCI’12 paper): https://ooni.torproject.org.

- We implement our analyser as several OONI tests.

- Finally, bundle OONI with our tests to a click-and-go executable.
class TestTorDNSEntries(DNSTest):
    a_records = ['38.229.72.14', '38.229.72.16', '86.59.30.40', '93.95.227.222']
    domains = ['www.torproject.org', 'bridges.torproject.org']

def test_domains(self):

    def gotResult(result, domain):
        self.report['a_records'] = result

        if set(result).intersection(self.a_records) == set(self.a_records):
            print 'Host names resolved as expected.'
        else:
            print 'WARNING: unexpected resolved host names!'

    for domain in self.domains:
        d = self.performALookup(domain, ('8.8.8.8', 53))
        d.addCallback(gotResult, domain)
    return d
Discussion

▶ Our analyser is not unobservable!

▶ Users with very strong threat models should not use the analyser.

▶ Additional desirable features
  ▶ Grammatical inference algorithm to uncover DPI fingerprints.
  ▶ Identify/cluster exact model of DPI hardware if possible.
Contact

**Email** philwint@kau.se

**OpenPGP** 2A9F 5FBF 714D 42A9 F82C
0FEB 268C D15D 2D08 1E16

**Twitter** @__phw

**Thanks to** Anonymous reviewers
Arturo Filastò
Simone Fischer-Hübner
George Kadianakis
Karsten Loesing
Tobias Pulls
Runa Sandvik