Grady Johnson, Nat Meysenburg, Chris Ritzo, and James Vasile

Getting Through the Block: Do Circumvention Tools Really Work?

Not everyone has the freedom to speak openly on the Internet. Many people turn to the use of tools that offer them hidden, protected, or obscured access to the Internet, opening up content that might otherwise be blocked or providing connectivity to blocked services. However, no one has identified a way to measure how well these "circumvention tools" work, when and why they might not work, and which tool to choose. Depending on which tool is selected and how it works, the simple act of attempting to use it could put someone at risk of being discovered.

Creating a common framework for testing circumvention tools is vital in supporting users to understand the risks, and rewards, of circumvention tools. In 2014-2015, with a small prototype grant from the Knight Foundation, OTI and the Open Internet Tools Project (OpenITP) began work on a common testing framework, dubbed the Network Interference Test Environment (NITE) project. This framework prototyped gathering data on where the tools work, where they do not, and the particular network conditions during successful or unsuccessful attempts to use them, both with or without user intervention. This project involved the prototyping of a set of automated, virtualized testing tools to detect and characterize Internet interference.

Although some resources exist about different circumvention tools, there is no information about when, why, and how they fail. Many of these tools are not tested systematically in the countries where they are meant to be used. While users of these products provide ad-hoc reports on their efficacy when used in locations around the world, and organizations such as APC, GISWatch, Freedom House, and others regularly compile information about the most dangerous countries in which to practice freedom of expression, this information becomes quickly outdated as the adversaries who engage in network censorship and interference continually work to thwart the technical interventions that circumvention tools provide.

Assessing the efficacy of circumvention tools in a particular environment upon request becomes an intensive process, one that requires quickly learning as much possible about the user's technical network environment and the sociopolitical conditions in their location. Providing this service in challenging conditions requires acquiring network access in-country and running single-point or ad-hoc tests. Single-point tests rarely reveal trustworthy information about the network environment. Ad-hoc tests only provide isolated data points at specific moments in time. Because tests are both ad-hoc and manual, they are somewhat akin to anecdotes, and are not systematic enough to analyze in a rigorous way. Practitioners who do this work regularly often supplement these anecdotes with data points gleaned from Twitter and anti-censorship mailing lists. Again, this combined set of anecdotal data is not enough to help users make an informed decision when selecting a tool.

The NITE project and prototype provided an opportunity to analyze and understand the many challenges that face interference measurement projects, beginning with the fact that categories of censorship vary from country to country. As OTI wrapped up the prototype project last year and took a step back to assess the lessons learned, it is clear that developing such a testing framework remains critical to helping users assess the risks of using selecting a particular circumvention tool. For many users of these tools, the risk of exposure by using these tools remains, at best, a question mark.

Our Proposed Solution and Prototype

Not everyone has the freedom to speak openly on the Internet. Many people turn to the use of tools that offer them hidden, protected, or obscured access to the Internet, opening up content that might otherwise be blocked or providing connectivity to blocked services. However, no one has identified a way to measure how well these "circumvention tools" work, when and why they might not work, and which tool to choose. Depending on which tool is selected and how it works, the simple act of attempting to use it could put someone at risk of being discovered.

Everyone in the circumvention field is in the same position of over-relying on anecdotal, ad-hoc, manual tests to provide a best guess on how these tools work, and, as such, it is difficult to give users reliable information in critical situations. Without systematic data capture over time, we never see the bigger picture. We are left without the ability to answer the most essential technical question users always ask us: "Which tool should I use?"

We set out to investigate whether an automated, virtualized testing tool could be designed to collect data on when and where tools work. Virtualized tools would allow them to be tested continuously in as many places as possible, with a goal of being able to answer essential tool-choice questions and back the answer up with real data. If the virtualized environment could be prototyped and tested successfully, we envisioned a network of testing end-points in countries susceptible to censorship. These end-points would run virtualized versions of circumvention tools and attempt to access a range of material, from the innocuous to the likely-censored. They could record and report the results of those attempts and thus produce a mapping of both restrictions and the ability of various tools to evade censorship.

Under a grant from the Knight Foundation, we were able to begin work on prototyping the virtualized environment, called the Network Interference Test Environment (NITE) where the circumvention tools could be run.

Debirf - A Virtual Environment for Testing

Because of the critical need for usability, we decided to decided base the images on Debirf – short for "DEBian in Init Ram Filesystem". While primarily used for creating rescue images that run only in RAM, Debirf provides a solid basis for creating a self-contained testing suite. Debirf allowed us to control the tools contained in the virtual environment. This control, in turn, allowed us to automate configuration options, so that the environment would be identical across all tests.

Similarly, using a live Debian image allowed us to avoid re-tooling the software for different operating systems and architectures. By using a widely supported virtual machine format (.iso files), the NITE test should be able to run on many platforms, and the user experience on most consumer devices should not differ widely.

Choosing Debirf also had the added advantage of a strong open source developer community. It was easy (and extraordinarily helpful) to liaise with the principal developers in overcoming development issues.

Design Considerations

Usability

Widespread adoption would be critical to the success of a functional product, and providing a suitably low barrier for entry was identified as an important design consideration. NITE uses a simple graphical user interface, with a few configuration options provided for flexibility. However all tests are designed to run with a single click or command as the only user input.

Since one of the principal challenges for circumvention projects is user error, NITE requires zero configuration. This feature was achieved by packaging the software as a self-contained operating system. Users boot the software directly on their device, or load it into a virtual machine on any major operating system (Linux, Mac, and Windows). Since the project is based on the Debian distribution of Linux, it is largely hardware agnostic, and can be made to run on a wide variety of devices, including Raspberry Pis and other low cost hardware.

Modularity and Extensibility

Much of NITE's usefulness stems from measuring a range of circumvention tools. As such, it was important that the code be both modular and extensible, in order to simplify the process of adding new tools in the future. The simple user interface was designed to easily accommodate additional tests. Tests can be run individually or in sequence.

Authenticity

One challenge identified in the early planning stages was authenticity. In order for test results to be useful, we must know that they came from the same source. Since the goal of some circumvention tools is to obscure the location of the source, it is vital to know that test results are coming in from a single instance of NITE. As a result, every copy of NITE is built with its own unique ssh encryption key. The public key for each image can be correlated at the time of test result collection. This measure has the added benefit of simplifying the secure transport of test results (via an encrypted ssh tunnel).

Intelligibility

In order to keep the results both actionable for users and meaningful for researchers, a simple pass/fail metric is used. Without prior knowledge of the specific environment in which a test is deployed, the heterogeneity of networks – even within the same country or the same service provider – poses significant challenges to deeper network analysis. Thus a simple pass/fail metric is deemed preferable.

With a sufficiently large data set researchers can begin to make reasonable inferences about the working status of tools in a given country or region. Comparing results over time, it should also

be possible to discern broad changes in network filtering practices with regards to circumvention tools.

Prototype Tool Support - Tor and Tor Bridges

During the research phase, we considered several circumvention tools for the prototype, but ultimately settled on testing Tor and Tor Bridges. The decision to use Tor stemmed from several factors:

- a very large user base, relative to other projects
- a robust and detailed set of documentation
- well-maintained Debian packages
- simplicity

We ultimately chose Tor for the prototype from the desire to limit project infrastructure. Unlike other tools, Tor is able to "run out of the box" and establish a connection to the Tor network with minimal configuration. Given the timeline and budget of the project, setting up additional infrastructure (such as a chat server or multiple dummy clients) was out of scope. Lastly, Tor bridges are also under-studied relative to the components of Tor, and are vital to Tor's most censored users.

Since the public Tor relays are blocked in many countries which practice censorship, it was necessary to also test Tor bridges– a semi-public list of relays which are distributed on demand by the Tor Project. Unfortunately, there is no simple or desirable mechanism for automatically obtaining a bridge – the recommended method is to request them by email via a Yahoo, Hotmail or Gmail account – so NITE users must input bridge IP addresses manually if the small existing list of bridges no longer works.

Lessons Learned

Once we understood the problem and began to build the prototype, the design process began to reveal the project's hidden stumbling blocks. Those obstacles ranged from tiny and easily surmounted to large and difficult to fix. We list some of the larger ones below.

First, we were aware that censorship cannot be neatly categorized globally, or even within a country. Every country has national-level censorship, filtering, and surveillance. Most countries also have local policies that strengthen or contradict national policies. Network policies differ by locality and ISP (Internet Service Provider). We knew this going into the project, but underestimated the difficulty of addressing this problem. Our initial tests quickly showed us that network policy varies more than we expected and navigating those variances proved difficult. Locale doesn't just affect policy at the regional level. In some places (e.g. parts of China), censorship is granular enough that it can differ for each building on a block. Even though we anticipated this, it did not prepare us for how often it happens. Ensuring that we test enough endpoints to account for such granular censorship is an impossible task. With the current design, we can, at best, detect the granularity but not fully map it.

Second, we quickly discovered that usefully virtualizing even open source tools in a testable environment is more difficult than expected. We initially estimated we would need two different

virtualization techniques on our platform. We discovered we would need at least three techniques on at least two platforms. In the meantime, we were able to test a subset of tools, which does not yield the comprehensive view needed to truly answer the question of which tools to recommend when and where.

Third, we focused our early work on laptop and desktop computers. We initially ignored mobile for several reasons, for one because our expertise is not in mobile development. This was a mistake. As soon as we began to examine mobile, we realized we needed to pay attention to it earlier in the process.

Desktop users are not a proxy for mobile users. Mobile networks have different technology and policy than wired ISPs. Virtualization technology is radically different between mobile and desktop platforms. Most importantly, the majority of users most at-risk of surveillance are mobile users, many of whom lack access to a non-mobile device. Once we identified the importance of mobile in the project, we began designing for it, although it came late in the process.

Fourth, it is difficult to distinguish between censorship and network outages. We initially thought this was a simple problem, and for many instances of failed tests, it is. However, the difficult cases leave us unable to make definitive statements about the causes of failed tests.

Fifth, the ethical questions are large. There are two kinds of ethical obligations with NITE tests. There are the obligations to users and also obligations to the toolmakers. We owe a duty to the users to provide informed consent for the risks they take with our tests. The benefits of taking the test are very small to the individual user, and much greater to the larger community, but the size of the risk is the inverse. e must take care not to encourage users to run software that will increase their risks in unacceptable or uninformed ways. We believe that this can be navigated via informed consent, but the initial estimate of our ability to truly provide relevant information was overly optimistic. These tests may not be the best way to engage a mass audience, who might use them to unwittingly engage in network activity that puts them at risk. Restricting usage to known and sophisticated partners might be an ethical requirement, even if it hinders adding more testing endpoints.

We also owe the toolmakers not to burden their infrastructure. Adding significant traffic to circumvention networks could degrade service for at-risk users. This was not a problem at our small stage of testing, especially in robust places like the Tor network. Adding more endpoints and more tests, though, could involve putting a perceptible testing load on smaller networks and drawing down on scarce resources. We are aware, for example, of circumvention tools that rely on paid transit of data across certain boundaries. Testing tools in those places would siphon funding from those projects and apply that funding to tests instead of users. We could not ethically make that trade-off without full consultation with the projects in question.

Considerations for Future Development

Scaling

As mentioned under “Design Considerations”, one of the principal requirements for this project was scaling. Unfortunately, it was also one of the principal challenges. Requiring every live image be packaged with its own encryption key forestalls viral distribution – instead, copies of the software would have to be built and disseminated by the project itself and perhaps trusted partners.

One solution to this problem could be a “vouching” system, whereby trusted users can build new copies and sign them with their own key, creating a web of trust – and unfortunately, a traceable social graph. If a key is considered compromised, that copy and any subsequent copies it has signed can be blocked from submitting future test results.

A distinct but related concern for scaling is that the project could become a victim of its own success. As the project scales, it could end up consuming critical resources from other projects, potentially crowding out “real” users.

This concern refers primarily to the Tor network where resources (such as the public relay network and the list of available bridges) are shared among all users. Other tools like Lantern and Psiphon rely on private networks (typically friends and family living abroad and sharing their unfiltered internet connection), and so these resources would likely be operated by the NITE project and its partners – though they would have to be scaled as well.

Censorship

One challenge common to all circumvention tools is that they themselves are often victims of censorship. After all, what good is a filtering system if users have ready access to tools which can mitigate it? The question became how does NITE function once it is popular enough to be noticed by censors?

While the current security model necessitates distribution via human networks and not a public downloads page, much of the value of NITE was in aggregating test results from hundreds of users, and with some frequency.

One solution to this problem was to configure NITE to submit test results to the server via a tool that has been successfully run. For example, if the test client could establish an unfiltered connection through Tor, NITE would submit the results via Tor. If no tools are successful, and the client could not reach the listening server, it could store the results locally and submit them at some later date.

Unfortunately, this method runs the risk of skewing results toward the positive, as tests with zero successes would not be reported to the server. However, given that this result would mean that several tools would all need to be actively (and effectively) blocked, the risk is relatively small.

Conclusions

Our initial work involved specifying a full stack of software and services to run the tests, compile and data, analyze the results, and provide real-time answers to users. We estimated this project would take 18 months to reach a minimum viable product at a cost greater than our modest annual budget.

The Prototype Grant that we received from the Knight Foundation provided a small sum of money to test the most essential assumptions of the project via rapid prototyping. The testing process yielded crucial information about the shape and scope of the problem we were trying to solve. Back at the drawing board, we feel as if we are on version 2 without the cost and pain of making and discarding the first version

The prototyping process revealed the places where we needed to focus resources to make the project succeed. We are currently re-examining the question of how to make this a feasible effort that yields real-time data of benefit to users at greatest risk. Whatever the final result, the prototyping process was of great benefit to making this project a limited success rather than a boondoggle of wasted resources.