2015 IEEE CS Security and Privacy Workshops

Tor Experimentation Tools

Fatemeh Shirazi
TU Darmstadt/KU Leuven
Darmstadt, Germany
fshirazi@cdc.informatik.tu-darmstadt.de

Matthias Goehring
TU Darmstadt
Darmstadt, Germany
de.m.goehring@ieee.org

Claudia Diaz
KU Leuven/iMinds
Leuven, Belgium
claudia.diaz@esat.kuleuven.be

***Abstract*—Tor is the most popular anonymity network, used by more than 2 million daily users. Engineering privacy enhancing tools such as Tor requires extensive experimentation in order to test attacks, evaluate the effects of changes to the Tor software or analyze statistical data on the Tor network. Since research should not be performed on the live Tor network for multiple reasons, various techniques have been employed for Tor research, including small-scale private Tor networks, simulation and emulation. In this paper, we provide an overview and discussion of existing techniques and tools used for Tor experimentation by categorizing techniques and highlighting advantages and limitations of each tool. The goal of this paper is to provide researchers with the necessary information for selecting the optimal Tor research tool depending on their specific requirements and possibilities.**

I. Introduction

Anonymous communication networks (ACNs) are an essential component in online privacy protection, enabling users to communicate or access information anonymously over the Internet. Tor [1] with more than 2 million daily users [2], establishes bi-directional channels that can be used for interactive applications with low-latency constraints, such as web browsing and online chat. The Tor network hides the identity of users by routing all traffic through circuits consisting of multiple relays, which are operated by volunteers and distributed around the world. Tor is designed to ensure that the origin and the destination of a communication flow remain unlinkable as long as an attacker is unable to monitor the relays used to enter and exit the Tor network at the same time [1].

Tor is under active development by the Tor project team and is supported by an active research community assisting in the development, testing, and analyzing the Tor network. Due to the popularity of the Tor network, a variety of people are developing or doing research on and about Tor. Frequent research topics include passive and active attacks on user anonymity, performance improvements and scalability of the Tor network, for example [3], [4], [5]. Privacy engineering on Tor requires to investigate the behavior and the impact of changes to the Tor network. Testing is an important stage in the system or software development process to make sure the design parameters are satisfying the design goal and are feasible and practical in terms of the resources they need. Improving design decisions, such as adjusting system parameters, is an iterative process which introduces the need of a reliable testbed.

Live Tor Research. An obvious approach is to employ the live Tor network for conducting research. Tor is open-source software which makes it very easy for researchers to become part of the network - either as users or as part of the network by running a relay or both. However, research on the live Tor network is discouraged since it induces several severe limitations and potential risks. First, the live Tor network cannot be employed for research on modified versions of the Tor software since these changes would have to be deployed in scale in order to examine their impact. However, software updates require extensive testing to ensure that updates do not weaken the anonymity of Tor users Using modified Tor software on a subset of relays or clients does not produce results that are representative for a wide-scale deployment of the modified software. Experimenting on the live Tor network also raises ethical issues since experiments may influence the performance and functionality of Tor and may endanger users in situations where the failure of Tor's anonymity has severe consequences. Loesing et al. discuss ethical issues that arise from collecting statistical data in Tor [6]. In their case study they attempt to collect data about Tor usage per city as well as traffic exiting the Tor network per port. While collecting this information, they discuss that measuring sensitive data can harm the anonymity of Tor users, in particular if researchers publish collected data. They name guiding principles which should be followed whenever measuring statistical data is necessary including legal requirements, user privacy concerns, ethical approval, informed consent and community acceptance [6]. However, these principles render measuring sensitive statistics impossible for the majority of cases.

Alternatives. Over the last years, several independent research groups presented different approaches to experiment with Tor. While each approach attempts to provide accurate results, some of them focus on specific aspects of the Tor network. As an example, the Tor Path Simulator (TorPS) focuses on the path selection algorithm used to construct circuits [7]. There have also been attempts to provide a general-purpose simulation or emulation framework for Tor, in particular Shadow [8] and ExperimenTor [9], but no single method has proven to be outstanding so far. The existence of multiple possibilities to experiment with Tor leads to research results which are based on specific tools and induces the risk of biased results depending on the choice of the experimentation platform. While each technique might have individual advantages, they also bring limitations and problems.

Depending on the research question, different aspects of the experimentation testbed need to be as accurate as possible. These aspects include the network effects, the Tor network (in terms of the Tor network structure, the routing approach of Tor, and security details such as encryption), the number of users and their behavior, and last but not least the behavior of the adversary. For example, a research process which aims at improving Tor's performance, the testing will depend rather on the accuracy of performance related factors such as round trip time, bandwidth, and jitter. On the other hand, when the

DOI 10.1109/SPW.2015.20

privacy of a system is under investigation and the system parameters need to be adjusted for better privacy, the accuracy of other factors such as AS-level distribution is of high interest.

Relevance. Meanwhile, Tor experimentation tools are mandatory prerequisite concerning privacy research in the Tor network. Analysis of user privacy, data protection and anonymity can only be performed in a safe and realistic experiment in a controllable environment resulting in reproducible results.

Comparing different Tor experimentation tools is challenging due to the inherent characteristics of Tor as an anonymity network. Thus, exact statistics on user behavior or traffic patterns are unavailable as they contribute to the anonymity protection of Tor users. Loesing et al. state, that measuring sensitive statistics in the Tor network endanger the anonymity of Tor users and therefore should follow several guidelines limiting the scope of measured statistics [6]. Consequently, it is hard to tell which tool accurately represents the Tor network since there is no reference model available that could be used to compare experimentation tools with the live Tor network. Instead, approaching an evaluation of experimentation tools requires comparing tools with each other in order to determine commonalities and differences.

Contributions. Our contribution consists of a detailed overview of techniques and tools that have been employed for experimentation with Tor. To do so, we categorize different techniques and define metrics which can be used to compare experimentation tools. We then discuss advantages and limitations of each tool using the defined metrics. Finally, we indicate issues that require future work in this area.

II. Background on Tor

We give here a brief background on Tor; more information can be found at www.torproject.org and in [1].

To communicate anonymously through Tor, users install a local application called onion proxy (OP) which handles connection setup and routing through the Tor network. The OP first downloads a *consensus* from a *directory server*. Consensuses are generated hourly, and comprise a list of available Tor nodes, also known as relays or onion routers (OR), together with their contact details, such as IP address, and additional information like node bandwidth and various node flags. The OP then selects three nodes from the most recent consensus, chosen randomly according to Tor's routing policy. The combination of three nodes is called circuit. The first node is called the *entry* or *guard node*, the second node is the *middle node*, and the last node is the *exit node*. While all routers can potentially act as middle nodes, only some nodes are flagged as guard or exit nodes. Each node's operator decides whether that node should advertise as an exit node; guard nodes, on the other hand, are flagged as guards based on superior bandwidth and mean uptime.

At the time of writing (February 2015), Tor consists of roughly 7000 nodes of which more than 1000 are exit nodes [2]. In the current Tor design each user is assigned a small number of guard nodes that are changed every 30–60 days. Guard flagged nodes were introduced to limit the probability of an adversarial router being selected as entry node [10]. A further development in the routing policy is to disallow a communication to pass through two nodes within the same /16 subnet IP address. (Note that the underlying topology is therefore not a complete graph.) Due to performance considerations, Tor's routing policy does not select each possible route with the same probability; preference is given to high-bandwidth nodes, and the probability that nodes are chosen depends on the ratios of overall guard and exit node bandwidths as well as some additional bandwidth weights aiming at balancing the use of exit and guard flagged nodes. If a node fails to reply within a certain time period during circuit construction, the circuit is dropped and the circuit construction is repeated. In addition, to increase the probability of successful path construction, nodes are preferred based on the ports that have been used by the user in the last hour [11].

CollecTor. The current state of the network is published as consensus by the directory authorities every hour. This document, also called network status document, contains information about all relays currently contributing to the network. An archive of consensus documents aggregated in monthly packages is available at CollecTor [12] dating back to late 2007. In addition to the consensus documents, relays publish server descriptors which contain detailed information for each individual relay. Server descriptors are updated as parameters change and are also available on CollecTor in monthly archives [12]. The combination of consensuses documents and server descriptors allow for the recreation of the network state at a given point in time including all information about relays and their corresponding exit policies, cryptographic keys, software version, and bandwidth. This information is frequently used in context of Tor experimentation.

III. Tor Experimentation

Tor research has employed a multitude of techniques and led to the development of numerous tools. In this section, we will categorize and discuss these techniques and present available tools accordingly. In particular, we will discuss advantages and possibilities, but also indicate problems and limitations.

Categorization. In order to assess different experimentation techniques, research on the Tor network can be disposed in six categories: a) analytical / theoretical modeling, b) private Tor networks, c) distributed overlay network deployments, d) simulation, and e) emulation. Note that the sixth category is live Tor research, however drawbacks and limitations of this approach have already been discussed previously.

Analytical / theoretical modeling is the foundation of every experiment; an abstract model is applied in almost all Tor research projects to a certain extent. Few years ago, most Tor experimentation made use of distributed overlay network services like PlanetLab to deploy testing Tor networks. However, recent research more commonly used simulation or emulation techniques to experiment with Tor since powerful frameworks are now available for Tor. Due to this observation, our main focus is set on simulation and emulation.

In simulation, the approach is to mimic a system in order to obtain similar results; the underlying functionality though maybe simplified and very different from the system that it is simulating. Emulation, on the other hand, tries to mimic the

same functionality to produce similar behavior and results, and hence is generally not using shortcuts to save resources. We address differences between simulation and emulation in detail later in this chapter.

Metrics. In order to be able to make an informed choice for experimenting tool various criteria are of interest. We address the following criteria for Shadow, TorPS, and ExperimentTor: Modeling Tor in terms of

1) *size / number of relays*. This is important because a small network might not capture all the same network effects that a large network might experience and when parameters are adjusted for better privacy the adjustments might not hold for a large network.
2) *routing approach*. The routing approach is influencing directly the achieved anonymity and performance of the network, hence it should ideally be identical to Tor's routing approach;
3) *topology* from both a spectral point of view such as AS-level / geographic distribution, and Tor's bandwidth distribution. This is important because the AS-level distribution highly influences the anonymity of Tor [13]. Moreover, Tor's routing approach is highly influenced by the bandwidth of nodes. Hence, the accuracy of this model influences privacy and performance that the experiment is going to obtain.
4) *network effects* such as congestion. This is important because Tor's main performance drawback is due to high congestion [14].
5) *number of Tor users* and
6) *usage pattern of Tor users*. The number of users and their usage pattern influence the workload and congestion of the network which in term influences performance directly and the privacy indirectly by influencing routing.
7) *Modeling adversaries*. This is important because a major part of research on Tor is about investigating active attacks (note that for passive attacks the adversary does not need to be modeled in the experimentation tool) against the anonymity and performance of Tor.

In addition, we are interested in the following aspects:

8) Whether the experimentation tool is *currently being maintained*, since Tor is constantly updated.
9) Whether the tool is using *unmodified, i.e. original Tor source code*.
10) *Required resources*, such as the minimum number of hosts required for experimentation. This influences the feasibility of the experiments.

Note, that there are certainly more features which could be applied; however we focus the discussion on features relevant to performance, security, and feasibility.

A. Analytical Modeling

A valid model of the Tor network is the foundation of experimentation with Tor since it is required to verify design choices of the experimentation environment. Jansen et al. approached modeling the Tor network in [15] in order to obtain a model which can be used to run experiments. However the AS-level distribution and geographic distribution is not taken into consideration in their modeling. Backes et al. have analyzed the anonymity of Tor's path selection by a building tool for computing anonymity guarantees [16]. However, most research questions cannot be answered with only a theoretical model of a system but require at some extent practical experimentation to verify the model.

B. Private Tor Networks and Chutney

A simple approach other than using the live version of Tor is setting up a private Tor network on physical or virtual machines. Using a private clone of Tor allows for full control and customization of the software, network topology and data collection. Setup can be assisted by Chutney, a tool for configuring and launching a testing Tor network [17]. Chutney generates Tor configuration files for all required components for the standalone Tor network, including directory authorities, relays, bridges and clients and deploys them. It can also be used to monitor the configured network while running experiments on it. However, running a private Tor network requires dedicated resources and will therefore be limited in scale. Even though Chutney can assist in managing the network, experiments do not scale to a large number of nodes. In addition, Chutney does not induce traffic, hence a realistic model of Tor usage is required to accurately simulate network traffic. Finally, Chutney is still in alpha phase and does not implement all promised functionality yet [17].

Chutney is for example used by the developers of the Scalable Network Emulator for Anonymous Communication (SNEAC). They employ a modified version of Chutney to setup the emulated Tor network and configure the nodes to route traffic through the emulator core [18].

C. Testbed Deployments

There are multiple distributed overlay networks available for research and development. Three of them have been used for Tor research: PlanetLab, Deter and Emulab.

1) PlanetLab: PlanetLab is a globally distributed overlay network which allows researchers to run an experiment in a certain share of the network called slice [19]. According to their website, PlanetLab consists of 1335 nodes at 645 sites at the time of writing [1]. Due to the absence of other suitable tools for Tor experimentation, PlanetLab has been widely used in the past for active Tor research. For example, Tang et al. proposed an improved circuit scheduling algorithm and used a PlanetLab deployment for its analysis [20]. In another example, Bauer et al. conducted an experiment on PlanetLab to analyze the impact of Tor's routing optimization on user anonymity [21].

However, using PlanetLab for Tor research has several disadvantages: First, experiments are limited in size to the available resources of PlanetLab, which in addition have to be shared with other researchers. Second, researchers cannot influence the underlying network topology as well as link parameters such as bandwidth, latency and delays. This leads to experimental results which are specific for the current state of the PlanetLab network and hence difficult (if not impossible)

[1] https://www.planet-lab.org

to reproduce [22]. Modeling specific users seems also to be not trivial with Planetlab. However, modeling an active adversary is straight forward for Planetlab.

Akhoondi et al. have proposed an alternative routing algorithm for Tor where the geographic location and the Autonomous System (AS) of each node is taken into consideration in order to avoid circuitous paths and used PlanetLab for evaluating their proposed changes to the Tor routing algorithm [23].

While PlanetLab has often been used for Tor experimentation some time ago, most experimentation in recent time was performed using either the Shadow simulator or the emulator ExperimenTor.

2) Deter: The Cyber Defense Technology Experimental Research Laboratory (DeterLab) is a network emulation testbed for cybersecurity experimentation [24], [25]. In contrast to PlanetLab, researchers can provide a network topology and run arbitrary applications. In Tor research, Deter has been used by Chakravarty et al. to demonstrate a traffic analysis attack on Tor users by injecting bandwidth fluctuations at server side and observing these fluctuations along the circuit [3]. Similar to PlanetLab, experiments on Deter are limited to the available resources and therefore do not scale. Available resources also have to be shared with other researchers. More information about Deter can be found on their website [2].

3) Emulab: Another network emulation testbed is Emulab [3]. The primary installation is maintained at the University of Utah [26] and can be used for research free of charge. Researchers can supply a network topology which is then used to generate an emulated network. Das et al. used Emulab to analyze the effects of selective DoS attacks on user anonymity and presented an algorithm to detect potentially compromised circuits [27]. However, Emulab induces the same limitations in terms of scalability and shared resources as Deter and PlanetLab.

D. Simulation

In simulation, an abstract model of a system is used to simulate the behavior and events of that system. This requires a realistic model of the system as precondition on which all simulation results will depend. Jansen et al. presented a model of the Tor network in [15]. Simulations usually happen in virtual time which reduces the amount of resources required for an experiment since not all components need to run at the same time. Along with reduced hardware requirements follows an improved scalability of experiments. In Tor research, simulation is widely used. Various simulators have been presented, some of them focusing on specific aspects of the Tor ecosystem, depending on the respective research topic. However, the most often used tools is Shadow, a general-purpose network simulator.

1) Shadow: Shadow is a discrete-event simulator based on the Distributed Virtual Network (DVN) simulator [8]. It is designed to run on a single machine with moderate hardware requirements and user privileges. To improve performance, Shadow scales down the number of hosts used in the simulation. Using a consensus and the corresponding server descriptors as input, Shadow generates a XML file containing a reduced set of nodes. This XML file is then used as the simulation blueprint for the virtual network.

The processed shadow host file contains the description of a downscaled set of Tor nodes which is used during the simulation. The developers of Shadow provide some exemplary downscaled networks and also the possibility to generate a downscaled network from any desired consensus. The process of downscaling is challenging since it induces the risk of inaccurately simulating the Tor network. Shadow attempts to mimic Tor bandwidth distribution in the downscaling but does not take spectral distribution into consideration. Wacek et al. have analyzed different routing algorithms proposed for Tor [28]; for their experimentation they have taken the AS-level and geographic distribution of Tor also into account. Jansen et al. [29] have modeled Tor in their work with more accuracy to represent the Tor network also from a geographical point of view. However, the Tor model used in their work is not available online due to it's large size (25GiB).

All aspects of the simulation can be controlled by the simulation script. The script defines the execution of the simulation, i.e. creation of the network, nodes and plug-ins and to associate parameters like latency, bandwidth and CPU speed. During the simulation, the scheduler takes care of executing events from the simulation script and receiving events to be scheduled from applications and the virtual nodes within the simulation [8].

Applications can be included in the simulation as Shadow plugins. A plugin consists of the original application, an application wrapper implementing callback functions for the simulator and a specification of the instance specific application state. Shadow takes care of swapping in and out the application specific state before and after the application is executed. Using this technique, Shadow is able to run multiple instances of the same application with reduced memory demands. To ensure proper simulation, Shadow requires the application to run in a single process and thread as well as being asynchronous, meaning non-blocking to prevent deadlocks. Specific calls to external libraries are intercepted by Shadow and redirected to the simulation framework (using selective function interception). This is used to perform operations within the simulation, for example operations on the virtual network, rather than the physical network interface of the host machine. Along with Shadow, Jansen et al. presented Scallion, a Shadow plugin which consists of the Tor application and an implementation of the required callback functions [8]. When installing Shadow, the latest version of Tor is installed and used by the simulator. Shadow is currently maintained.

However, the scalability of Shadow is limited by the resources available at a single machine, since it is not possible in the current implementation to distribute a simulation on multiple machines. Also, Shadow as a network simulator makes certain assumptions to reduce the complexity of the system. For example, Shadow does not perform cryptographic operations in order to reduce simulation run-time. While abstraction offers improved scalability, it comes at the cost of accuracy since it is difficult to verify that the model used does not disregard certain peculiarities of the system. Finally, Shadow implements an own virtual socket library which is

[2]https://www.deter-project.org
[3]https://www.emulab.net

used instead of the socket library from the operating system of the simulation machine. This adds an extra layer of complexity to the experimentation setup.

Figure 1 presents the correlation between real time and virtual time for simulations of different sizes. The comparison is made for simulations running one virtual hour on a machine with the following processor: Intel(R) Xeon(R) CPU E5-4650 0 @ 2.70GHz. Note that the complexity of the simulation is shown to be non-linear, in particular for a large simulation size.

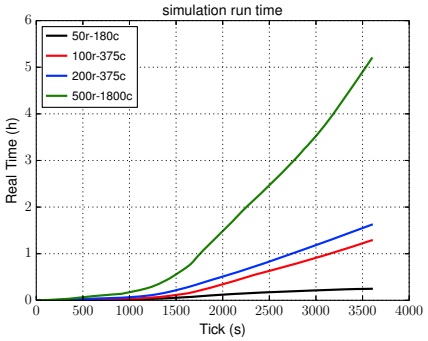

Fig. 1. The figure displays the performance of Shadow for different simulation sizes. It points out the difference between the real time and virtual time, where the comparison is made for simulations running one virtual hour. The virtual time reflects the amount of Tor activity (such as downloads and path constructions).

2) TorPS: Johnson et al. presented the Tor Path Simulator (TorPS), which aims to simulate the process of relay selection for circuit construction [7]. Given one or multiple Tor consensus files and the corresponding server descriptors, TorPS in a first step converts these into a network state file which can be used as input to the simulation. TorPS then simulates the selection of relays for circuit construction based on one of five predefined user models. Output of the simulation is a list of circuits, each consisting of three ordered IP addresses. In addition, TorPS provides an option to run simulation with a congestion aware algorithm proposed by Wang et al. [14]. TorPS is only intended to provide results for the circuit construction process. It is therefore suitable for research on improving or changing the path selection algorithm in Tor, while it can not be used in experiments where actual user workload and traffic measurements are required to be part of the simulation. TorPS models only the usage pattern (used port) of Tor users for path establishment.

TorPS is written in Python and has no automatic way of updating to the newest version of tor, however currently it is maintained to follow the latest path construction algorithm of Tor.

We ran several simulations on TorPS to measure the resources that a TorPS simulation requires. A simulation based on a single consensus simulating 100,000 circuits for a client which connects to Google (74.125.131.105) took around 4 minutes on a machine with the following processor: Six-core AMD Opteron (tm) processor 8435 @ 2.60GHz. On the same processor a simulation where 1,000,000 circuits were established took 40 minutes.

3) Changing of the Guards: In 2012, Elahi et al. presented Changing of the Guards (COGS), a simulation framework designed for evaluating effects of Tors entry guard selection on user privacy [5]. COGS utilizes consensus documents and server descriptors from CollecTor to recreate the Tor network state at a given point in time. Then, COGS runs a large amount of path selection simulations while generating log files. These log files can then be parsed in order to obtain the desired information.

COGS uses original Tor source code (version 0.2.2.33) and adds inline commands to mimic Tor guard selection and path construction. COGS is currently not maintained. There exist some flaws in COGS mimicking Tor's exact routing approach. For example, COGS is ignoring exit nodes 16/ family conflicts.

E. Emulation

Network emulation is another technique to experiment with Tor. In contrast to simulation, emulators perform all operations in real time on virtual nodes, aggregating to an accurate representation of the network. Since all operations have to be performed simultaneously, this results in extensive resource requirements. There are two dedicated Tor emulators: ExperimenTor and SNEAC.

1) ExperimenTor: ExperimenTor is a network emulator intended to improve Tor research. It is based on the ModelNet network emulation testbed [30] and was presented by Bauer et al. in 2011 [9]. ExperimenTor requires at least two physical or virtual machines: While the emulator core is responsible for emulating the network, Tor and other applications like BitTorrent clients run on one or more emulators. and one or more emulator edges. ExperimenTor uses unmodified Tor software and the size of experiments is only limited by available hardware resources. ExperimenTor generates a downscaled network representing Tor in term of bandwidth distribution. The developers have generated a downscaled network with 1,000 nodes using two machines [9].

ExperimenTor also induces a model for Tor clients based on characteristics by McCoy et al. [31]. In contrast to Shadow, experiments do not include background effects of the Internet like non-deterministic jitter or packet loss [15]. Various research projects used ExperimenTor for emulation of the Tor network. As an example, Wacek et al. used ExperimenTor to analyze which relay selection technique provides best anonymity and performance properties [32]. However, ExperimenTor is based on an outdated version of FreeBSD and is therefore no longer maintained. ExperimenTor has been replaced during recent work at the University of Waterloo by SNEAC, the Scalable Network Emulator for Anonymous Communication.

AlSabah et al. [33] and Moore at al. [34] have used ExperimenTor for evaluating their research on Tor. In another work, AlSabah et al.[35] experiment on the live Tor network and complement their research using an ExperimenTor emulation.

2) SNEAC: Sukhbir Singh presented the Scalable Network Emulator for Anonymous Communication (SNEAC) in her master thesis at the University of Waterloo in 2014 [18]. It is an emulator dedicated to anonymity networks and based on modified versions of Mininet and Open vSwitch. SNEAC is in particular designed for large-scale Tor emulation. Similar to

Metric	Shadow	TorPS	ExperimenTor
1. Size / number of relays	downscaling, simulation with 500+ relays possible	no downscaling	limited by available resources
2. Routing approach	not using additional weighting in node selection	ignoring paths being dropped due to timeouts	-
3. Topology	geographic distribution ignored, bandwidth distribution based on Tor	both same as Tor	geographic distribution of Tor ignored, bandwidth distribution based on Tor
4. Network effects (e.g. congestion)	yes	no	yes (simplified)
5. Number of Tor users	downscaled	no	downscaled
6. Usage pattern of Tor users	5 usage patterns	5 usage patterns	2 usage patterns
7. Modeling adversaries	possible	possible	possible
8. Currently being maintained	yes	yes	no
9. Using original Tor code	yes	no, Python application	yes
10. Required resources	single host, user privileges	single host, user privileges	min. 2 hosts, high resource requirements

TABLE I. COMPARING SHADOW, TORPS, AND EXPERIMENTOR USING METRICS RELEVANT TO MEASURING PRIVACY AND PERFORMANCE AND THE FEASIBILITY OF THE EXPERIMENTS.

ExperimenTor, SNEAC requires at least two physical or virtual machines: The emulator core is responsible for network emulation and package routing to and from the edge nodes. Either one or more edge nodes run multiple instances of applications and communicate with each other using TCP/UDP through the emulator core. Since SNEAC is emulating all operations in real time, it has very high hardware requirements. The experiment presented in [18] made use of eight machines, each having at least 1 TB of memory, 80 CPU cores and four 40 Gigabit Ethernet network interfaces with interface bonding. SNEAC takes care of emulating network communication without the need to adapt running applications. While this leads to very high hardware requirements, it guarantees realistic emulation results since all applications including Tor are running in real time without modification. The emulator itself does not impose any limitation on the results of the experiment. Hence, it's the responsibility of the researcher to collect and extract the relevant information during the emulation by measuring relevant parameters. Since SNEAC is rather new, it has not been used for Tor research other than in [18]. Further advantages and limitations will arise once SNEAC will be used for Tor research by the community.

IV. DISCUSSION

To choose the most appropriate experimentation tool, it is important to consider the necessary requirements of the experiment and the available resources and then find a tool that matches this trade-off.

Using simulation or emulation is an important differentiation. In simulation, the goal is to create an accurate model of the system that is about to be simulated. For simplification, the model might take certain assumptions which - ideally - do not change the behavior of the system. Simulation usually happens in virtual time, which improves the flexibility and reduces hardware requirements at the cost of an increased simulation runtime. As an example, Shadow runs as a regular application without root access. While this accounts for improved usability and reduces configuration complexity, it requires Shadow to reimplement major parts of a network stack (e.g. the SOCKS library) which induces the risk of implementation errors. On the other hand, emulation techniques are applied to encapsulate multiple instances in an emulation with as little modifications as possible. Emulation takes place in real time, meaning that all entities of the emulated system need to be running at the same time. Hence, emulation has significantly increased hardware requirements compared to simulation.

Table III-E1 investigates Shadow, TorPS, and ExperimenTor using the metrics defined in Section III. Note that, for example, while TorPS makes use of different user models for selecting appropriate exit relays, network effects caused by user traffic is not taken into consideration. Hence, dropping circuits caused by timeouts due to congestion is not covered by the simulation model. In real Tor some nodes may be rarely connected due to timeouts. Two examples of weaknesses of Shadow are as follows. In the current version of Shadow node selection ignores the bandwidth weights that are given at the end of the consensus to balance off the use of nodes in different position in the relay. Shadow's downscaling of the Tor network is also not ideal, ignoring the geographic distribution and the AS-level distribution of the Tor network [28]. The latter is influencing the anonymity of the downscaled network. However, ExperimenTor on the other hand has worse scalability than Shadow and TorPS and is also ignoring the AS-level distribution when it is downscaling Tor. Moreover, ExperimenTor is based on outdated version of FreeBSD, currently not maintained nor online available. Note that the weaknesses indicated in the table are not an exhaustive list.

V. Summary

In this paper, we discussed Tor experimentation tools, both general-purpose tools like Shadow, ExperimenTor, and SNEAC and specific-purpose tools like TorPS and COGS. Before Shadow and ExperimenTor were available, research would often use deployments on emulation testbeds like Deter, Emulab or PlanetLab for experimentation. Shadow, TorPS, and COGS are simulators while ExperimenTor and SNEAC are emulators. Consequently, hardware requirements of ExperimenTor and SNEAC are much higher than the requirements of the simulators. However, they promise more accurate results. While Shadow and ExperimenTor have been often employed for Tor experimentation, the use of TorPS and COGS has been rather limited. SNEAC has not been used so far since it was just published. Furthermore, we have identified ten metrics that are relevant to measuring anonymity and performance of Tor and we compared Shadow, TorPS, and ExperimenTor based on these metrics.

VI. Future Work

We intend to compare the testing results from various experimentation testbeds in terms of accurate representation of the Tor network. Inspired by [15], it is important to find metrics that can be verified from samples of the live Tor network.

A systematic approach for comparing these tools is to create a simulation blueprint using data from CollecTor [12] and convert it to the required input format for Shadow, TorPS, and SNEAC. Finding metrics for comparing the experimentation results requires more work to be done; some ideas include the comparison of circuits, e.g., by bandwidth, latency or relay usage and the position of relays in circuits. Our work is intended to contribute in standardizing Tor measurements.

Acknowledgements

We would like to thank Rob Jansen for sharing with us information on Shadow's performance and scalability. We would also like to thank Aaron Johnson for his input on the performance of TorPS and the characteristics of circuit construction in COGS. Finally, we appreciate the help of Ian Goldberg, Kevin Bauer, and Sukhbir Singh in context of ExperimenTor and SNEAC.

References

[1] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proceedings of the 13th Conference on USENIX Security Symposium*, USENIX Association, 2004.

[2] The Tor Project, "Tor metrics." https://metrics.torproject.org. Accessed in December 2014.

[3] S. Chakravarty, A. Stavrou, and A. D. Keromytis, "Traffic analysis against low-latency anonymity networks using available bandwidth estimation," in *Proceedings of the European Symposium Research Computer Security - ESORICS'10*, Springer, September 2010.

[4] R. Jansen, F. Tschorsch, A. Johnson, and B. Scheuermann, "The sniper attack: Anonymously deanonymizing and disabling the Tor network," in *Proceedings of the Network and Distributed Security Smposium - NDSS '14*, IEEE, February 2014.

[5] T. Elahi, K. Bauer, M. AlSabah, R. Dingledine, and I. Goldberg, "Changing of the guards: A framework for understanding and improving entry guard selection in tor," in *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2012)*, ACM, October 2012.

[6] K. Loesing, S. J. Murdoch, and R. Dingledine, "A case study on measuring statistical data in the Tor anonymity network," in *Proceedings of the Workshop on Ethics in Computer Security Research (WECSR 2010)*, LNCS, Springer, January 2010.

[7] A. Johnson, C. Wacek, R. Jansen, M. Sherr, and P. Syverson, "Users get routed: Traffic correlation on tor by realistic adversaries," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, CCS '13, ACM, 2013.

[8] R. Jansen and N. Hopper, "Shadow: Running tor in a box for accurate and efficient experimentation.," in *Proceedings of the Network and Distributed System Security Symposium - NDSS'12*, The Internet Society, 2012.

[9] K. Bauer, D. Mccoy, M. Sherr, and D. Grunwald, "Experimentor: A testbed for safe and realistic tor experimentation," in *In: Proceedings of the USENIX Workshop on Cyber Security Experimentation and Test (CSET)*, 2011.

[10] M. K. Wright, M. Adler, B. N. Levine, and C. Shields, "The predecessor attack: An analysis of a threat to anonymous communications systems," *ACM Trans. Inf. Syst. Secur.*, vol. 7, pp. 489–522, Nov. 2004.

[11] The Tor Project, "Tor path specification." https://gitweb.torproject.org/torspec.git/tree/path-spec.txt. Accessed in December 2014.

[12] The Tor Project, "Collector your friendly data-collecting service in the tor network." https://collector.torproject.org/. Accessed in December 2014.

[13] M. Edman and P. Syverson, "As-awareness in tor path selection," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, (New York, NY, USA), pp. 380–389, ACM, 2009.

[14] T. Wang, K. Bauer, C. Forero, and I. Goldberg, "Congestion-aware Path Selection for Tor," in *Proceedings of Financial Cryptography and Data Security (FC'12)*, February 2012.

[15] R. Jansen, K. Bauer, N. Hopper, and R. Dingledine, "Methodically modeling the tor network," in *Proceedings of the USENIX Workshop on Cyber Security Experimentation and Test (CSET 2012)*, August 2012.

[16] M. Backes, A. Kate, S. Meiser, and E. Mohammadi, "(Nothing else) MATor(s): Monitoring the Anonymity of Tor's Path Selection," in *Proceedings of the 21st ACM Conference on Computer and Communications Security (CCS)*, ACM, 2014.

[17] The Tor Project, "Chutney the chutney tool for testing and automating tor network setup." https://gitweb.torproject.org/chutney.git/tree/README. Accessed in December 2014.

[18] S. Singh, "Large-scale emulation of anonymous communication networks," Master's thesis, University of Waterloo, 2014.

[19] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "PlanetLab: An Overlay Testbed for Broad-Coverage Services," *ACM SIGCOMM Computer Communication Review*, vol. 33, July 2003.

[20] C. Tang and I. Goldberg, "An improved algorithm for Tor circuit scheduling," in *Proceedings of the 2010 ACM Conference on Computer and Communications Security (CCS 2010)* (A. D. Keromytis and V. Shmatikov, eds.), ACM, October 2010.

[21] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker, "Low-resource routing attacks against Tor," in *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2007)*, October 2007.

[22] L. Peterson, V. Pai, N. Spring, and A. Bavier, "Using PlanetLab for Network Research: Myths, Realities, and Best Practices," Tech. Rep. PDN–05–028, PlanetLab Consortium, June 2005.

[23] M. Akhoondi, C. Yu, and H. V. Madhyastha, "LASTor: A Low-Latency AS-Aware Tor Client," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, May 2012.

[24] T. Benzel, "The Science of Cyber Security Experimentation: The Deter Project," in *ACSAC* (R. H. Zakon, J. P. McDermott, and M. E. Locasto, eds.), ACM, 2011.

[25] J. Mirkovic, T. B. S. Schwab, J. Wroclawski, T. Faber, and B. Braden, "The DETER Project: Advancing the Science of Cyber Security Experimentation and Test," in *Proceedings of the IEEE Homeland Security Technologies Conference (IEEE HST)*, 2010.

[26] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *Proc. of the Fifth*

Symposium on Operating Systems Design and Implementation, (Boston, MA), USENIX Association, Dec. 2002.

[27] A. Das and N. Borisov, "Securing anonymous communication channels under the selective dos attack," in *Financial Cryptography and Data Security* (A.-R. Sadeghi, ed.), vol. 7859 of *Lecture Notes in Computer Science*, pp. 362–370, Springer Berlin Heidelberg, 2013.

[28] C. Wacek, H. Tan, K. S. Bauer, and M. Sherr, "An empirical evaluation of relay selection in tor," in *20th Annual Network and Distributed System Security Symposium, NDSS 2013, San Diego, California, USA, February 24-27, 2013*, 2013.

[29] R. Jansen, J. Geddes, C. Wacek, M. Sherr, and P. Syverson, "Never been kist: Tor's congestion management blossoms with kernel-informed socket transport," in *23rd USENIX Security Symposium (USENIX Security 14)*, (San Diego, CA), pp. 127–142, USENIX Association, Aug. 2014.

[30] A. Vahdat, K. Yocum, K. Walsh, P. Mahadevan, D. Kostić, J. Chase, and D. Becker, "Scalability and accuracy in a large-scale network emulator," *SIGOPS Oper. Syst. Rev.*, vol. 36, Dec. 2002.

[31] D. McCoy, K. Bauer, D. Grunwald, T. Kohno, and D. Sicker, "Shining light in dark places: Understanding the Tor network," in *Proceedings of the Eighth International Symposium on Privacy Enhancing Technologies (PETS 2008)*, Springer, July 2008.

[32] C. Wacek, H. Tan, K. Bauer, and M. Sherr, "An Empirical Evaluation of Relay Selection in Tor," in *Proceedings of the Network and Distributed System Security Symposium - NDSS'13*, The Internet Society, February 2013.

[33] M. AlSabah, K. Bauer, I. Goldberg, D. Grunwald, D. McCoy, S. Savage, and G. Voelker, "Defenestrator: Throwing out windows in tor," in *Proceedings of the 11th Privacy Enhancing Technologies Symposium (PETS 2011)*, July 2011.

[34] W. B. Moore, C. Wacek, and M. Sherr, "Exploring the potential benefits of expanded rate limiting in tor: Slow and steady wins the race with tortoise," in *Proceedings of the 27th Annual Computer Security Applications Conference*, ACSAC '11, (New York, NY, USA), pp. 207–216, ACM, 2011.

[35] M. Alsabah, K. Bauer, T. Elahi, and I. Goldberg, "The path less travelled: Overcoming tor's bottlenecks with traffic splitting," in *Proceedings of the 13th Privacy Enhancing Technologies Symposium (PETS 2013)*, July 2013.