Traffic Analysis Countermeasures Using Software-Defined Internet Exchanges

R.R. Brooks
Department of Electrical
and Computer Engineering
Clemson University
Clemson, SC
E-mail: rrb@g.clemson.edu

Kuang-Ching Wang
Department of Electrical
and Computer Engineering
Clemson University
Clemson, SC
E-mail: kwang@g.clemson.edu

Lu Yu
Department of Electrical
and Computer Engineering
Clemson University
Clemson, SC
E-mail: lyu@g.clemson.edu

G. Barrineau
Department of Electrical
and Computer Engineering
Clemson University
Clemson, SC
E-mail: cbarrin@g.clemson.edu

Q. Wang
Department of Electrical
and Computer Engineering
Clemson University
Clemson, SC
E-mail: qwsss@g.clemson.edu

Jonathan Oakley
Department of Electrical
and Computer Engineering
Clemson University
Clemson, SC
E-mail: joakley@g.clemson.edu

***Abstract* — The current Internet architecture has a fixed mapping of IP addresses/ranges to services and client organizations. This makes it easy for individuals to hijack sessions, perform traffic analysis, launch denial of service (DoS) attacks, and create man in the middle (MitM) attacks. This paper discusses experimentation using a border gateway protocol (BGP) testbed, a large range of IPV6 space, and software defined networking (SDN) to create software defined Internet exchanges (SDX) that create random mappings between clients and software services. This paper first discusses traffic analysis vulnerabilities inherent in the current approach. It then consider an ideal approach, which removes these problems but is inconsistent with current practice. Finally, the paper concludes by describing a prototype SDX that mitigates current vulnerabilities.**

Keywords* — *Traffic analysis*, *Border gateway prootocol*, *Security*, *Man-in-the-middle*, *Covert communications*, *SDN*, *SDX*, *GENI*, *PEERING

I. Introduction

Today's Internet is a global information ecology that supports financial interactions, industrial research, political discourse, and interpersonal communications. These applications are subject to constant surveillance by industrial spies, local governments, foreign governments, marketing firms, and others [1, 2]. All countries have some level of network traffic monitoring. Many countries use these abilities to hinder legitimate journalism, oppress minorities, and control political interactions. Commercial enterprises throughout the world are subject to unwelcome surveillance by rival, often foreign, economic interests.

In addition to surveillance, network filtering is widespread. The national firewalls of Iran and China are the censorship gold-standards. Both countries routinely perform DNS and IP address blacklist filtering. These filters can block users from accessing large ranges of network addresses if desired. Deep packet inspection (DPI) can block network streams, sometimes by inserting reset packets, if certain keywords are detected.

The most extreme form of filtering is Denial of Service (DoS), where legitimate access to a service is denied. Frequently DoS attacks are done by flooding a site with unwelcome packets. This is commonly done by botnets, which are large collections of surreptitiously hijacked machines controlled by an anonymous bot-herder.

There are a number of common tools for circumventing traffic filters (censorhsip) [3]. These include The Onion router (Tor), Psiphon, and Lantern. These tools usually maintain a set of intermediate nodes whose IP addresses are not freely available, encrypt trafficpassing through the network, and obfuscate traffic patterns by using the intermediate nodes. For Tor, this practice requires coordinating a global network of almost 7000 nodes[1], where each packet is sent through at least 3 different computer nodes and encrypted/decrypted at least 3 times. Advanced traffic obfuscation approaches to foil traffic monitoring have been integrated into Tor as *pluggable transports*. This process is an inefficient use of network resources; introduces large latencies into connections; and has poor Quality of Service[2].

This paper uses SDN concepts to hinder traffic analysis. The goal is to increase user privacy and make the network resistant to filtering, blocking, MitM attacks, and

[1] Measured on 12/23/2015 by https://www.dan.me.uk/tornodes

[2] The lack of QoS is tied to introducing excessive jitter into the network connection.

978-1-5386-9456-5/18/$31.00 ©2018 IEEE

DoS attacks. The rest of the paper is organized as follows. Our problem statement is given in Section II. To adress this problem, we create a network with SDX intermediaries. Our SDX architecture is outlined in Section III. To test these concepts, we use testbed facilities described in Section IV. Results from our experiments are given in Section V. We end the paper with a discussion of our conclusions in Section VI.

II. Problem Statement

Today's IP networks are vulnerable to traffic analysis. Personal privacy and proprietary information are also vulnerable to the analysis of DNS queries and IP access patterns. For example, network sniffing of DNS queries leaving an industrial research laboratory reveals in depth information about proprietary work. While encryption and virtual private networks provide some protection, packet size [4, 5] and timing side-channels [6, 7, 8, 9] easily identify the web sites accessed and/or protocols used. Many of these approaches work even when anonymization proxies are used [7, 10, 11]. The underlying vulnerability exploited by traffic analysis is that IP network sessions are continuous streams of packets between two known addresses written in clear text in packet headers.

The approach shown in Fig. 1 removes the underlying architecture vulnerability that lets IP addresses be exploited to detect classes of network traffic. Instead, our approach uses seemingly random IP addresses for communications sessions. IPv6's vast number of unused addresses helps make this approach particularly flexible. Secure DNS techniques can bootstrap this addressing scheme by securely distributing functions used for address "hopping." Traffic to these "ephemeral" addresses are routed to the desired destination using BGP route injection, intermediate software defined Internet exchanges (SDX), or a combination of the two. Communication sessions can "scatter" their traffic, "customize" their traffic forwarding across Internet, or include other techniques to hide communications patterns to simultaneously minimize both the likelihood of being "tracked" and session latency.

SDX creates new degrees of freedom for end applications to put in place custom, secure forwarding schemes. The notion of SDX was first described in 2013 by Feamster et al. [12] as a software defined Internet exchange, applying software defined networking (SDN) at Internet exchange points (IXP) to enable finer grain, application specific peering beyond what BGP is capable of. Since then, different applications, and variations, of SDX have been proposed. In [13], SDX was defined as a software defined networking exchange that supports interconnection of multiple network domains and services via signaling among federated network controllers. At the 2014 NSF workshop [14], the concept was extended to be an exchange point for software defined infrastructure (SDI) facilities, with the exchange itself capable of both SDN interconnection and injecting computation services into the network path. We consider SDX as an SDI facility that 1) interconnects distinct IP networks and 2) supports network function virtualization (NFV) services among the interconnected networks.

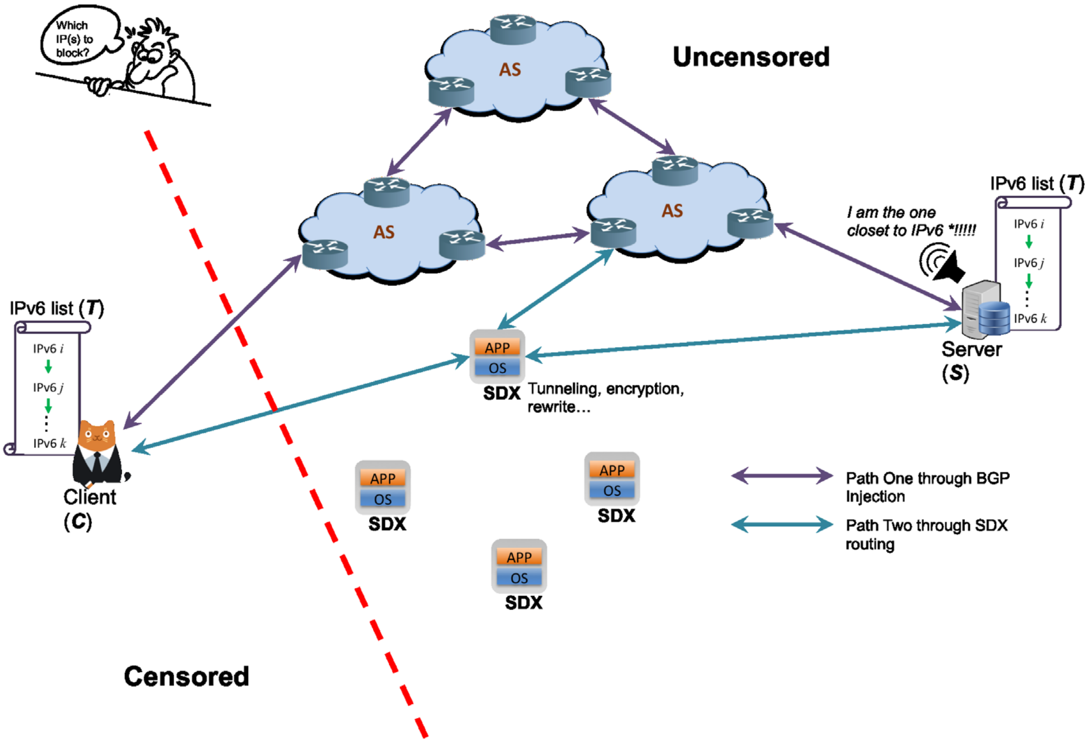


Fig. 1. Users split their communications with server *S* among multiple, uncorrelated IPv6 addresses. Use of SDX and/or BGP injections routes the packets to *S* in a manner opaque to the attacker. The resulting system is Traffic Analysis Resistant Networking (TARN)

Our goal is to allow client node *C* to access a server *S* in a different autonomous system (AS), in a manner that does not allow attacker *A* to determine whether or not communications exist between *C* and *S*. We assume *A* can observe traffic patterns to, from, or within the AS's of either *C* or *S* but not both[1].

There is a large range of solutions possible with different implications. For example, a pseudo-random number generator with a chosen seed could generate a different set of *N* addresses for each time period *T*. For *C* to know the pseudo-random number generator and seed, this information may be encoded and encrypted into *S*'s DNS entry, which only a properly authenticated client *C* can access. Address collisions can occur, while the the vast IPv6 address space makes this unlikely. Collisions can be addressed either by the end hosts (*C* and *S*) or by SDX. We envision eventually allowing all nodes, both *C* and *S* in this example, to use randomized addresses. We also consider the case where one end resides in a legacy IP network where such addressing schemes are either infeasible or put the client at risk. We do not preclude the use of out-of-band communications for providing address generation schemes[2]. The product of this research is a Traffic Analysis Resistant Networking (TARN) system.

III. Architecture

The core idea behind TARN is announcing IP prefixes for a given network or service dynamically, enabling clients to access services using random IP addresses [15, 16]. This IP address "hopping" is similar to frequency-hopping spread spectrum (FHSS) [17] for wireless communications, which has been used to solve similar issues [18].

Network censorship circumvention is an illustrative use case for TARN. A client wishes to access the New York Times, which is blocked by the the client's local ISP. To reach nytimes.com, this client must keep the ISP from detecting and blocking the connection. We assume the ISP is able to monitor all traffic passing through the AS. Nytimes.com could deploy a TARN server[3] to dynamically announce random prefixes for TARN clients to connect to. The TARN client (run on the client's local machine) connects to nytimes.com by using the same logic as nytimes.com's TARN server to calculate the same sequence of randomized IP prefixes. In this way, TARN avoids IP blocking. This is radically different from most existing privacy-preserving technologies, including Tor, that avoid IP blocking either by slowly introducing new IP addresses[4], or by using IP addresses within IP ranges used by important cloud services[5]. The IP prefixes used by TARN exist for a very limited period of time: they are "ephemeral."

Consider the three TARN use-cases in Fig. 2:

1) *Client-based*: TARN, Fig. 3, can be executed by a software agent on the client. This client-based TARN rewrites destination IP address of outgoing traffic to randomized IP addresses within a predefined IP prefix space. BGP announcements are used to route traffic to the TARN server, where the IP address is mapped back to it's original value. TARN uses IP address remapping to forward traffic to it's final destination. The destination IP addresses of the traffic leaving the client has no clear relationship with the desitnation server's IP address.

2) *Gateway-Based*: The gateway solution, Fig. 3, assumes the client's local network is not hostile. An industrial research laboratory may not want to leak information about the URL's visited in researching their projects. In this case, the client's gateway does IP address rewrites. Alternatively, a TARN Content Distribution Network (CDN) could reduce the burden on the TARN server. The TARN server becomes a load balancing router, directing traffic to the most available TARN server. Each TARN server performs equivalent remapping operations.

3) *SDX-Based*: Numerous implementations of SDXs (SDN-based Internet Exchange Point (IXP)) have been proposed as solutions to different problems [12]. We propose a TARN SDX variation that allows a large pool of IP addresses to be mapped into the SDX, with no pre-assigned destinations. Leveraging TARN, we remap IP addresses from that pool back to their original destination. In this way, TARN is run as a service for a wide range of customers, that provides high performance, anonymous channels for customers.

IV. Testbed

We are currently testing TARN prototype implementations. These prototypes use randomized IP addresses for routing. For this new routing approach to work within the legacy Internet, we leverage a number of new network prototyping technologies/services.

A. GENI

GENI [19] is a network prototyping environment used to simulate network topologies and complex geographical configurations. The GENI infrastructure leverages virtual machines (VMs) and SDN to support the flexible creation of large-scale networking topologies. While GENI serves

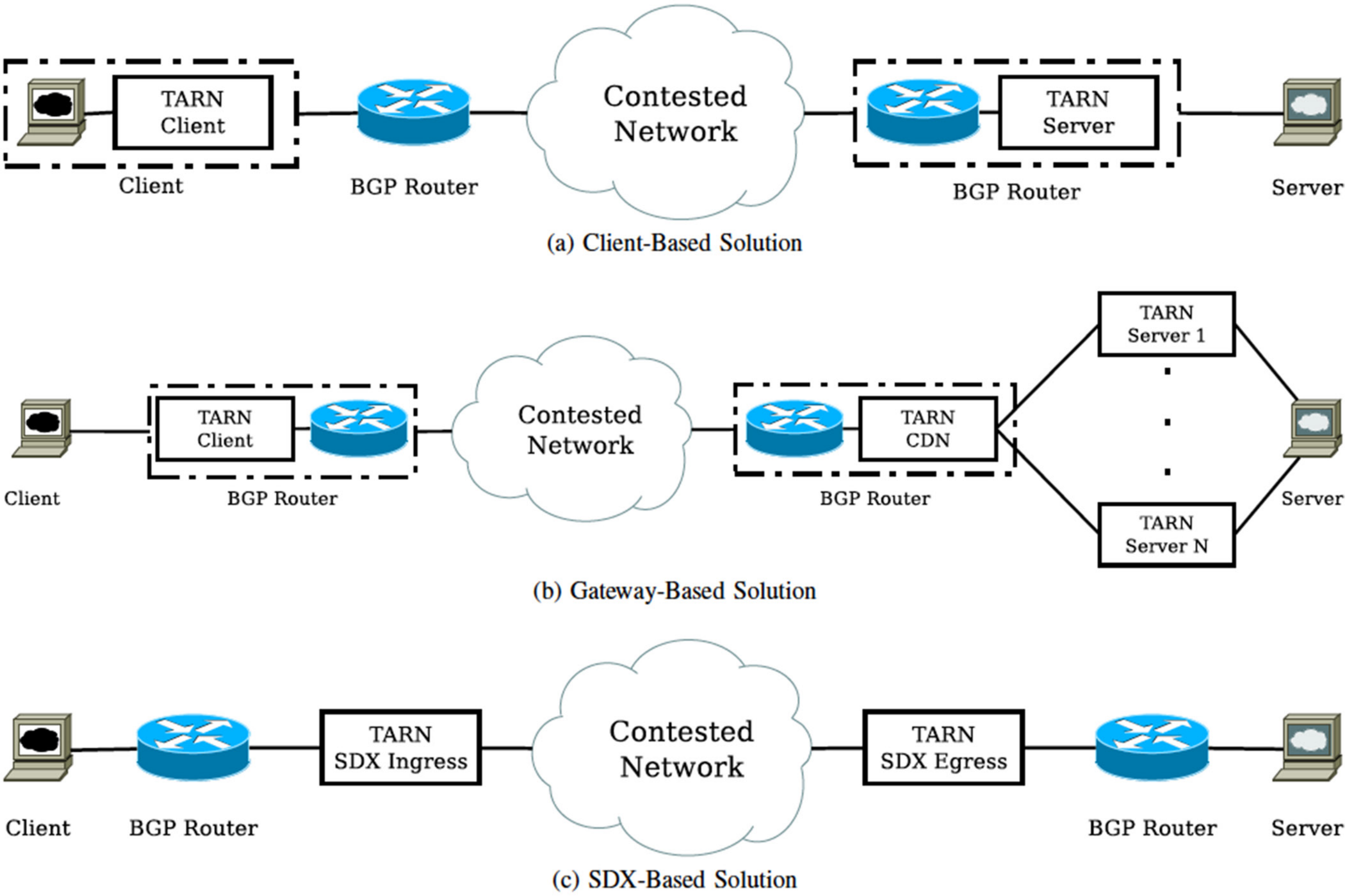
[1] While the proposed approach would make traffic analysis more difficult for an adversary with the ability to monitor network traffic globally, that problem is outside of the scope of this work. We consider this problem outside of our scope, because it creates an unrealistically strong adversary that is not representative of current threats. We note that the anonymity approaches proposed to date, with the possible exception of I2P, are not designed to resist a global adversary.

[2] For example, our user group of democracy advocates and journalists working in parts of Africa with limited freedom of expression could be personally given a prototype version of this system for accessing our proxy system being maintained by Internet without Borders at their annual reunions.

[3] This is analgous to its current https://www.nytimes3xbfgragh.onion darkweb address.

[4] https://www.torproject.org/docs/bridges

[5] https://blog.torproject.org/domain-fronting-critical-open-web



(a) Client-Based Solution

(b) Gateway-Based Solution

(c) SDX-Based Solution

Fig. 2. Implementation strategies

as the frame for the testbed, PEERING [20] provides the core functionality. PEERING is a testbed that includes Internet BGP routers.

B. PEERING

The PEERING testbed provides BGP presence at IXPs around the world. In this specific experiment, the PEERING points of presence (PoP) were Amsterdam (client) and Seattle (server). These PoPs were chosen for their proximity to the GENI aggregates used in the experiment.

We were given the 184.164.240.0/22 prefix range to use in our experiment, giving us a total of four class C prefixes. Our experimental setup is illustrated in Fig. 3, where a Clemson GENI client communicates with a server located at a GENI rack located at Stnaford via TARN. On the client side, the prefix 184.164.240.0/24 was used to assign addresses to the local network. Upon announcing that prefix with the client side BIRD router[1], the client became publicly routable. On the server side, the prefix 184.164.243.0/24 was used to assign addresses to the local network. The Regional Internet Registry for Europe (RIPE) [21] has allotted us a range of IPv6 addresses to use in future testing. PEERING is in the process of integrating IPv6 functionality on their testbed, which will make integration easy. Figure 3 is an overview of our testbed.

OpenvSwitch (OVS)[2] Version 2.8.1 was used on both sides of the topology for packet header rewriting during TARN sessions. The current version supports header modification using subnet masks. Clients and servers all contain TARN controllers. The controller is a modified instance of the Floodlight open source controller[3]. Each OVS and Floodlight node was a GENI Xen VM running Ubuntu 16.04.

V. Results

Figure 4 shows one-way IP randomization that can be achieved with TARN. Each pair of nodes connected by a directed edge represents a separate session captured and randomized by TARN. In this experiment, 35 TCP sessions were randomized when a client connected to a popular news site.

Table 1 shows the throughput comparison of our system. We compare the throughput of TARN with the baseline case and find that our system has a slightly higher throughput. We speculate that the OVSes perform worse when the floodlight controller is disconnected (which was our baseline case).

[1] http://bird.network.cz/

[2] http://www.openvswitch.org

[3] http://www.projectfloodlight.org/floodlight/

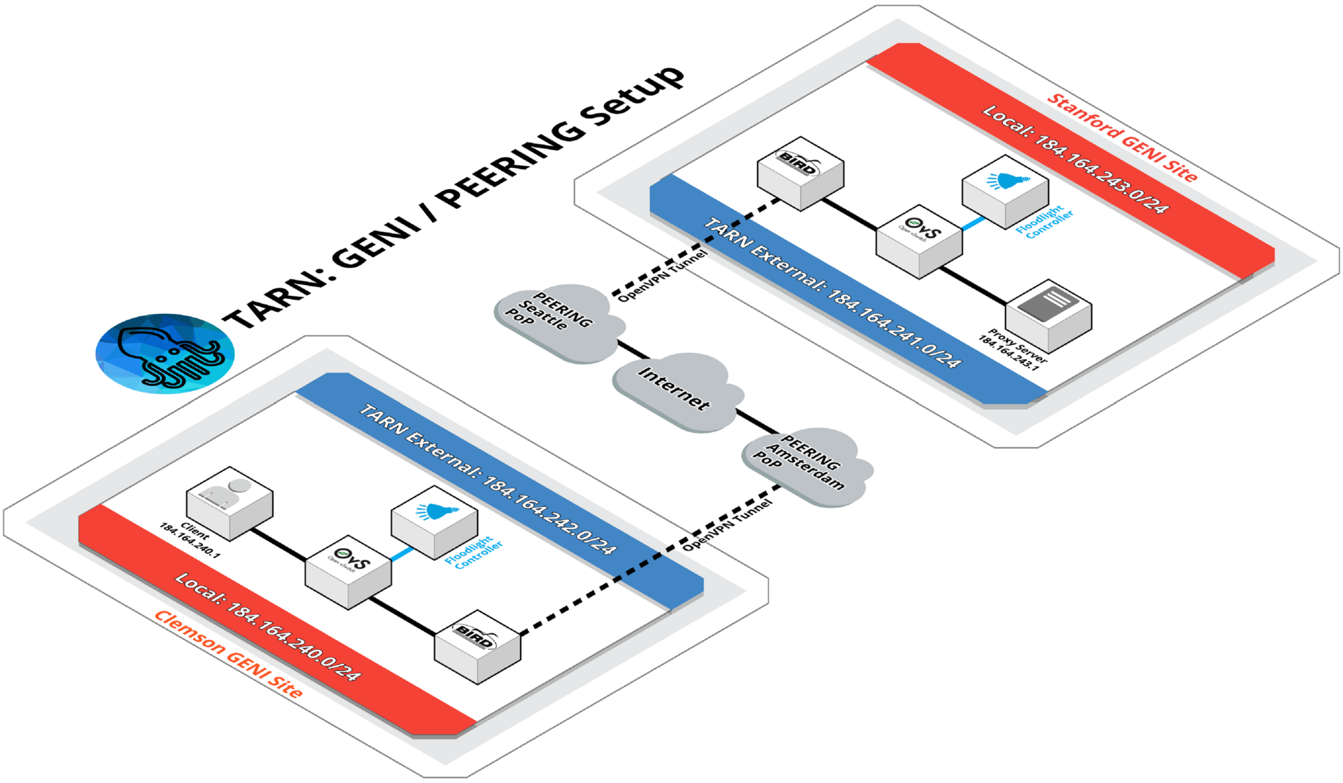


Fig. 3. TARN experimental setup on the GENI testbed

Table 1. Throughput comparison between system with TARN and system without TARN

Method	Throughput (Mb/s)	
	μ	σ^2
TARN	19.1	0
Without TARN	19.0	0.082

VI. Conclusion

TARN provides a novel infrastructure-level approach to circumventing traditional traffic analysis. TARN leverages emerging SDN technologies, GENI and PEERING testbeds which ensure the test conditions are representative of the Internet.

Future work will fully test TARN's throughput limitations and examine the effect of randomizing source *and* destination IP addresses on anonymity. We will also conduct an in-depth security analysis that extends beyond the proof-of-concept we present here and focus on deep packet inspection, side-channel attacks, and a complete statistical comparison between TARN and unprotected traffic.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 1643020 "EAGER: Towards a Traffic Analysis Resistant Internet Architecture." Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. R. J. Deibert, J.G. Palfrey, R. Rohozinski, and J. Zittrain, Access denied: The practice and policy of global internet filtering (information revolution and global politics), 2008.
2. B. Schneier, Data and Goliath: The hidden battles to collect your data and control your world. WW Norton & Company, 2015.
3. R.R. Brooks, L. Yu, Y. Fu, O. Hambolu, J. Gaynard, J. Owono, A. Yepmou, and F. Blanc, "Internet freedom in west africa: technical support for journalists and democracy advocates," Communications of the ACM, vol. 61, no. 5, pp. 72–82, 2018.
4. S. Chen, R. Wang, X. Wang, and K. Zhang, "Side-channel leaks in web applications: A reality today, a challenge tomorrow," in Security and Privacy (SP), 2010 IEEE Symposium on. IEEE, 2010, pp. 191–206.
5. X. Zhong, P. Arunagirinathan, A. Ahmadi, R. Brooks, and G.K. Venayagamoorthy, "Side-channels in electric power synchrophasor network data traffic," in Proceedings of the 10th Annual Cyber and Information Security Research Conference. ACM, 2015, p. 3.
6. X. Zhong, A. Ahmadi, R. Brooks, G.K. Venayagamoorthy, L. Yu, and Y. Fu, "Side channel analysis of multiple pmu data in electric power systems," in Power Systems Conference (PSC), 2015 Clemson University. IEEE, 2015, pp. 1–6.
7. Y. Zhu, X. Fu, B. Graham, R. Bettati, and W. Zhao, "Correlation-based traffic analysis attacks on anonymity networks," Parallel and Distributed Systems, IEEE Transactions on, vol. 21, no. 7, pp. 954–967, 2010.
8. H. Bhanu, "Timing side-channel attacks on ssh," Master's thesis, Clemson University, 2010.
9. Y. Guan, X. Fu, D. Xuan, P.U. Shenoy, R. Bettati, and W. Zhao, "Netcamo: camouflaging network traffic for qos-guaranteed mission critical applications," Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, vol. 31, no. 4, pp. 253–265, 2001.
10. R. Craven, "Traffic analysis of anonymity systems," Master's thesis, Clemson University, 2010.

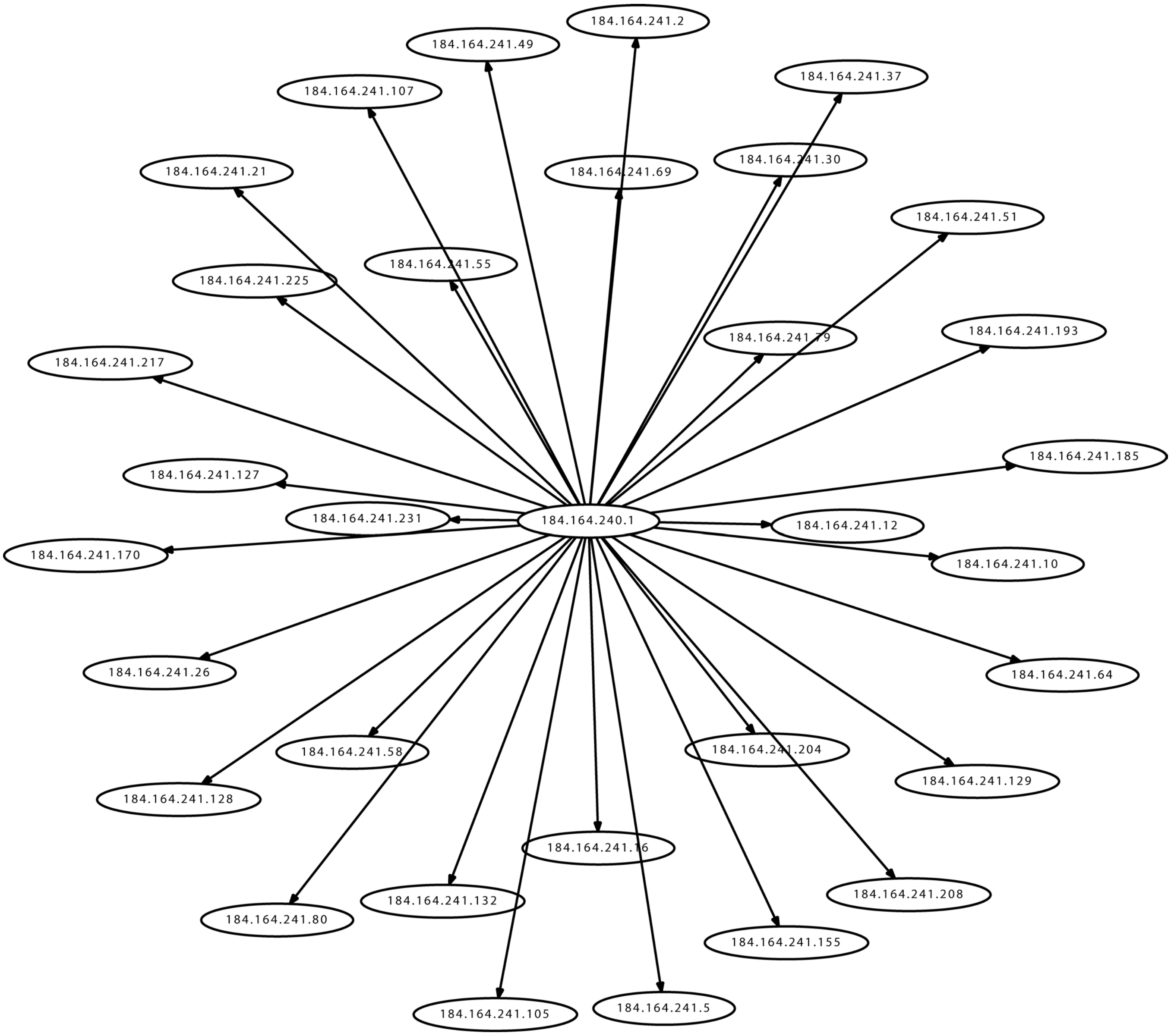


Fig. 4. Distribution of destination IP addresses generated by TARN

11. S.J. Murdoch and G. Danezis, "Low-cost traffic analysis of tor," in Security and Privacy, 2005 IEEE Symposium on. IEEE, 2005, pp. 183–195.
12. N. Feamster, J. Rexford, S. Shenker, R. Clark, R. Hutchins, D. Levin, and J. Bailey, "Sdx: A software defined internet exchange," Open Networking Summit, 2013.
13. J. Mambretti, J. Chen, and F. Yeh, "Software-defined network exchanges (sdxs): Architecture, services, capabilities, and foundation technologies," in Teletraffic Congress (ITC), 2014 26th International. IEEE, 2014, pp. 1–6.
14. "Workshop on prototyping and deploying experimental software defined exchanges (sdxs)," 2014. [Online]. Available: https://www.nitrd.gov/nitrdgroups/images/4/4d/SDX Workshop Proceedings.pdf
15. L. Yu, Q. Wang, G. Barrineau, J. Oakley, R.R. Brooks, and K.C. Wang, "Tarn: A sdn-based traffic analysis resistant network architecture," in 2017 12th International Conference on Malicious and Unwanted Software (MALWARE), Oct 2017, pp. 91–98.
16. K.-C. Wang, R.R. Brooks, G. Barrineau, J. Oakley, L. Yu, and Q. Wang, "Internet security liberated via software defined exchanges," in Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization. ACM, 2018, pp. 19–22.
17. S.M. Schwartz, "Frequency hopping spread spectrum (fhss) vs. direct sequence spread spectrum (dsss) in the broadband wireless access and wlan arenas," white paper, 2001.
18. D. Kahn, "Cryptology and the origins of spread spectrum: Engineers during world war ii developed an unbreakable scrambler to guarantee secure communications between allied leaders; actress hedy lamarr played a role in the technology," IEEE spectrum, vol. 21, no. 9, pp. 70–80, 1984.
19. National Science Foundation. GENI (global environment for network innovations). Http://www.geni.net/.
20. PEERING — the bgp testbed. Https://peering.usc.edu/.
21. RIPE. Ripe network coordination centre. Https://www.ripe.net.