New Selection Algorithm to Secure Tor Connection between Client and Guard

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Siyang Sun

Spring, 2022

Yixin Sun, Department of Computer Science

New Selection Algorithm to Secure Tor Connection between Client and Guard

Siyang Sun
University of Virginia
Charlottesville, Virginia, USA
ss6cuf@virginia.edu

ABSTRACT

The Onion Router (Tor) is a well-known anonymous browsing tool used by whistle-blowers, people living in censored countries, and netizens that values privacy. Tor achieves anonymity by routing client data through three volunteer-run relays[16]. This will hide the client's identity from the final destination server. But when routing between Ases to travel through the volunteer-run relays, these Ases use Border Gateway Protocol which is based on trust causing BGP hijacking[14]. These hijackings could expose the Tor client's identity [15]. Resource Public Key Infrastructure (RPKI) helps AS to stop BGP hijacking. This research project aims to develop a new guard relay selection algorithm for Tor to increase the percentage of Tor clients with a RPKI-covered guard to ensure more protection against BGP hijacking.

1 INTRODUCTION

The 21st century is known as the age of the Internet with 5.16 billion Internet users worldwide[9]. This means a lot of data from internet users traveling around the world through different servers. However, this data is not completely private. Servers could tell the origin of their clients through information such as their IP Addresses. Many internet users, including whistleblowers and people living in censored countries, value their anonymity on the web. These users often use The Onion Router (Tor) to achieve higher anonymity on the internet.

Tor is an anonymous browser tool that is built through an open-source project called the Tor Project. Tor achieves higher anonymity for its client by routing data through three volunteer-run relays, server in the Tor terminology, with encryption to the final destination server[16]. This will ensure that not a single relay knows the whole path from the client to the destination server to avoid losing all anonymity when encountering a malicious relay. Also, the destination server will only see information on the final exit relay but will not know the client's information.

Although Tor has achieved anonymity by hiding client identity from the final destination server, it still has security weaknesses on the Autonomous System (AS) level. When Tor routes data through relays and finally to the destination, the data is routed on the AS level. Routing on the AS level depends on the Border Gateway Protocol (BGP) which is based on trust[14]. Therefore, a malicious AS could falsely advertise a Tor client's IP address to receive data from the client's Tor relay[5]. This hijacking could result in Denial of Service or eavesdropping on the client by passing on the package to the final destination[6].

To eliminate hijacking on AS level, Resource Public Key Infrastructure (RPKI) is developed and deployed on some ASes. RPKI has two components: Route Origin Authorizations (ROA) and Route Origin Validation (ROV). ROA is a way to declare the legitimate owner of an IP prefix and ROV is the validation performed when routing using the information in ROA[6, 13].

Currently, RPKI is not deployed to all ASes on the web. So only certain relays have RPKI and these relays will have a lower chance of encountering BGP hijacking. The vanilla Tor selects guards and relays based solely on the guards' bandwidth. The guard relay is the first point of contact between the Tor client and the Tor relays. This means if the guard relay is compromised, then the client's identity is exposed. This research project aims to develop a new guard selection algorithm for Tor to increase the percentage of clients with RPKI-covered guard.

2 ROA AND ROV MEASUREMENT DATA

There needs to be a set of control data on RPKI before attempting to increase its coverage on the Tor network. Tor network's ROA and ROV coverages were measured so comparisons could be made with the results from different relay selection algorithms. The necessary data for relays, including IP address and prefix, were obtained from Tor Metric's Network Status Consensus [3]. Each relay found in the consensus file was transformed into a Python relay object to store its various information. Routeviews Prefix to AS mapping data is used to find the ASN of a relay based on the relay's IP address[2].

To check for ROA coverage, the relays were assigned a list of their ROA data from josephine repo including ASN and IP prefixes[11]. The relay's prefix and ASN were compared with that of the ROA data. A valid ROA would need to have the same ASN as the relay and the prefix length announced by the relay must be smaller than the maximum length specified by the ROA[10]. The following table represents the ROA coverage for all relays in Figure 1 and guards only in Figure 2.

Figure 1 shows the ROA coverage for Tor all IPv4 relays at different snapshots of the network in an eight-month interval. The blue curve shows the percentage of guards that have a valid ROA: having the same ASN and less specific prefix than the max length attribute in ROA. The percentage of ROA that is valid out of all ROAs is shown using the red curve. The green curve shows valid ROA with a tighter restriction: max length ROA parameter equals the prefix length of the relay, which will protect the relay from sub prefix BGP hijacking[12]. The percentage of protected relay bandwidth is shown in the orange curve. Figure 2 shows the same information but for guards only.

Figure 3 shows the reasons for invalid ROAs in the 7 snapshots of the TOR network in 6-month duration. AS number mismatch is the most common reason for invalid ROA, where the AS of the relay is not among the allowed relay to announce such an IP prefix. The second most common reason for invalid ROA is Max Length

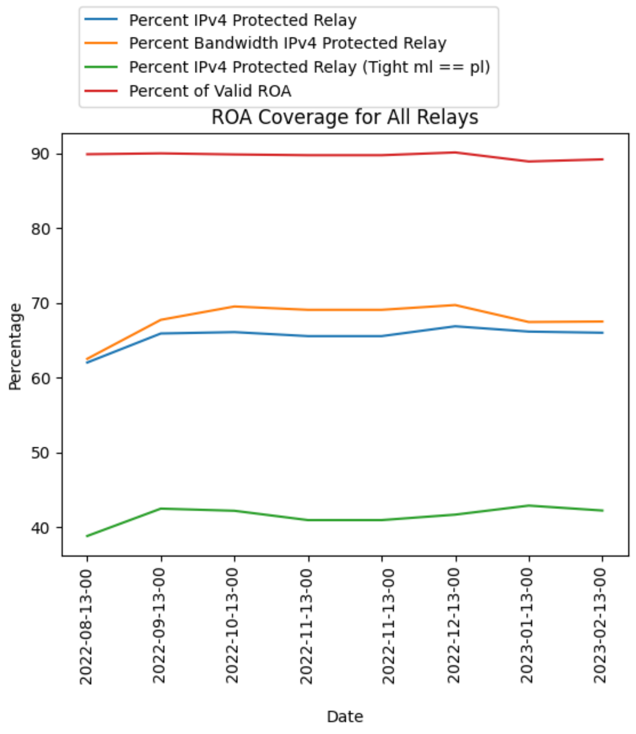

Figure 1: ROA Coverage and Validity for All Relays

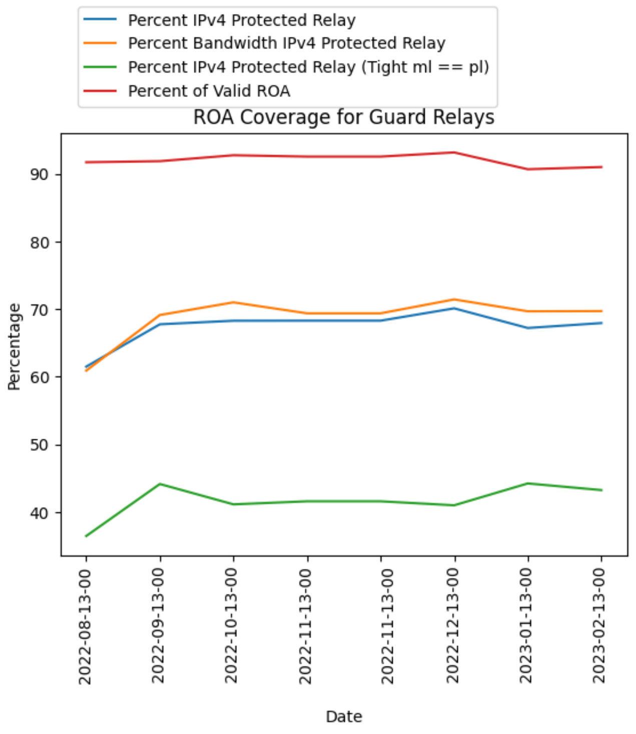

Figure 2: ROA Coverage and Validity for Guard Relays

Violation where the AS announces a more specific prefix than the ROA allows it to. Lastly, the two previous errors combined only have less than 1% of all ROAs.

3 DISCOUNT SELECTION ALGORITHM

The Discount Selection Algorithm aims to increase the number of Tor clients with guard covered by ROA. ROA benefits clients regardless of the client's ROA or ROV coverage. ROV is done by AS while sending data. The sending AS looks at all the AS declaring the target IP prefix and uses ROV to decide which is the legit owner of the IP prefix. Guards with ROA coverage could avoid BGP hijacking when the route from client to guard has an ROV enforcing AS to verify the true owner of the guard's IP Address.

3.1 Implementation and Methodology

In vanilla Tor, guards are selected at random with bandwidth as weights. To take ROA coverage into account, the Discount Algorithm simply reduces the weight of non-ROA guards by multiplying a number $d, 0 < d < 1$, onto the guard's original weight. In this way, clients will have a higher chance of selecting a ROA-enforcing guard because the non-ROA-enforcing guards have a lower weight from the discount.

To simulate the performance of this selection algorithm, I have modified the Python guard selection simulator written by Abby Glaubit[8]. This simulation first creates a list of Client objects that have, similar to the TOR client, a list of guards to choose from. Within the simulation duration, the simulator reads in hourly Tor

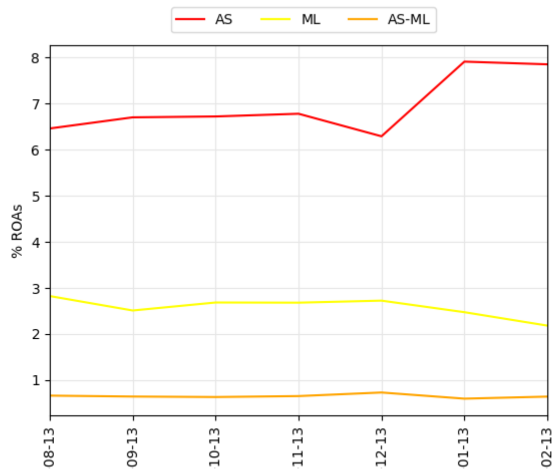

Figure 3: Reasons for Invalid ROA

consensus file, then each client object would add more guards sourcing from the consensus file if its guard list is below the required minimum and remove guards if the guards are expired according to the consensus[3].

To simulate vanilla TOR, the add guard function within the client object selects purely based on the bandwidth and attaches a different weight based on exit or guard status. The discount algorithm will add a ROA coverage attribute to the guard object. The ROA coverage status is obtained using the same technique as described in section 2. Each time a client attempts to add a guard to its list, the weight of each non-ROA guard will be multiplied by d, the discount value. In this way, the client will have a higher chance of selecting a ROA-covered guard.

3.2 Results And Analysis

Figure 4 below, shows the percentage of clients with ROA covered guard at different discount values when using the Discount Selection algorithm. The coverage only increased slightly, around 5%, at $d = 90\%$. At $d = 50\%$, the ROA coverage increased by 15% compared to vanilla TOR in most months, reaching 85% in selective months. This is very high coverage, considering only around 70% of relays have ROA coverage according to Figure 2. However, this increase in ROA coverage for clients comes with a price in the form of worsened load balance compared to Vanilla Tor.

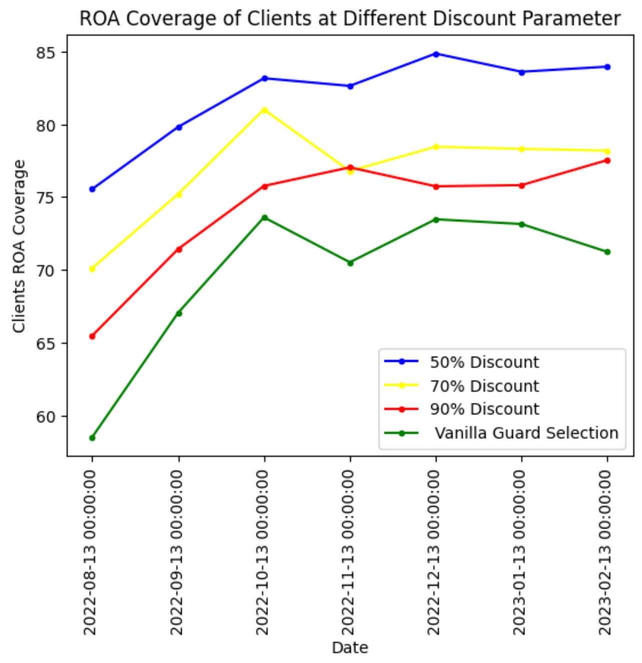

Figure 4: Percentage of Clients with ROA Covered Guard at Different Dicount Value

Figure 5, shows the expected load balance relative to Vanilla Tor, calculated by

$$\frac{\sum VanillaGuardWeight * d}{\sum VanillaGuardWeight * NetworkLoad}$$

at different network load values. We could see that the discount algorithm only affects the expected load balance when the Tor network load is 90% and above. In other words, when all Tor clients occupy less than 90% of the advertised bandwidth of all relays, the Discount Algorithm will not have a poorer load balance compared to Vanilla. Even at full network load which is 1, the expected value is still more than 86% when the $d = 50\%$, which results in around 80% of all clients being covered by ROA guard.

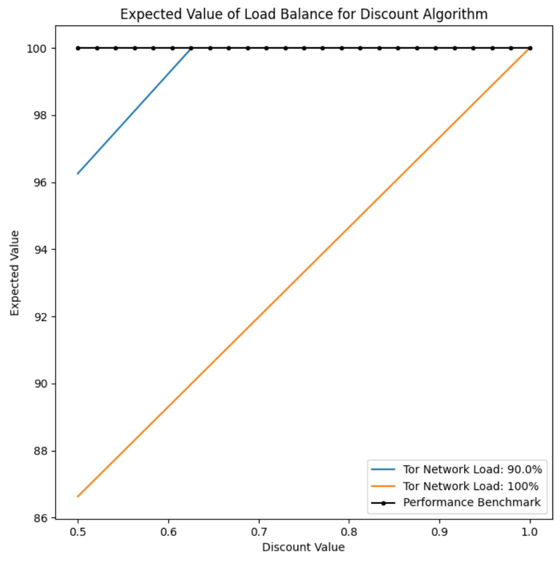

Figure 5: Expected Load Balance of Discount Algorith at Different Network Load and Discount Value

4 MATCHING SELECTION ALGORITHM

The above-mentioned Discount Selection Algorithm aims to increase the percentage of clients that are connected to a ROA covered guard. This approach depends on ROV performed by ASes en route to the client to avoid BGP hijack. There is no way to ensure a secure connection since the network environment is very dynamic and there are multiple ways to route the data, so there cannot always be a ROV enforcing AS along the way. The Matching Algorithm focuses on increasing the percentage of guard-client pairs that have ROA on one side and ROV on the other. This type of guard-client pair is termed ROA ROV matches in this experiment. In this way, the ROA ROV match connection is always secure since the guard-client pair is static for months in a row. ROA on one side will always be checked by ROV on the other side to avoid BGP hijack.

Current ROV coverage among Tor clients and guards is lower than ROA coverage. Therefore, the maximum number of secure connections is upper bound by the total number of ROV enforcing guards and clients. The discount selection algorithm will assign different weights to guards depending on the client's ROA and ROV coverage.

4.1 Implementation and Methodology

All Tor clients are assigned to 3 RPKI coverage classes: ROA covered, ROV covered, and non-covered. Clients with both ROA and ROV coverage are placed in the ROV-covered class since the number of ROV-covered clients or guards dictates the upper bound of the number of secure connections. The Matching Algorithm assigns different weights to guards depending on the client's coverage situation to encourage more ROA ROV matches. In the ROV-covered

Table 1: Matching Guard Selection Strategy

Client RPKI Coverage Status	Selection Strategy
ROV-Covered	discount non-ROA guards
ROA-covered	discount non-ROV guards
Non-Covered	use vanilla weights

class, guards without ROA coverage will be discounted so the ROV-covered client will have a higher chance of selecting a ROA-covered guard. In the ROA-covered class, guards without ROV coverage will be discounted, so the client will have a higher chance of selecting a ROV enforcing guard resulting in a secure connection with ROA on the client side and a guard that performs ROV. Any client could benefit from a ROA-covered guard. Since ASes along the way might implement ROV, resulting in a secure connection. Thus, in the non-covered class, clients are given the vanilla weight for guard selection, so clients without RKI coverage will have the same chances of selecting a ROA guard as in vanilla TOR. Table 1 has a summary of the above strategies.

In the implementation, the Matching Algorithm takes in 3 parameters - ROV discount, ROA discount, and non-discount, which all range from 0 to 1. These 3 parameters are used to discount certain guards' weight in the three respective client coverage classes mentioned above. For example, if the client is in the ROV-covered class, all non-ROA guards' weight will be multiplied by the ROV discount value decreasing their chance of being selected.

To simulate the Matching algorithm, the previous Python simulator for the discount algorithm is modified to fit this purpose[8]. The simulator still uses the same process of reading in hourly consensus and changing the client's guard list accordingly. Since the matching algorithm decides the guards' weight based on the client's ROA and ROV coverage, the client object would need to include an ASN and IP address.

To assign ASN and IP addresses to clients, the clients are first assigned to a country based on a list of the Top 10 countries by relay users from TOR Metrics and caida Inferred AS to Organization Mapping Dataset[4, 7]. Then, each client is assigned to an AS within that country. The weight is decided by the number of IPv4 addresses each AS announces. This data is obtained by tallying the IP prefix announced by each AS using the Routeviews prefix to AS map[2]. Finally, each client is assigned an IP address with the AS randomly.

Also, the add guard function within the simulator is changed so that different discounts are applied to guards based on the client's RPKI coverage. ROA coverage is decided using the same method as described in section 2 and ROV coverage is checked using data from ROV Deployment Monitor[1]. The certainty columns in ROV Deployment Monitor are ignored so that any relay and client within one of the AS listed in the ROV Deployment Monitor are treated as ROV enforcing.

4.2 Parameter Optimization

The Matching Selection Algorithm discounts guards based on the clients' coverage, but the specific discount value is not decided. Intuitively, a lower discount value will cause poor load balance by directing more traffic to ROA and ROV-covered guards that have a heavier weight. Conversely, a higher discount value will have a lower percentage of secure connection, but a better load balance. To decide the optimal value, the simulator needs to run using all permutations of parameters to find the best trade-off between load balance and ROA ROV matches.

Table 2: Mean of Client Objects' RPKI Coverage After 10 Runs

ROV-Covered	ROA-Covered	ROA and ROV-Covered	non-Covered
11	275	7	707

Table 3: Performance of Vanilla Selection Method

% ROA ROV Matching	Expected Load Balance	%Client with ROA Covered Guard
2.06	275	7

Since the clients are assigned ASN randomly using the method above, each time the simulation runs the clients' ROA/ROV coverage will be slightly different due to the randomness of the distribution. This will affect the percentage of secure connections since changes in ROV coverage will change the overall percentage of secure connections. So a static client set is produced by making 1000 clients repeatedly for 10 times and recording the ROV, and ROA coverage. Then, 1000 client objects are made using the average RPKI coverage from the 10 runs. Table 2 shows the client coverage situation after taking the mean of 10 runs.

With this static client set, the results will be comparable between different parameters. The simulator ran with all different permutations of the ROA and ROV discount values ranging from 0 to 1 with a step of 0.1. The non-discount will be constant being 1 to ensure the same ROA coverage as vanilla TOR for clients with neither ROA nor ROV coverage. The total permutations tally up to 81 different sets of parameters. All results with expected load balance less than 75%, ROA ROV matches less than 3 percent, and percent of ROA covered guards less than 73% are filtered out. These benchmarks are decided using the performance result from vanilla Tor present in Table 3 and the RPKI coverage measurement section . This will eliminate all the parameters that perform worse than the vanilla algorithm. The filtered result is present in this Google Sheets

4.3 Results and Analysis

Similar to the Discount Algorithm, the Matching Algorithm favors certain relays more than others, which causes these relays to be more congested than others compared to vanilla Tor's load balance. Because the matching algorithm favors different relays depending on the client's coverage, the overall load balance is measured by taking the arithmetic mean of the expected load balance from all clients. Each client's expected load balance is calculated using

$$\frac{\sum VanillaGuardWeight * d}{\sum VanillaGuardWeight}$$

.

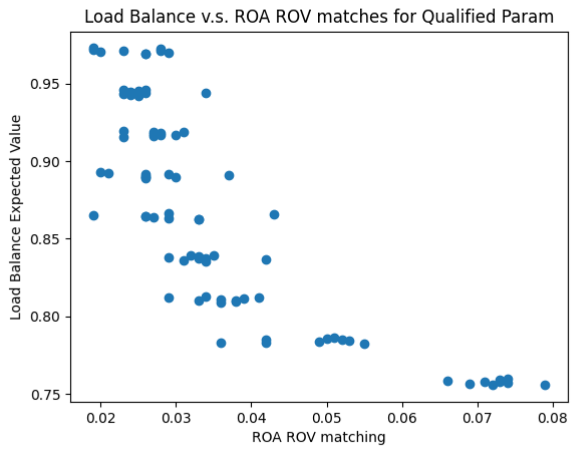

Figure 6: Scatter Plot of the percentage of Secure Connection v.s. Load Balance

Figure 6 shows the relationship between load balance and the percentage of secure connections. The number of secure connections increases more than three fold when the load balance is around 75%. This decrease in load balance would not cause significant congestion for users in network conditions as of early 2023. Since, according to Figure 7, the bandwidth taken by clients barely reaches half of the advertised bandwidth. Thus, at 75% load balance in early 2023 network conditions, Tor clients would not experience more delay than vanilla Tor due to poor relay load balance.

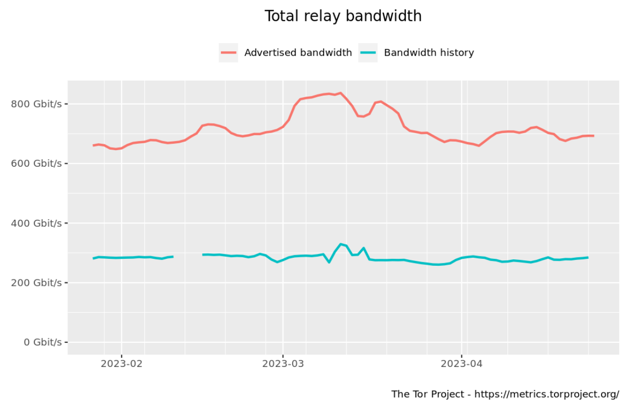

Figure 7: Tor Network's Relay Bandwidth

Figure 8 shows the security performance - percentage ROA ROV matches - in a half a year time interval. The red line is the ROA ROV matched for vanilla Tor. The more conservative parameter, having a load balance of 94%, beats vanilla Tor in ROA ROV matches at most months except 2. The more aggressive parameter almost triples ROA ROV matches at all months, while still maintaining a 75% load balance compared to vanilla Tor.

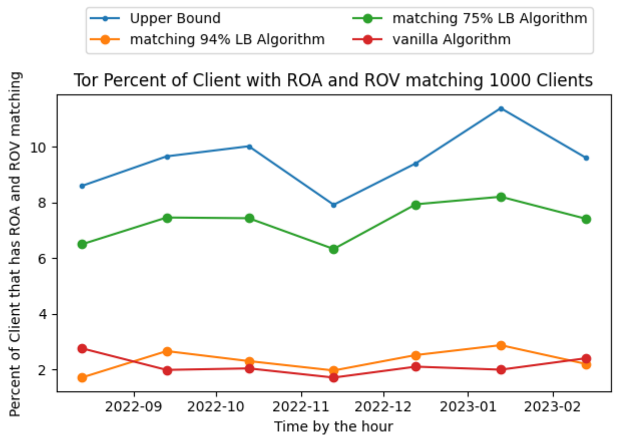

Figure 8: Matching Algorithm Security Performance

5 DISCUSSION AND FUTURE WORKS

The Discount Selection Algorithm successfully increased the percentage of clients with ROA-covered guards. This means that Tor clients have a higher chance of avoiding BGP hijacking compared to vanilla. But this is not guaranteed as there is no way to ensure ASes en route perform ROV. The loss of load balance in this algorithm will not affect the client's usage in most cases since the Tor network's load is almost always around 50% according to Figure 6. As more relay implements ROA, Discount Algorithm's effect on load balance will be lower. But, as ROA coverage rises among guards, there will be less of a difference between vanilla and Discount Selection Algorithms. Clients will most likely have a ROA-covered guard even under vanilla due to the abundance of ROA-covered guards.

Future works on this Discount Selection Algorithm could focus on dynamically adjusting the discount value based on the network load data. This will optimize the parameter of the Discount Algorithm in real-time and provide the best performance. There needs to be further research into if there is a delay between guard selection and frequent usage of Tor in users. So clients are not assigned to the few RPKI-covered relays during low traffic periods, then during a high traffic period, the RPKI-covered relays will be very crowded.

The Matching Selection Algorithm successfully increases the percentage of ROA ROV matches between guards and clients. These matching connections mean at least one direction of connection between the client and its guard is secure and without BGP hijacking, whether from client to guard or vice versa. This Matching Algorithm will benefit those clients with RPKI coverage. These ROA or ROV-covered clients have a better chance of a guaranteed secure connection. Clients without ROA or ROV coverage will have the same chance of selecting a ROA-covered guard, thus they are not worse off compared to vanilla.

Future works in the Matching Selection Algorithm could focus on automatically selecting an optimal parameter each month. Because of the dynamic nature of the web and client base, this algorithm could perform better if it is frequently tuned. There needs to be a more objective way of selecting the best algorithm among the top few that have trade offs between load balance and security performances. Both algorithms, if implemented in Tor, could show users their own and their guard's RPKI coverage. This will allow Tor users to be more conscious of the security environment they are in. In this way, users in sensitive jobs or regions could plan their web activities accordingly.

REFERENCES

[1] Measuring rpki route origin validation deployment. Available at https://rov.rpki.net/.
[2] routing/routeviews-prefix2as. Available at https://publicdata.caida.org/datasets/routing/routeviews-prefix2as/.
[3] Tormetric1. Available at https://metrics.torproject.org/collector/archive/relay-descriptors/consensuses/.
[4] Users. Available at https://metrics.torproject.org/userstats-relay-table.html.
[5] What is bgp hijacking? | cloudflare.
[6] Internet registry roa and rpki, Oct 2018.
[7] Inferred as to organization mapping dataset, Oct 2019. Available at https://www.caida.org/catalog/datasets/as-organizations/.
[8] aglaubit. Available at https://github.com/aglaubit/tor-rpki.
[9] Published by Ani Petrosyan and Apr 3. Internet and social media users in the world 2023, Apr 2023.
[10] Guillermo Cicileo. What is an incorrect roa and where can i verify my information?
[11] josephine. Available at http://josephine.sobornost.net/josephine.sobornost.net/rpkidata/.
[12] Pavlos Sermpezis, Vasileios Kotronis, Petros Gigis, Xenofontas Dimitropoulos, Danilo Cicalese, Alistair King, and Alberto Dainotti. Artemis: Neutralizing bgp hijacking within a minute. *IEEE/ACM Transactions on Networking*, 26(6):2471–2486, 2018.
[13] Internet Society. What is route origin validation?, Oct 2020.
[14] Internet Society. Securing border gateway protocol (bgp), Sep 2021.
[15] Yixin Sun, Anne Edmundson, Nick Feamster, Mung Chiang, and Prateek Mittal. Counter-raptor: Safeguarding tor against active routing attacks. *2017 IEEE Symposium on Security and Privacy (SP)*, 2017.
[16] torproject. About tor browser.