Limiting the Power of RPKI Authorities

Kris Shrishak
TU Darmstadt

Haya Shulman
Fraunhofer SIT

ABSTRACT

Although Resource Public Key Infrastructure (RPKI) is critical for securing the inter-domain routing, one of the arguments hindering its adoption is the significant power that it provides to the Regional Internet Registries (RIRs), allowing prefix takedowns. In this work, we propose a small change to RPKI to distribute the power of RIRs preventing any single one of them from taking down a prefix. We design and implement a distributed RPKI system that relies on threshold signatures. This ensures that any change to the RPKI certificates requires a joint action by a number of RIRs, avoiding unilateral IP address takedowns. We evaluate the performance of our design and use historic RPKI data to analyse its performance and efficiency.

CCS CONCEPTS

• **Networks** → **Routing protocols**; • **Security and privacy** → **Privacy-preserving protocols**.

ACM Reference Format:
Kris Shrishak and Haya Shulman. 2020. Limiting the Power of RPKI Authorities. In *Applied Networking Research Workshop (ANRW '20), July 27–30, 2020, Online (Meetecho), Spain.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3404868.3406674

1 INTRODUCTION

Resource Public Key Infrastructure (RPKI) [17] secures the interdomain routing against prefix and subprefix hijacks and is a prerequisite for path-end validation [5]. RPKI binds the IP address blocks with its Autonomous Systems (ASes) via cryptographic signatures, stored in Route Origin Authorizations (ROAs). ASes then apply Route Origin Validation (ROV) to identify and discard BGP announcements that contradict the information in ROAs. This enables to filter misconfigured as well as malicious announcements attempting to hijack prefixes. Commonly, ROAs are maintained in repositories operated by the five Regional Internet Registries (RIRs): RIPE NCC, LACNIC, ARIN, APNIC, AFRINIC [1].

Despite the significant benefits of RPKI, its deployment is progressing slowly [10, 13, 14]. One of the arguments against deployment of RPKI is the exposure to IP prefix takedowns by forcing RIRs to modify information in the ROAs [6]. Accidental or deliberate changes to the RPKI can cause a prefix of the affected AS to become unreachable for ASes that enforce ROV. RIRs in RPKI have the power to revoke and change any certificate that they have issued and, thus, invalidating BGP routes to the affected AS.

These concerns are not unfounded. In 2011, the Dutch police ordered RIPE NCC—the RIR for Europe, the Middle East and parts of Central Asia—to lock registration of four IP address blocks [24]. Although RIPE NCC took the State of Netherlands to court over this order, their case was dismissed [23] and registrations were locked down. In another case, a court in the United States of America issued a writ of attachment on the country-level top-level domain of Iran, Syria and North Korea and the associated IP addresses [15]. Such demands from the court are possible due to the centralization of power and the hierarchical structure of systems.

In this work, we replace the trust that the ASes have to place in individual RIRs with a mechanism that limits the power of RIRs. We devise a solution based on threshold signatures which limits access to the key material of RPKI and uses multi-party computation (MPC) for signing and revoking resource objects. In Section 2 we compare our proposal to previous research and show why MPC is the most suitable tool for achieving our goal. Our proposal addresses three issues: (1) prevents IP address takedowns, (2) limits the scope and implications of attacks on RIRs, and (3) enables validation in case of missing trust anchor. Our proposal is easy to deploy and does not require changes to BGP and RPKI.

The threat model. RPKI uses hierarchical and centralized authorities who are trusted. That is, attacks against RPKI authorities, malfunction or malicious intent of RPKI authorities and coercion by law enforcement officials are not part of the threat model. A corrupt authority can revoke certificates and ROAs that it has issued.Such a system creates an imbalance of power between the RPKI authorities and organizations lower in the hierarchy. This power imbalance is unlike the distributed nature of BGP. Moreover, the power imbalance with RPKI is greater than Web PKI as only the

[1] Although ASes can host their ROAs themselves, it uncommon.

ANRW '20, July 27–30, 2020, Online (Meetecho), Spain

ACM ISBN 978-1-4503-8039-3/20/07…$15.00
https://doi.org/10.1145/3404868.3406674

specific RPKI authorities can issue certificates to its children in RPKI while in a Web PKI the child has the possibility to request certificates from any of the other CAs.

Furthermore, although the Internet cuts across national boundaries, the design of RPKI opens up the possibility for state actors to perform legalized coarse-grained censorship as court orders are not part of the threat model of RPKI. In fact, RIRs have been presented with court orders in the past [19, 24]. For the purpose of accountability, all the RIRs present numbers on the number of law enforcement requests they receive in a year [20]. The RIRs are bound by the law of the state they are based in, e.g., Dutch law for RIPE NCC and US law for ARIN. However, the members of the RIRs are in numerous countries. What recourse does a member have if it has been affected by a mistake or an attack by the RIR?

RPKI infrastructure was designed because ASes could not be trusted and there were many cases of fat-finger mistakes and attacks that made routing insecure. Nevertheless, RPKI replaces the trust placed on ASes with trust placed on RPKI authorities. Just as ASes couldn't be trusted, *what if the trust placed in RPKI authorities is also misplaced?*

Threshold signatures for RPKI. We propose a distributed RPKI system that has threshold signatures at its foundation. Threshold signature is a cryptographic technique where a set of n parties jointly compute a signature on a message such that at least a threshold $t+1$ parties, with $t < n$, are required to participate. This technique distributes the signing process and is more robust to adversarial attacks or corruptions without changing the original verification procedure of the signature scheme. That is, a threshold signature protocol generates signatures that can be verified as if it were generated using the traditional variant of the signature scheme. Threshold signatures are a specific instance of MPC where signatures are generated in a system with untrusted parties. One such scenario is RPKI, which can benefit from the distribution of trust that threshold signature offers.

RIRs offer *hosted RPKI* service to their members to improve the deployment of RPKI; RIRs host a CA that can be used by its members to issue ROAs. This service makes it convenient for the members of RIRs to use RPKI without the overhead of managing a CA. However, this convenience comes at the cost of further centralization of power as the RIRs also handle the private keys used to sign ROAs.

On the brighter side, this is the kind of scenario where threshold signatures are not only suitable for distribution of trust, but they are also efficient. Threshold signatures are efficient when the number of parties involved is small. There are five RIRs, in different continents, making it suitable to run a distributed RPKI system between them with threshold signature as the engine driving it. As RIRs are at the first level of the RPKI hierarchy, each of them are in a powerful position where they can revoke ROAs of a large subset of the IP-address space. However, RIRs do not usually collude with each other, and often disagree with each other when it comes to their response to law enforcement agencies [19]. Hence, a system that requires a threshold of them to agree makes it harder for any one RIR to act maliciously or be coerced by a state as it is beyond its capability to unilaterally make changes. Our solution not only distributes trust, it also cuts across national legislations.

Our solution can be described as follows: threshold signatures use shares of the private key, where each of the five RIRs will have a share of the private key while none of them have the entire private key. Using only the shares, the RIRs can collaboratively, but not unilaterally, sign ROAs and Certificate Revocation Lists (CRLs). Most importantly, threshold signatures support a stronger threat model where corrupted RPKI authorities are not entirely trusted and yet play a significant role in making BGP secure.

2 RELATED WORK

One approach that has been proposed to address the issue of disproportionate power in the hands of RPKI authorities is to add transparency logs and `.dead` objects to RPKI to note the consent of the Internet Number Resource (INR) owner for revocation [12, 18]. [12] use a detector to identify when a ROA has downgraded from valid to invalid or valid to unknown state and check whether a `.dead` object is present. Three problems with their approach: (1) It requires effort not only from the CAs but also from relying parties. This method depends on relying parties to perform ROV and to alarm other relying parties. In practice, only about 100 ASes perform ROV as can be observed at ROV deployment monitor [2], a monitoring platform [22], while there are 68289 ASes as of 1 April 2020 [3]. (2) It performs detection while we attempt to prevent malicious activities. Attacks can be detected after the fact due to the transparency and the effort of the relying parties. Accountability in the form of malicious activity detected post-mortem is not sufficient as ASes may have already lost out on their business. (3) These `.dead` objects are used to signify consent from the child and they are to be signed by the child node. In the hosted RPKI setting, as the parent manages signing for the child node, the parent can create and sign `.dead` objects by impersonating the child.

Another approach is to replace the existing RPKI system by using blockchains [11]. This approach eliminates the possibility of RPKI authorities revoking previously allocated resource while they remain part of the blockchain by providing new resources. The use of blockchain raises other deployment issues such as consensus algorithm and incentive for the nodes to run the blockchain. If Proof-of-Stake is used as the consensus algorithm, as proposed in [21], then

[2] https://rov.rpki.net

[3] https://www.caida.org/data/as-relationships/

the nodes with greater stake, e.g., large providers who are allocated large subsets of IP addresses will become powerful players, which will create another form of power imbalance. As blockchain-based proposals suffer from scalability issues, RouteChain [25] employs a hierarchy of ASes which are assigned in subgroups to validate BGP announcements on the blockchain. However, in practice, ASes may have conflicting policies that prevent dynamic grouping and, thus, an incentive mechanism is also required.

3 DISTRIBUTED RPKI

3.1 System and Threat Model

In this section, we state the threat model of existing RPKI before introducing the threat model in our system. We introduce different threat models from MPC literature that our solution supports along with the motivation behind each of them. Then, we introduce the system and communication model we use in our work.

3.1.1 Threat Model. In our distributed RPKI (DRPKI) system, we consider a stronger threat model than the existing RPKI system. The existing threat model of RPKI does not consider a malicious RIR. In this work, in addition to the threats considered in the existing system, we do not consider the RIRs to be entirely trustworthy. We distribute trust so that individual RIRs do not need to be completely trusted.

Standard MPC terminology provides us with a tool kit to discuss threat models that not only includes external adversaries but also the participating parties. As there are five RIRs, we consider $n = 5$ for threshold signatures. We consider two scenarios: one where a majority of RIRs are corrupted and another where a minority are corrupted by an adversary. Although the minimum is to prevent unilateral power of individual RIRs, we assume a stronger threat model to prevent colluding RIRs from breaking the security of the system.

The adversary can be passive or active. A passive adversary follows the protocol while an active adversary can behave arbitrarily. Security against passive adversaries is sufficient if the goal is to secure signing keys against internal and external adversaries while trusting the RIRs with operational integrity. When the operational integrity of the RIRs is not guaranteed, security against active adversaries is required.

3.1.2 System and Communication Model. Our system incorporates all parts of the RPKI that requires generating signatures, which includes the creation of signed objects, ROAs, as well as signing of the resource certificates of children and the issuance of CRLs. Unlike traditional RPKI, we propose a DRPKI using MPC such that a ROA cannot be signed by a single RIRs on its own. Our system focuses on the role of RIRs as CAs in RPKI, specifically in the hosted RPKI setting where the RIR that allocates IP resources also runs the CA to validate the ROAs. We focus on the key generation and the signing operation in a distributed RPKI system, such that no RIR has access to the signing key. RIRs have access only to parts of the signing key (known as key shares) and not to the signing key. Thus, RIRs cannot unilaterally sign ROAs of ASes or revoke the associated end-entity certificate.

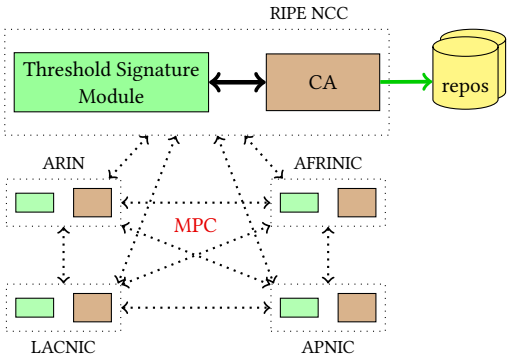

Figure 1: Distributed RPKI architecture

We inherit the communication model from the underlying MPC protocols. More specifically, we assume the existence of synchronous communication network where the protocol is executed in rounds [3, 16]. The communication between the RIRs is implemented and run on a point-to-point network. We assume that the RIRs securely communicate with each other using a TLS connection with authenticated endpoints.

3.2 System Setup

We present the system architecture in this section. Each RIR has has two components: Trust anchor and Hosted RPKI. Each RIR has a CA and a threshold signature module. *RIR CA* is the top-level CA that acts as a trust anchor in the RPKI. RIR CA issues the CA certificates to its members and issues manifests and CRLs for the members. In addition, it also issues a self-signed certificate for itself and a certificate for the Hosted CA. *Hosted CA* is responsible to produce signed objects: ROAs, CRL and manifests for the members. It is available to all members of the RIR who choose to use Hosted RPKI. The public key and the share of the private key of the members is stored at the hosted CA. All the certificates and the signed objects issued by the two CAs are published in public access repositories, through rsync or RRDP [2].

Unlike existing RPKI, our system requires interaction between the RIRs for the creation of certificates and signed objects (Figure 1). RIRs are in different continents and the communication takes place over the public internet using secure and authenticated channels. At a high-level, each RIR has a share of a private key for each member and uses this share to collaboratively issue signed objects. None of the RIRs get to access the entire private key. They use the shares of the private keys to create signed objects. We note that solutions we present in this paper are in the security-with-abort model, that is, the protocols abort when participants are not

Key generation $\mathsf{KGen}(1^\lambda)$

(1) Each RIR takes a security parameter 1^λ as the input and generates a signing key share for the j^{th} member by randomly sampling $[\mathsf{sk}_j] \leftarrow \mathbb{Z}_p$.
(2) Each RIR locally converts $[\mathsf{sk}_j]$ to $[sk_j] \cdot G$.
(3) RIRs compute the public key $\mathsf{pk}_j = \mathsf{Open}([sk_j] \cdot G)) = \mathsf{sk}_j \cdot G$.
(4) Output the secret key shares and the public key $([\mathsf{sk}_j], \mathsf{pk}_j)$.

Figure 2: Key generation protocol

available. However, it is possible to design threshold signature protocols that can function when a subset of parties are unavailable, albeit with modified design parameters.

3.3 DRPKI protocol phases

All phases instigated by a CA and the interaction takes place between the threshold signing modules. MPC adds computation and communication overhead to traditional signing and, hence, we require a protocol that is efficient when the signing is to be performed. Protocols in the preprocessing model generated message independent material apriori and require little computation and communication to complete the signing when the message to be signed is available. Furthermore, most threshold signature protocols only satisfy some of the threat models we consider in our system. As we want to be able to consider the efficiency of our system under all the threat models we discussed in Section 3.1.1, we chose to use the protocol of Dalskov et. al. [8], which is the most efficient protocol that fulfils all our requirements. We describe the protocols in Figures 2 and 3.

Automation. All the steps in our system are automated and does not require human intervention at the RIRs. Note that Step 4 of Online signing phase in Figure 3 requires checking the message before the message is signed. We automate this step through a simple consent mechanism during Step 1. For example, if the customer (AS1) of RIPE-NCC is transferring an IP-space to AS2, then AS1 gives consent to revoke the old ROA and to issue the new ROA pointing to AS2. This consent is sent to all RIRs (instead of only to one RIR) from the customer facing software [4]. Hence, all RIRs are informed of the intent of AS1. When the RIRs run the threshold signature protocol, there is an automatic check for consent. If there is no consent, then the protocol aborts.

3.4 Deployment scenarios

In this part, we propose different deployment models. We present two solutions with their associated trade-offs among the stake holders. We emphasise that the trade-offs are not with respect to the security, but with respect to the responsibilities of the different stake holders.

[4]We need a standalone software to send the consent to all RIRs.

Signing Protocol

Member Independent preprocessing

(1) RIRs generate tuples of secret shared values of the form $([a], [b], [c])$ such that $a, b, c \in \mathbb{Z}_p$ where $c = ab$.
(2) They open the share $[c]$ by running $c \leftarrow \mathsf{Open}([c])$.
(3) Let $[k^{-1}] = [a]$.
(4) Each RIR locally generates $\langle k \rangle = ([b] \cdot G) \cdot c^{-1}$.
(5) Output the initial preprocessing tuple $(\langle k \rangle, [k^{-1}])$.

Member Dependent preprocessing

(1) RIRs input the generated signing key shares $[\mathsf{sk}_j]$ and the initial preprocessing tuple $(\langle k \rangle, [k^{-1}])$.
(2) They compute $[\mathsf{sk}'_j] = [\mathsf{sk}_j / k]$ by generating an additional tuple and Beaver's rerandomization technique [1].
(3) Output the final preprocessing tuple $(\langle k \rangle, [k^{-1}], [\mathsf{sk}'_j])$.

Online signing phase

(1) The member uses a standalone application to give consent, e.g., to transfer IP-space to another AS. The consent is sent to all the RIRs.
(2) Input the message to be signed M and the final preprocessed tuple $(\langle k \rangle, [k^{-1}], [\mathsf{sk}'_j])$.
(3) The RIR initiating the protocol sends the message M to the other RIRs.
(4) The RIRs check the contents of M and the consent by the member before proceeding. If the check fails, they abort $\perp$. Else, they continue.
(5) Then the RIRs compute $R \leftarrow \mathsf{Open}(\langle k \rangle) = (bc^{-1}) \cdot G = a^{-1} \cdot G = k \cdot G$.
(6) Let $(r_x, r_y) \leftarrow R$.
(7) Locally compute the share of the signature $[s] = H(M) \cdot [k^{-1}] + r_x \cdot [\mathsf{sk}'_j]$.
(8) Finally compute the signature $s \leftarrow \mathsf{Open}([s])$ and output $\sigma = (r_x, s)$ or $\sigma = \perp$.

Figure 3: Signing Protocol

Our solutions: Both our solutions distribute the trust anchor (TA). Before discussing our solutions, we give an intuition behind our choice to distribute RPKI trust anchor. The notion of a TA requires all child nodes to unconditionally trust an entity. In RPKI, there are five TAs, one at each RIR, which the relying parties use to verify RPKI signatures. The concentration of power at TAs in the Internet infrastructure extends beyond RPKI and is also observable in DNS(SEC) and Web PKI. However, unlike DNS and Web PKI, there are already five TAs in RPKI that allows for a smooth transition to a distributed TA. Furthermore, the existing system of five

TAs has had its issues. As the policies of each RIR with regards to TA is different, some relying parties do not use the TA of ARIN and ROAs issued under ARIN's trust anchor locator (TAL) fall to the status of 'Not Found' [26]. This means that even when RPKI is implemented, a significant portion of the networks do not validate routes originating from North America due to policy decisions and legal barriers [4, 27]. Thus, in practise large parts of the world are prevented from having better routing security. These issues can be prevented if the TA is not located at individual RIRs with their own policies and is instead distributed across them.

Two-layered deployment: In our first solution, we propose a two-layered approach. The upper layer generates a distributed TA to the five RIRs, while the lower layer uses the threshold signing module for the Hosted CAs. In both layers, the RIRs use our threshold signing module. In the upper layer, a distributed TA is established using our key generation protocol in Figure 2. Each RIR generates their signing key share and participates in the key generation protocol to obtain the public key. Once the public key is obtained, each RIR adds the public key to their TAL as the `subjectPublicKeyInfo` [7]. Each RIR has a TA that has the same public key in the TAL. As no RIR has the private key associated with this certificate, the RIR CAs do not need to be kept offline. Thus, the RIRs do not need a subordinate CA to issue child certificates. Furthermore, as each RIR has the same public key as part of the TA and they have the same `subjectPublicKeyInfo` in their TAL, access to the TAL from one RIR is sufficient for relying parties to validate routes originating from any part of the world that has deployed RPKI.

In the lower layer, our threshold signing module is used by the hosted CAs to generate signed objects such as ROAs. We are able to support delegated CAs as the distributed TA at the RIR CAs is used to generate child certificates. Furthermore, this solution allows for incremental deployment as the LIRs who have already deployed their own CAs can continue to use them to serve their child nodes while those who have not deployed their own CAs can start using hosted CA. Note that the concerns regarding some LIRs being coerced by their country of registration remains.

Flat deployment: In our second solution, instead of having the RIRs run two CAs, RIR CA and the hosted CA, we combine the two so that the RIRs only need to run one CA. Furthermore, we do not need a TA as we replace the top-down architecture with a flat deployment architecture. Not only do we eliminate the hierarchical structure of existing RPKI, we also distribute trust. Moreover, this solution accounts for a stronger threat model where individual RIRs do not need to be completely trusted. However, we do not support delegated CAs in this solution. The CAs only generate end-entity certificates and signed objects; they do not generate any CA certificate that will allow child nodes to generate their own signed objects. This also means that child nodes will need the RIRs to generate signed objects for their child nodes. Nevertheless, we prevent any single entity to be all powerful and require the participation of a threshold number of RIRs for a signed object to be generated and ejected.

Majority \ Adversary power	Honest-but-curious	Malicious
Honest	Shamir	Mal. Shamir
Dishonest	Semi. OT	MASCOT

Table 1: Four MPC protocols

4 IMPLEMENTATION AND EVALUATION

We have implemented our system in C++ and have used MP-SPDZ [9] for the threshold ECDSA MPC protocols. MP-SPDZ includes threshold ECDSA protocol implementations for all the security models that we are concerned with: honest-but-curious and malicious as well as honest and dishonest majority protocols. In particular, we use four protocols—Shamir, Mal. Shamir, Semi. OT and MASCOT—that are shown in Table 1. The former two are based on Shamir secret sharing while the latter two are based on additive secret sharing. We use all four protocols to implement our system.

4.1 Deployment Setups

For performance evaluation, two deployments were set up. For each node, we used an Amazon AWS c5.2xlarge instance with a 64-bit Intel Xeon CPU with 3 GHz and 16 GB RAM. We ran all the evaluations on a single thread. To make our evaluations as realistic as possible, we chose to run the experiments based on the location of the RIRs. The five RIRs are in different continents of the world. So, in the first setting, we run experiments on five Amazon AWS instances that are placed around the world such that they are representative of the location of the RIRs. Specifically, we use the instances at Frankfurt, N.Virginia, Sydney, Sao Paolo and Mumbai while the RIRs are based in Amsterdam, Virginia, Brisbane, Sao Paolo, Mauritius, respectively. Furthermore, we also consider the setting where the RIRs could, in the future, have virtual servers located close to other RIRs. For this purpose, we also run our experiments on the LAN in Frankfurt.

4.2 Experimental evaluations

We benchmark the preprocessing time (member dependent and independent) to generates tuples and the online signing time per signature. As the preprocessing does not depend on the message to be signed, thousands of preprocessed tuples can be generated and stored. They can be used when a new message is to be signed. Although dishonest majority protocols are generally more costly than honest majority protocols, Semi OT has the highest preprocessing throughput in LAN setting (Table 2). Semi. OT protocol uses additive sharing which is cheaper than elliptic curve operations, which is the predominant cost during preprocessing. In the WAN setting,

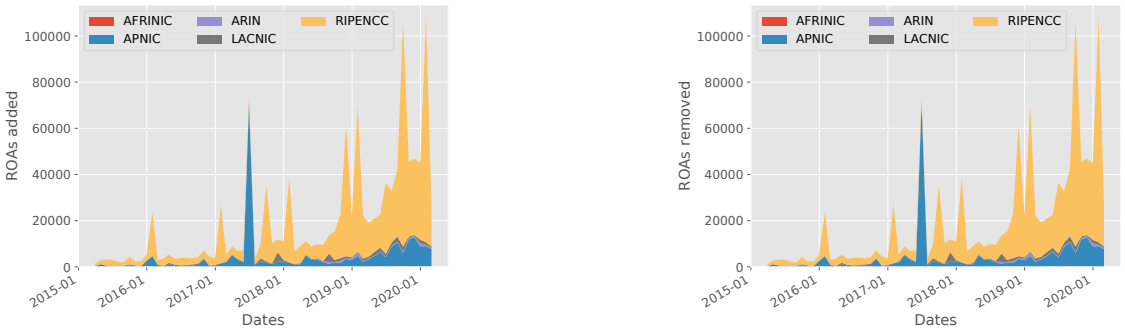

Figure 4: Number of ROAs added and removed from March 2015 to February 2020

	LAN		WAN	
	Preprocessing	Online	Preprocessing	Online
MASCOT	209	529	20	0.95
Semi OT	1042	662	111	2.05
Mal. Shamir	699	714	91	3.53
Shamir	1020	769	265	3.54

Table 2: Breakdown of throughput for preprocessing (tuples/sec) and online phases (signatures/sec).

communication becomes more predominant than local operations. We also observe that the cost of malicious security in the case of honest majority protocol is very small. This is especially true in the WAN setting as the extra checks for Mal. Shamir are local operations while communication becomes the predominant cost. The communication overhead is less than a KByte for online signing phase. MASCOT and Semi. OT require a communication of 624 KByte and 99 KByte respectively for the preprocessing while Shamir and Mal. Shamir require 0.437 KByte and 1.345 KByte respectively.

5 ANALYSIS

Our distributed RPKI system will need to generate numerous signatures for it to be deployable. To understand how many signatures are required if we are to deploy the system, we use historical RPKI data.

RPKI data. Each RIR maintains an rsync repository with RPKI data. We accessed the publicly available historical RPKI data maintained by RIPE NCC that includes the daily archive of the repositories of all the five RIRs [5]. We use the historical data from 1 March 2015 till 19 February 2020 for our analysis.

ROA analysis. We use the RPKI data to analyse the changes, specifically, the number of ROAs that have been added and removed per day in the measured time period. This information allows us to estimate the number of signatures that are to be generated by our distributed RPKI system. Figure 4 shows the monthly change in ROAs for the five RIRs. On average, we need less than 20000 signatures per day. However, there are exceptions, such as on 1 January 2020, when there was a factor 10 increase change in the ROAs. We observe from Table 2 that for our slowest protocol MASCOT, we are able to produce 0.95 signatures/sec or 82080 signatures/day in the WAN setting. For our fastest protocol, we are able to produce 3.54 signatures/sec or 305856 signatures/day in the WAN setting. Note that even our slowest protocol is able to satisfy the requirements on an average day. On days with peaks, our fastest protocols are able to satisfy the requirements. Moreover, the threshold ECDSA signature protocols we use are the first to be designed in the preprocessing model and there is room for optimizations that can assist further in scaling our solution when the adoption of RPKI increases. Note that, in the LAN setting, all our protocols are fast and can produce two orders of magnitude more signatures.

[5] https://ftp.ripe.net/rpki/

6 CONCLUSION

Although RPKI offers significant security benefits, it exposes the Internet to IP prefix takedown attacks. Although there have been proposals to detect takedowns, detecting alone does not eliminate the damage that is done to the affected AS when it goes offline. Furthermore, proposals for detection of such attacks assume that the ASes perform ROV—most of which currently do not. Another problem with the existing proposals is the need for changes to the existing BGP infrastructure. We propose a mechanism for preventing the prefix takedown attacks, without requiring changes to the BGP or RPKI. We implemented our proposal and demonstrate its effectiveness and performance. We note that our solution is purely technical, and legal and policy barriers need to be addressed to make the work truly practical.

ACKNOWLEDGMENT

This work has been co-funded by: the German Federal Ministry of Education and Research and the Hessen State Ministry for Higher Education, Research and Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE; the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation): GRK 2050/251805230 and SFB 1119/236615297.

REFERENCES

[1] Donald Beaver. Efficient multiparty protocols using circuit randomization. In *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 420–432. Springer, 1991.

[2] Tim Bruijnzeels, Oleg Muravskiy, Bryan Weber, and Rob Austein. The RPKI repository delta protocol (RRDP). *RFC*, 8182:1–24, 2017.

[3] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145. IEEE Computer Society, 2001.

[4] Ben Cartwright-Cox. The State of RPKI: Q4 2018, 20 December 2019. https://blog.benjojo.co.uk/post/state-of-rpki-in-2018.

[5] Avichai Cohen, Yossi Gilad, Amir Herzberg, and Michael Schapira. One hop for RPKI, one giant leap for BGP security. In *HotNets*, pages 10:1–10:7. ACM, 2015.

[6] Danny Cooper, Ethan Heilman, Kyle Brogle, Leonid Reyzin, and Sharon Goldberg. On the risk of misbehaving RPKI authorities. In *HotNets*, pages 16:1–16:7. ACM, 2013.

[7] David Cooper, Stefan Santesson, Stephen Farrell, Sharon Boeyen, Russell Housley, and W. Timothy Polk. Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. *RFC*, 5280:1–151, 2008.

[8] Anders P. K. Dalskov, Marcel Keller, Claudio Orlandi, Kris Shrishak, and Haya Shulman. Securing DNSSEC keys via threshold ECDSA from generic MPC. *IACR Cryptology ePrint Archive*, 2019:889, 2019.

[9] Data61. MP-SPDZ - Versatile framework for multi-party computation, 7 June 2019. https://github.com/data61/MP-SPDZ.

[10] Yossi Gilad, Avichai Cohen, Amir Herzberg, Michael Schapira, and Haya Shulman. Are We There Yet? On RPKI's Deployment and Security. In *NDSS*, 2017.

[11] Adiseshu Hari and T. V. Lakshman. The internet blockchain: A distributed, tamper-resistant transaction framework for the internet. In *HotNets*, pages 204–210. ACM, 2016.

[12] Ethan Heilman, Danny Cooper, Leonid Reyzin, and Sharon Goldberg. From the consent of the routed: improving the transparency of the RPKI. In *SIGCOMM*, pages 51–62. ACM, 2014.

[13] Tomas Hlavacek, Italo Cunha, Yossi Gilad, Amir Herzberg, Ethan Katz-Bassett, Michael Schapira, and Haya Shulman. Disco: Sidestepping rpki's deployment barriers. In *NDSS*. The Internet Society, 2020.

[14] Tomas Hlavacek, Amir Herzberg, Haya Shulman, and Michael Waidner. Practical experience: Methodologies for measuring route origin validation. In *DSN*, pages 634–641. IEEE Computer Society, 2018.

[15] ICANN. ICANN Tells U.S. Court That ccTLDs Are Not "Property" | Files Motion to Quash in U.S. Legal Action Aimed at Seizing Top-Level Domains, 30 July 2014. https://www.icann.org/resources/press-material/release-2014-07-30-en.

[16] Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Universally composable synchronous computation. In *TCC*, volume 7785 of *Lecture Notes in Computer Science*, pages 477–498. Springer, 2013.

[17] Matt Lepinski and Stephen T. Kent. An infrastructure to support secure internet routing. *RFC*, 6480:1–24, 2012.

[18] Yaping Liu, Shuo Zhang, Qingyuan Li, and Sufang. Requirement for the transparency of RPKI, 5 November 2019. Work in Progress.

[19] M. Mueller, M. van Eeten, and B. Kuerbis. In important case, ripe-ncc seeks legal clarity on how it responds to foreign court orders, 23 November 2011. https://www.internetgovernance.org/2011/11/23/in-important-case-ripe-ncc-seeks-legal-clarity-on-how-it-responds-to-foreign-court-orders/.

[20] NRO. Handling requests for information by law enforcement authorities, 2018. https://www.nro.net/accountability/rir-accountability/rir-governance-matrix/#lawenforcement.

[21] Jordi Paillisse, Miquel Ferriol, Eric Garcia, Hamid Latif, Carlos Piris, Albert Lopez-Bresco, Brenden Kuerbis, Alberto Rodríguez-Natal, Vina Ermagan, Fabio Maino, and Albert Cabellos. Ipchain: Securing IP prefix allocation and delegation with blockchain. In *iThings/GreenCom/CPSCom/SmartData*, pages 1236–1243. IEEE, 2018.

[22] Andreas Reuter, Randy Bush, Ítalo Cunha, Ethan Katz-Bassett, Thomas C. Schmidt, and Matthias Wählisch. Towards a Rigorous Methodology for Measuring Adoption of RPKI Route Validation and Filtering. *ACM SIGCOMM Computer Communication Review*, 48(1):19–27, January 2018.

[23] RIPE NCC. The RIPE NCC's Case Against the State of the Netherlands Dismissed, 14 February 2013. https://www.ripe.net/publications/news/about-ripe-ncc-and-ripe/ripe-nccs-case-against-the-state-of-the-netherlands-dismissed.

[24] RIPE NCC. RIPE NCC Blocks Registration in RIPE Registry Following Order from Dutch Police, 9 November 2011. https://www.ripe.net/publications/news/about-ripe-ncc-and-ripe/ripe-ncc-blocks-registration-in-ripe-registry-following-order-from-dutch-police.

[25] Muhammad Saad, Afsah Anwar, Ashar Ahmad, Hisham Alasmary, Murat Yuksel, and Aziz Mohaisen. Routechain: Towards blockchain-based secure and efficient BGP routing. In *IEEE ICBC*, pages 210–218. IEEE, 2019.

[26] Mark Tinka. RPKI ROV & Dropping of Invalids – Africa, 09 April 2019. https://www.mail-archive.com/apops@apops.net/msg00796.html.

[27] Christopher S Yoo and David A Wishnick. Lowering legal barriers to rpki adoption. *U of Penn Law School, Public Law Research Paper*, (19-02), 2019.