On the Risk of Misbehaving RPKI Authorities

Full version from November 5, 2013

Danny Cooper* Ethan Heilman* Kyle Brogle† Leonid Reyzin* Sharon Goldberg*
*Boston University, Boston, MA 02215 USA
†Stanford University, Stanford, CA 94305 USA
{dannyc,heilman}@bu.edu, broglek@stanford.edu, {reyzin,goldbe}@cs.bu.edu

ABSTRACT

The RPKI is a new security infrastructure that relies on *trusted authorities* to prevent some of the most devastating attacks on interdomain routing. The threat model for the RPKI supposes that authorities are trusted and routing is under attack. Here we discuss the risks that arise when this threat model is flipped: when RPKI authorities are faulty, misconfigured, compromised, or compelled to misbehave. We show how design decisions that elegantly address the vulnerabilities in the original threat model have unexpected side effects in this flipped threat model. In particular, we show new targeted attacks that allow RPKI authorities, under certain conditions, to limit access to IP prefixes, and discuss the risk that transient RPKI faults can take IP prefixes offline. Our results suggest promising directions for future research, and have implications on the design of security architectures that are appropriate for the untrusted and error-prone Internet.

This is the full version of a short paper that appears at Hotnets '13, November 21–22, 2013, College Park, MD.

1. INTRODUCTION

A number of crucial Internet security infrastructures derive their security from information provided by *authorities* — trusted third parties who attest to information about cryptographic keys, domain names, and/or IP prefixes. Examples include DNS/DNSSEC; the public key infrastructure used for web (SSL/TLS) security; and, most recently, the RPKI [43], a new infrastructure for securing interdomain routing. When authorities behave correctly, each security infrastructure effectively prevents attacks on the system it was designed to protect [15, 20, 35]. However, what happens if an authority malfunctions, is misconfigured, or is compromised by an external attacker? Centralized authorities are also an easy target for lawful (or extralegal) coercion by state-sponsored actors seeking to impose censorship, information control, or surveillance. As state-sponsored interference in Internet systems has become more common in recent years [26, 54, 58], questions of Internet security also begin to have implications on Internet freedom.

We study the RPKI to gain insight on open questions related to the design of network security architectures that are robust to errors, misconfigurations, and abuse by authorities. This analysis is particularly timely given the recent problems with authorities in established systems like DNS and the web PKI; indeed there is ample evidence of authorities in both systems being hacked [16, 28, 47], misconfigured [65], or compelled by government agencies to delete information (*e.g.,* DNS takedowns [54]) or to attest to bogus information [58]. We discuss how the RPKI presents a new point in the design space, show how its design creates unexpected side effects when authorities are compromised, and raise open problems with implications on the design of future Internet security infrastructures.

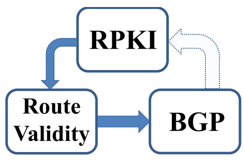


Figure 1: Dependencies.

The RPKI. The RPKI [43] is a security infrastructure built on top of interdomain routing that has recently been standardized by the IETF and adopted by the Regional Internet Registries (RIRs). It is slowly being rolled out by individual network operators. The purpose of the RPKI is to provide a trusted mapping from an IP prefix to a set of autonomous systems (ASes) that are authorized to originate (*i.e.,* claim to be the destination for) this prefix in interdomain routing. This trusted mapping can then be used to protect against the most devastating attacks on interdomain routing with BGP; namely, prefix and subprefix hijacks [20], where an AS originates ("hijacks") routes for IP prefixes that it is not authorized to originate, causing the traffic intended for those prefixes to be intercepted by the hijacker's network. As shown in Figure 1, information in the RPKI determines whether a route is valid, which can, in turn, determine the routes selected in BGP.

The RPKI is the necessary prerequisite for many more advanced proposals for securing BGP (*e.g.,* [38, 41, 64]). Moreover, almost all of the routing attacks seen in the wild (*e.g.,* [24, 48, 57]) could be prevented if Internet routers dropped routes that the RPKI deems invalid;

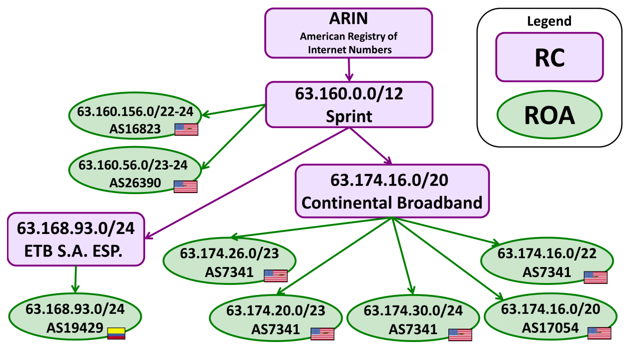


Figure 2: Excerpt of a model RPKI

dropping RPKI-invalid routes is also surprisingly effective against more advanced routing attacks, even those that the RPKI was not designed to prevent [29, 45].

A question. The potential for faulty or compromised RPKI authorities to instantaneously affect BGP routing has lead to some concern among practitioners and policy makers [10, 22, 50, 51, 56, 61]. Does the RPKI create new risks that can take IP prefixes offline?

Our answer. One might expect this question to be completely addressed by the RPKI specifications. However, the RPKI is designed to operate in a threat model where authorities are trusted, but BGP is under attack. We therefore address operational and policy concerns by flipping the threat model: what if RPKI authorities are faulty, misconfigured, compromised, or coerced into behaving incorrectly? In Sections 3-4, we show how design decisions that elegantly address the vulnerabilities in the original threat model have unexpected side effects when analyzed in this flipped threat model.

The scope and variety of these threats is quite different than in a typical PKI. Section 3.1 shows how the hierarchical structure of the RPKI allows abusive authorities to exercise targeted control over their distant descendants (rather than just the objects they issue directly, as in a typical PKI). Section 4 shows how design decisions that are essential to preventing to subprefix hijacks on BGP mean that routing can be harmed if RPKI objects are simply missing (rather than revoked, corrupted, or forged, as in a typical PKI). We also close the loop in Figure 1 by showing how side effects can interact in a circular manner that can turn transient faults into persistent problems (Section 6). Finally, we discuss why (a) robustness to threats to BGP, and (b) robustness to threats to the RPKI, may be at odds (Section 5); the risk that an RPKI problem can take a prefix offline therefore depends on the policies that routers use to balance the two threats against each other.

Organization. Section 2 overviews all components in Figure 1. Sections 3-6 analyze each individual component in the flipped threat model. Our results are based on measurement-driven models and analysis of RPKI software and RFCs (cited where appropriate throughout, along with related work). Our data-driven models and results are given in Appendix B.

To our knowledge, other research on the architecture of the RPKI is sparse, and covers network measurement [53, 62], and policy [22, 50, 51, 61]. Our contributions are summarized in Section 7.

2. OVERVIEW: ROUTING WITH THE RPKI

The RPKI. Most vulnerabilities in the web PKI result from architectural decisions that allow (almost) any authority to issue certificates for any subject [58]. In contrast, the RPKI follows the "principle of least privilege", arranging authorities in a strict hierarchy that mirrors the IP address allocation hierarchy. An authority may issue cryptographic objects for IP addresses that are *covered* by its own IP addresses.[1] Today, IANA sits at the root of this hierarchy, allocating IP addresses to the Regional Internet Registries (RIRs), which allocate subsets of their address space to national/local internet registries (NIRs or LIRs) or ISPs, who further allocate subsets to other ISPs or customers.[2] In RPKI, each authority has a *resource certificate (RC)* that contains its cryptographic public key and its set of allocated IP addresses [46]. An authority may issue signed cryptographic objects for IP addresses covered by its allocation, specifically: (1) an RC that suballocates a subset of its addresses to another authority, or (2) a *route origin authorization (ROA)*[3], that authorizes a specified AS to originate a prefix, and its subprefixes up to a specified length, in BGP [43, Section 2.2].

Figure 2 shows how an RIR (ARIN) uses its RC to suballocate a prefix to another authority (Sprint), which then issues RCs suballocating this prefix to other authorities (ETB S.A. ESP., Continental Broadband). (This is a excerpt from our "Non-stub CA model"; see Appendix B.3.1.) We say Sprint is the *parent* of Continental Broadband, and extend this to child, grandparent, *etc.* in the obvious way. Sprint issues two ROAs that authorize specified prefix and its subprefixes of length up to 24; the remaining ROAs shown authorize only a single prefix.

Route validity (Section 4.) A *relying party* is a party that uses information in the RPKI to make routing decisions in BGP. For our purposes, a BGP *route* is an IP prefix and an origin AS. RPKI objects are stored in publicly-available repositories distributed throughout the Internet. Once a relying party has "access to a local

[1] An IP prefix P *covers* prefix π if π is a subset of the address space in P (*e.g.,* 63.160.0.0/12 covers 63.168.93.0/24) or if $P = \pi$. Also, a prefix 63.160.0.0/12 has *length* 12.
[2] The root(s) of the RPKI hierarchy are not yet specified, but will likely be the five RIRs or IANA [43, Section 2.4].
[3] Strictly speaking, an authority issues a one-time-use end-entity (EE) certificate, which is then used to sign the ROA, but that detail is not important for this paper.

cache of the complete set of valid ROAs" [33, Sec. 2], these valid ROAs are used to classify each route learned in BGP into one of three *route validation states.* Routes with matching valid ROAs are classified as *valid.* Other routes are either *invalid* or *unknown.* The RPKI allows arbitrary prefix lengths, but the smallest IPv4 prefix length which is globally routable in BGP is a /24; so the presence or absence of finer-grained RCs and ROAs has little impact on BGP.

BGP (Section 5.) A relying party uses a route's validation state to decide what routes to *select* in BGP. What impact does an invalid (or unknown) route have on BGP? This depends on "local policies" at each relying party [33] that reflect tradeoffs between robustness to RPKI attacks and robustness to BGP attacks.

3. MANIPULATIONS OF THE RPKI

Recall that a route is authorized by using ROA. Here we show how the architecture of the RPKI's certificate hierarchy enables targeted manipulations that can cause ROAs to become invalid. Sections 4-5 discuss the impact of an invalid ROA on BGP routing.

Design Decision: Revocation. In a traditional PKI, an authority can revoke any *child* certificate it issued, to remedy compromises of its child's cryptographic keys [23]. The RPKI inherits this functionality.

Side Effect 1: Unilateral reclamation of IP address allocations, with little recourse. Revocation of RCs or ROAs in the RPKI creates a new technical mechanism for an authority to *unilaterally* reclaim IP address space. Extending Amante's apt comparison of an RIR to a registry of deeds [10] for real estate, we can think of RPKI as a system of leases and subleases of IP addresses. RPKI design gives a landlord unilateral power to evict a tenant with whom it may have a business dispute or a political disagreement. This creates precisely the imbalance of power that eviction laws try to correct. The RPKI's hierarchical nature also means that the holder of the reclaimed space has little recourse available, since its space may only be reissued by authorities holding supersets of the reclaimed space (similar to DNS but in stark contrast with the web PKI, where any authority may issue any certificate).

Revocation is typically done via a CRL, a publicly-available list of revoked certificates that is signed by the revoking authority [23]. Relying parties could use this list to detect and react to abusive revocations. However, we now show that other design decisions allow RPKI objects to be revoked in a less transparent manner.

Design Decision: Distributed RPKI repositories and out-of-band certificate delivery. In the traditional PKI, the subject of the certificate delivers it to the verifier [40, p. 40]; a website sends its web certificate to a client in an SSL/TLS handshake. In contrast, BGP lacks a handshake phase, and the RPKI was designed to require minimal changes to BGP. In RPKI, relying parties download and verify RPKI objects out of band (rather in real time as part of BGP), and RPKI objects are stored at directories that are *controlled by their issuer* [31] [43, Section 8]. For example, the two RCs and two ROAs issued by Sprint in Figure 2 are held by entities other than Sprint but are published by Sprint at a directory controlled by Sprint. In this sense, the RPKI is more similar to a trusted directory (*e.g.,* DNS) than to a traditional PKI.

Design Decision: Objects can be overwritten. An RPKI authority may overwrite RCs and ROAs that it issued, so that objects can have persistent names (which simplifies operations like key rollover [34]).

Side Effect 2: Stealthy revocation of a child's object. Therefore, an authority can delete any ROA or RC it issued from its repository [36], or even overwrite it with one for a smaller set of IP addresses. This complicates attempts to monitor the RPKI for abusive revocations, especially since distinguishing between abusive behavior and normal RPKI churn could be difficult.

We now present new attacks that can make a ROA invalid in the RPKI. To unify terminology, we say that an *RPKI manipulator whacks* a *target* ROA, regardless whether this is accomplished by a known method above or by a new method below.

3.1 Targeted whacking of distant descendants.

Revocation is a blunt instrument in a hierarchical PKI, as it invalidates an entire subtree of certificates, causing obvious and undesirable damage. For example, if Sprint wanted to target the ROA (63.174.16.0/20, 17054) in Figure 2, it could revoke the RC issued to Continental Broadband, but this would whack four additional ROAs as collateral damage; one might argue that the outcry from this collateral damage could deter deliberate revocations [55]. However, we show that an RPKI manipulator can exercise fine-grained control over ROAs that are its distant descendants without whacking other ROAs as collateral damage.

Design Decision: Fine-grained resource allocation. In a traditional PKI, an authority binds a *single* name to a cryptographic key. By contrast, RPKI authorities bind *arbitrary sets* of IP addresses to a key.

Side Effect 3: Targeted whacking of a grandchild. Because an authority may issue RCs for arbitrary subsets of its IP addresses, a manipulator can whack any grandchild ROAs by removing, from the target's parent RC, a portion of the address space contained in the target ROA. If the removed portion of the space overlaps no other RCs or ROAs issued by the target's parent, this action will cause no collateral damage. For example, Sprint can surreptitiously the target ROA (63.174.16.0/20, 17054) in Figure 2 by overwriting

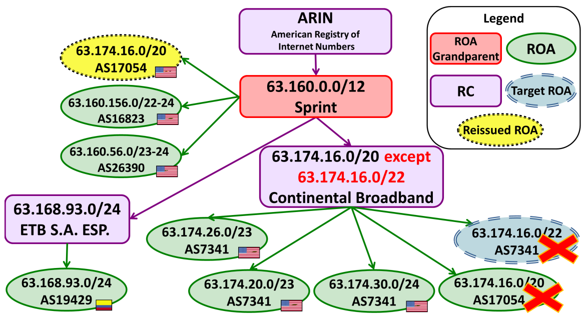


Figure 3: A ROA whacked by its grandparent.

the RC it issued for Continental Broadband with the one for the two IP ranges [63.174.16.0–63.174.23.255] and [63.174.25.0–63.174.31.255]. Because this new RC covers all the ROAs issued by Continental Broadband (except the target ROA), all other ROAs remain valid.

When non-overlapping space cannot be found (*e.g.,* if Sprint wants to target the ROA (63.17.16.0/22, AS 7341) in Figure 2), the manipulator can first (1) reissue the damaged descendant objects as its own ("make-before-break"), and then (2) overwrite the appropriate child RC (as shown in Figure 3). This situation is easier to detect, due to the suspiciously-reissued ROA. One of the open problems we are working is the design of monitoring schemes that deter RPKI manipulations by detecting suspiciously reissued objects.

To understand how frequently whacking results in suspiciously-reissued objects, we modeled the eventual deployment of the RPKI using BGP data and information about IP address allocations provided by the RIRs (today's production deployment is too small—about 1200-1400 ROAs, which is less than 1% of projected deployment, according to our models and [53]). See Appendix B.3 for details.

Side Effect 4: Whacking of great-grandchildren and beyond. ROAs below grandchild level can also be whacked without collateral damage. However, details of how RPKI objects point to RPKI repositories mean that this whacking requires more suspiciously-reissued objects, and could be easier to detect. Thus, the manipulator must changes the set of IP prefixes in the RC that is the child of the manipulator and the ancestor of the target ROA. This action will damage the subtree rooted at the RC that is the manipulator's *grandchild* and the ancestor of the target ROA, so the manipulator also needs to reissue the damaged objects as well. For example, if ARIN wanted to whack its greatgrandchild's ROA (63.174.16.0/20, AS 17054), it could (a) overwrite Sprint's RC with an RC for IP ranges [63.160.0.0–63.174.31.0] and [63.174.32.0–63.176.0.0], (b) reissue Continental Broadband's RC (now for [63.174.16.0–63.174.31.0]) and (c) reissue all ROAs that were issued by Continental Broadband, except the target ROA.

Holder	RC	Countries
Level3	8.0.0.0/8	RU,FR,NL,CN,TW,JP,GU,AU,GB,MX
Cogent	38.0.0.0/8	GU,GT,HK,GB,IN,PH,MX
Verizon	65.192.0.0/11	CO,IT,AN,AS,GB,EU,SG
Sprint	208.0.0.0/11	AS,BO,CO,ES,EC
Sprint	63.160.0.0/12	FR,CO,YE,AN,HN
Tata Comm.	64.86.0.0/16	GU,CO,MH,HN,PH,ZW
Columbus	63.245.0.0/17	NI,GT,CO,AN,HN,MX
Servcorp	61.28.192.0/19	FR,AE,CA,US,GB
Resilans	192.71.0.0/16	US,IN

Table 4: A few RCs & the countries they cover that are outside jurisdiction of their parent RIR.

Details on why this works, as well a generic procedure for whacking ROAs, are provided in Appendix A.

3.2 International borders.

When a manipulator whacks a ROA in the same legal jurisdiction, the holder of the target ROAs may have some legal recourse against the manipulator's action. But what if the manipulator and target are in different jurisdictions?[4] Indeed, many IPv4 addresses were historically suballocated with little regard for questions of international jurisdiction. Using BGP data, information about IP address allocations, and AS-to-country mappings provided by the RIRs (Appendix B.1 has details) we found that cross-country certification is not uncommon. RIRs can whack ROAs for ASes in *non-member* countries, even though they are are accountable only to their member countries. For example, through its certification of Sprint, North America's ARIN can whack ROAs for Europe and the Middle East. Europe-based RIPE can whack ROAs in Asia and the Americas. A few RCs held by *private* entities also cover ROAs in multiple countries. Table 4 has a few salient examples.

4. RPKI ⇒ ROUTE VALIDITY

Route validity decisions are made by *relying parties* once they have determined a *complete set* of all valid ROAs and stored them in a *local cache* [33, Sec. 2]. If a ROA is whacked, expires, or is missing due to a fault or misconfiguration, it will not be in this local cache. What impact does its absence have on route validity? We show how the semantics of determining route validity, which were designed to limit the risk of subprefix hijacks on BGP, can lead to unintuitive consequences.

Design Decision: Retaining BGP's subprefix semantics. BGP is vulnerable to subprefix hijacks because of longest-prefix-match routing: when a router is offered BGP routes for a prefix and its subprefix, it always chooses the subprefix route. Subprefix hijackers exploit this by originating routes for subprefixes of a victim prefix [57]. This leads to a natural design goal: a subprefix hijacker's route should be invalid when victim's route has a matching valid ROA. To achieve this

[4]For example, RIPE is under Dutch jurisdiction and subject to Dutch laws [59]. However, RIPE allocates prefixes to all of the European nations.

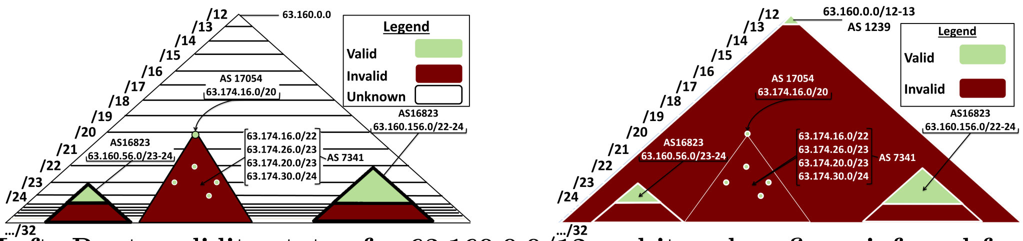


Figure 5: Left: Route validity status for 63.160.0.0/12 and its subprefixes, inferred from the RPKI of Figure 2. Right: The change in route validity if a ROA (63.160.0.0/12-13, AS1239) is added.

goal, a relying party performs *origin authentication* as follows. Each BGP route for prefix π and origin AS a is classified with one of three *validation states*, based on all the valid ROAs in the relying party's local cache [33,49]:

- *Valid:* There is a valid *matching ROA*. A matching ROA has (1) a matching origin AS a, and (2) a prefix P that covers prefix π, and (3) the specified maximum length no shorter than the length of π.
- *Unknown:* There is no valid *covering ROA*. A covering ROA is any ROA for a prefix that covers π.
- *Invalid:* The route is neither unknown or valid.

Figure 5 (left) shows how the ROAs in Figure 2 determine the validity of routes for 63.160.0.0/12 and all its subprefixes. The rules above elegantly achieve the design goal; the ROA for (63.174.16.0/20, AS 17054) protects the corresponding route from subprefixes hijacks, because all routes for its subprefixes are "invalid" (except routes with matching ROAs of their own).

These rules also imply that route is "unknown" only if there is *no covering ROA*.[5] The RPKI in Figure 2 contain ROAs for neither 63.160.0.0/12 nor 63.174.17.0/24. However, Figure 5(left) shows that routes for 63.160.0.0/12 are "unknown" (there is no covering ROA) but routes for 63.174.17.0/24 are "invalid" (because of the ROA for 63.174.16.0/20). This has two side effects.

Side Effect 5: A new ROA can cause many routes to become invalid. Figure 5 (right) shows that if Sprint issues a new ROA (63.160.0.0/12-13, AS 1239) that covers previously unknown routes, those routes become "invalid". This creates a deployment challenge, since (a) the earliest adopters of the RPKI are likely to be large networks (like Sprint) that hold large prefixes, but (b) a new ROA for a large prefix should be issued only after *all* ROAs for its subprefixes, to prevent routes from mistakenly becoming invalid. Indeed, [62] found that the production RPKI classified many production BGP routes as invalid, likely for this very reason.

Side Effect 6: A missing ROA can cause a route to become invalid. Missing information is problematic in any secure system, especially so in the RPKI, because the absence of a ROA in a relying party's local cache does *not* mean that the corresponding route is merely "unknown" (as in *e.g.,* DNSSEC or the web PKI). The requirement that relying parties have access to a *complete* set of valid ROAs [33, Sec. 2] is therefore crucial; for example, if the ROA (63.174.16.0/22, AS 7341) is missing from the RPKI of Figure 2, the corresponding route will be classified as "invalid" (instead of "unknown") because of the covering ROA for prefix 63.174.16.0/20 (see Figure 5 (left)). This makes the RPKI vulnerable to faults that disrupt the delivery of valid ROAs, a side effect that is easily misunderstood [13,25]. Information can be missing for a variety of reasons: the renewal of an expiring ROA could be delayed (accidentally or maliciously); the filesystem or server storing the ROA could become corrupted; *etc.*. While this may cause only temporary disruptions, Section 6 discusses how this can create persistent failures.

A difficult tradeoff: What to do about incomplete information? The RFCs do not specify what action should be taken when a relying party suspects a valid ROA might be missing from a repository (*e.g.,* see [12, Sect 6.5]). Should a party stop relying on the RPKI if it thinks ROAs could be missing? On one hand, this avoids the problems discussed in Side Effect 6. On the other, it opens up the relying party to BGP attacks.

It is an open problem to design architectures for route validity that prevent subprefix hijacks but are not brittle in case of missing information or misconfiguration. Alternatively, monitoring and configuration tools could be used to mitigate these risks.

Summary: RPKI problems ⇒ invalid routes? We have seen that a route can become "invalid" due to: (1) a misconfiguration by an RPKI authority (Side Effect 5), (2) missing information in a relying party's cache (Side Effect 6), (3) a ROA that is whacked AND is also covered by a valid ROA (Section 3).

5. ROUTE VALIDITY ⇒ BGP

What impact does an invalid (or unknown) route have on BGP routing? That depends on to the "local policies" at each relying party [33]. We now consider the two most plausible policies, as suggested by [33]:

Drop Invalid: This policy requires that a relying party never selects an invalid route. It fully realizes RPKI's

[5]Note that, in principle, other designs choices are possible, *e.g.,* requiring each ROA to explicitly indicate which routes for its subprefixes should remain valid or unknown.

relying-party policy	Prefix reachable during...	
	routing attack	RPKI manipulat'n
drop invalid	✓	X
depref invalid	subprefix hijacks possible	✓

Table 6: Impact of different local policies.

potential to protect routing, stopping prefix and subprefix hijacks (Section 1). However, if RPKI problems causes a route to become invalid, the relying party will lose connectivity to the corresponding IP prefix.

Depref invalid: This (more lenient) policy requires that, for a given prefix, a router prefers valid routes over invalid routes. This means that a router still selects an invalid route when there is no valid route for the *exact same* IP prefix. Thus, the router may still be able to reach prefixes whose routes have become invalid due to problems with the RPKI.[6] However, this policy does *not* prevent subprefix hijacks; see [18, Section 5].

A difficult tradeoff: RPKI attacks vs. BGP attacks? Table 6 highlights a tradeoff that is implicit in the RPKI RFCs; namely, that the local policy that is best at protecting against problems with BGP is worst at protecting against problems with RPKI. Balancing these considerations is a challenging open problem.

6. CLOSING THE LOOP: BGP ⇒ THE RPKI

Finally, we highlight the complexities involved in architecting a system like the RPKI, by closing the loop in Figure 1. To do this, we show how a chain of (unlikely, but plausible) events can lead to persistent failures.

Design Decision: Delivery of RPKI information over TCP/IP. A PKI is typically deployed as a layer on top of a (possibly unauthenticated) communication infrastructure; web (HTTPS) certificates, for example, are delivered over TCP/IP. Similarly, the only delivery method mandated by the RPKI is the rsync protocol [31, Section 2.2], which runs on top of TCP/IP. (Other delivery methods are allowed at operator discretion.) However, unlike web certificates, RPKI objects can affect the availability of BGP routes, and therefore also of TCP/IP, the very infrastructure over which they are delivered. This can create a circular dependency.

Side Effect 7: Transient faults cause long-term failures. We now show how this design decision, the decision to allow distributed RPKI repositories located anywhere in the Internet (Section 3), and the issues in Sections 4-5, can cause a *transient* error to become a persistent failure. Suppose that (1) route validity is as shown in Figure 5 (right), (2) Continental Broadband (AS 17054) hosts its own repository at 63.174.23.0, and (3) a relying party drops invalid routes in BGP.

[6]However, availability of a route at one router can depend strongly on local policy used at other routers. For example, a router that uses the lenient 'depref invalid' policy can lose reachability to a prefix due to a problem with the RPKI if all its neighboring routers use the strict 'drop invalid' policy.

This example contains a circularity: for the relying party to retrieve ROAs issued by Continental Broadband, it must have a valid or unknown route to Continental Broadband's repository at 63.174.23.0 and AS 17054. Because route validity is as in Figure 5 (right), the route to the repository will be invalid unless the relying party can retrieve the ROA binding 63.174.16.0/20 to AS 17054. However, this ROA is issued by Continental Broadband, and is therefore hosted at Continental Broadband's repository. Thus, for the relying party to access Continental Broadband's repository, it must first access a ROA that is stored at that repository.

Now suppose a transient error causes the relying party to receive a corrupted ROA for (63.174.16.0/20, AS 17054) (see Side Effect 6). As explained above, the relying party will lose access to Continental Broadband's repository. Even if the fault is remedied and the repository is ready to serve the missing ROA, the relying party cannot obtain the missing ROA, because it cannot reach the repository. This can be fixed (manually), but there no are recommended procedures for recovery.

The example arises because (a) the ROA for a route to an RPKI repository is stored at that same repository, and (b) another ROA covers but does not match the route to the repository, and (c) the relying party drops invalid routes. (Condition (a), but not its implications, was also pointed out by [33].) More complex circular dependencies can exist, involving multiple ROAs and repositories, and it is an open question to develop operational guidelines that eliminates these dependencies.

7. CONCLUSION & OPEN PROBLEMS

The RPKI has the potential to eliminate most of the routing attacks seen in the wild (*e.g.,* [24,48,57]), and is a prerequisite for more advanced proposals for securing BGP [20]. However, we showed that its architecture creates new technical means for authorities to *unilaterally* reclaim IP address allocations (Side Effects 1–2), in a targeted manner, even to distant descendants (Side Effects 3–4). This leaves the target with little recourse, especially when the relationship between the target and authority crosses international borders (Section 3.2). We note that these manipulations are more coarse-grained than domain name seizures [55], because current BGP practices limit their granularity to a /24 IPv4 prefix, *i.e.,* 256 IPv4 addresses.

We also showed how confusion (Side Effect 5) and sensitivity to missing information (Side Effect 6) can lead to RPKI misconfigurations that cause routes to become invalid. Finally, we showed how circular dependencies between the RPKI and BGP can lead to persistent failures (Side Effect 7). Our results leave RPKI relying parties with a seemingly difficult tradeoff (Section 5): They can use local policies that (a) send traffic on invalid routes, thus reducing their vulnerability to problems with the RPKI while increasing vulnerability to problems with BGP, or (b) stop sending traffic on invalid routes, which has the opposite effect.

Open Problems. The routing security improvements promised by the RPKI [20, 29, 45] motivate efforts to harden the RPKI against errors, misconfigurations, and abuse; indeed, concurrently to our work there have been new steps in this direction in the IETF [19,27,39]. There are a number of issues to address. Can abuse by RPKI authorities be made more difficult to execute, more limited in scope, or easier to detect? Is the RPKI's sensitivity to missing objects caused by fundamental design requirements, or are there alternate architectures that are more robust? Can we develop better local policies for relying parties that overcome the difficult tradeoff we mentioned above? How should Internet routing be secured if the only means of communication is the Internet itself? Addressing these issues in the context of the RPKI will also inform the design of future security architectures that are appropriate for the inherently untrusted and error-prone Internet.

Acknowledgements. We thank Steve Kent for suggesting the term "ROA whacking" to us, Tony Tauber for useful discussions, FCC CSRIC Working Group 6 - Secure BGP Deployment for inspiring this study, the Measurement Factory for IPv4 Hilbert curve software [60], CAIDA for early access to their AS to organization mapping [21], and Dimitris Papadopoulos, Davide Proserpio, and Jennifer Rexford for comments on this draft. This work was supported by NSF awards 1017907, 1012798, and 1012910.

8. REFERENCES

[1] ftp://ftp.afrinic.net/pub/stats/afrinic/2012/.
[2] ftp://ftp.apnic.net/pub/stats/apnic/2012/.
[3] ftp://ftp.arin.net/pub/stats/arin/archive/2012/.
[4] ftp://ftp.lacnic.net/pub/stats/lacnic/.
[5] ftp://ftp.ripe.net/pub/stats/ripencc/2012/.
[6] RIPE RIS raw data. http://www.ripe.net/data-tools/stats/ris/ris-raw-data.
[7] Rpki spider. http://rpkispider.verisignlabs.com/.
[8] University of oregon route views project. http://www.routeviews.org/.
[9] Working group 6, secure bgp deployment, final report. Technical report, FCC CSRIC Working Group 6, March 2013.
[10] S. Amante. Risks associated with resource certification systems for internet numbers, 2012.
[11] ARIN. Resource public key infrastructure (rpki). https://www.arin.net/resources/rpki/index.html, Retrieved April 2013.
[12] R. Austein, G. Huston, S. Kent, and M. Lepinski. *RFC 6486: Manifests for the Resource Public Key Infrastructure (RPKI)*. Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6486.
[13] A. Band. "Re: rpki vs. secure dns?", msg566. seclists NANOG Archive, apr 2012. http://seclists.org/nanog/2012/Apr/566.
[14] A. Band. Resource certification (RPKI). In *RIPE'64*, 2012. https://ripe64.ripe.net/presentations/77-RIPE64-Plenery-RPKI.pdf.
[15] M. Benantar. The internet public key infrastructure. *IBM Systems Journal*, 40(3):648–665, 2001.
[16] P. Bright. arstechnica: How the Comodo certificate fraud calls CA trust into question, March 2011. http://arstechnica.com/security/2011/03/how-the-comodo-certificate-fraud-calls-ca-trust-into-question/.
[17] R. Bush. *Responsible Grandparenting in the RPKI*. Internet Engineering Task Force Network Working Group, 2012. http://tools.ietf.org/html/draft-ymbk-rpki-grandparenting-02.
[18] R. Bush. *RPKI-Based Origin Validation Operation*. Internet Engineering Task Force Network Working Group, 2012. http://tools.ietf.org/html/draft-ietf-sidr-origin-ops-19.
[19] R. Bush. *RPKI Local Trust Anchor Use Cases*. Internet Engineering Task Force (IETF), 2013. http://www.ietf.org/id/draft-ymbk-lta-use-cases-00.txt.
[20] K. Butler, T. Farley, P. McDaniel, and J. Rexford. A survey of BGP security issues and solutions. *Proceedings of the IEEE*, 2010.
[21] CAIDA. AS to organization mapping. http://as-rank.caida.org/?mode0=as-intro#as-org.
[22] Communications Security, Reliability and Interoperability Council III (CSRIC). Secure bgp deployment. *Communications and Strategies*.
[23] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. *RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. Internet Engineering Task Force (IETF), 2008. http://tools.ietf.org/html/rfc5280.
[24] J. Cowie. Rensys blog: China's 18-minute mystery. http://www.renesys.com/blog/2010/11/chinas-18-minute-mystery.shtml.
[25] J. Curran. "Re: [sidr] Princeton University:: Impacting IP Address Reachability via RPKI Manipulations", msg05906. IETF, sidr archive, apr 2013. http://www.ietf.org/mail-archive/web/sidr/current/msg05906.html.
[26] R. Deibert, J. Palfrey, R. Rohozinski, and J. Zittrain. *Access controlled: The shaping of power, rights, and rule in cyberspace*. MIT Press, 2010.
[27] R. Gagliano, T. Manderson, and C. M. Cagnazzo. *Multiple Repository Publication Points support in the Resource Public Key Infrastructure (RPKI)*. Internet Engineering Task Force (IETF), 2013. http://tools.ietf.org/html/draft-ietf-sidr-multiple-publication-points-00.
[28] E. Galperin, S. Schoen, and P. Eckersley. A post mortem on the iranian diginotar attack. *EFF Blog*, September 2011.
[29] S. Goldberg, M. Schapira, P. Hummon, and J. Rexford. How secure are secure interdomain routing protocols? In *SIGCOMM'10*, 2010.
[30] G. Huston. *RFC 6485: The Profile for Algorithms and Key Sizes for Use in the Resource Public Key Infrastructure (RPKI)*. Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6485.
[31] G. Huston, R. Loomans, and G. Michaelson. *RFC 6481: A Profile for Resource Certificate Repository Structure*. Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6481.
[32] G. Huston, R. Loomans, and G. Michaelson. *RFC 6487: A Profile for X.509 PKIX Resource Certificates*. Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6487.
[33] G. Huston and G. Michaelson. *RFC 6483: Validation of Route Origination Using the Resource Certificate Public Key Infrastructure (PKI) and Route Origin Authorizations (ROAs)*. Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6483.
[34] G. Huston, G. Michaelson, and S. Kent. *RFC 6489: Certification Authority (CA) Key Rollover in the Resource Public Key Infrastructure (RPKI)*. Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6489.
[35] D. Kaminsky. Black ops 2008: ItŠs the end of the cache as we know it. *Black Hat USA*, 2008.
[36] S. Kent and A. Chi. Rfc draft: Threat model for bgp path security. 2013. http://tools.ietf.org/html/draft-kent-bgpsec-threats-01.
[37] S. Kent, D. Kong, K. Seo, and R. Watro. *RFC 6484: Certificate Policy (CP) for the Resource Public Key Infrastructure (RPKI)*. Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6484.
[38] S. Kent, C. Lynn, and K. Seo. Secure border gateway protocol (S-BGP). *J. Selected Areas in Communications*, 18(4):582–592, April 2000.
[39] S. Kent and D. Mandelberg. *Suspenders: A Fail-safe Mechanism for the RPKI*. Internet Engineering Task Force (IETF), 2013. http://tools.ietf.org/html/draft-kent-sidr-suspenders-00.
[40] L. M. Kohnfelder. *Towards a Practical Public-key Cryposystem*. Massachusetts Institute of Technology, 1978. Bachelor's Thesis. http://groups.csail.mit.edu/cis/theses/kohnfelder-bs.pdf.
[41] M. Lepinski, editor. *BGPSEC Protocol Specification*. IETF Network Working Group, Internet-Draft, July 2012. Available from http://tools.ietf.org/html/draft-ietf-sidr-bgpsec-protocol-04.
[42] M. Lepinski, A. Chi, and S. Kent. *RFC 6488: Signed Object Template for the Resource Public Key Infrastructure (RPKI)*.

Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6488.
[43] M. Lepinski and S. Kent. *RFC 6480: An Infrastructure to Support Secure Internet Routing*. Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6480.
[44] M. Lepinski, S. Kent, and D. Kong. *RFC 6482: A Profile for Route Origin Authorizations (ROAs)*. Internet Engineering Task Force (IETF), 2012. http://tools.ietf.org/html/rfc6482.
[45] R. Lychev, S. Goldberg, and M. Schapira. Is the juice worth the squeeze? BGP security in partial deployment. In *SIGCOMM'13*, 2013.
[46] T. Manderson, L. Vegoda, and S. Kent. *RFC 6491: Resource Public Key Infrastructure (RPKI) Objects Issued by IANA"*. Internet Engineering Task Force (IETF), 1973. http://tools.ietf.org/html/rfc6491.
[47] M. Marquis-Boire. A brief history of dns hijackings (at google). ICANN'43, March 2012.
[48] S. Misel. "Wow, AS7007!". Merit NANOG Archive, apr 1997. www.merit.edu/mail.archives/nanog/1997-04/msg00340.html.
[49] P. Mohapatra, J. Scudder, D. Ward, R. Bush, and R. Austein. *RFC 6811: BGP prefix origin validation*. Internet Engineering Task Force (IETF), 2013. http://tools.ietf.org/html/rfc6811.
[50] M. Mueller and B. Kuerbis. Negotiating a new governance hierarchy: An analysis of the conflicting incentives to secure internet routing. *Communications and Strategies*, (81):125–142, 2011.
[51] M. Mueller, A. Schmidt, and B. Kuerbis. Internet security and networked governance in international relations. *International Studies Review*, 15(1):86–104, 2013.
[52] NIST. Bgp secure routing extension (BGP-SRx). http://www-x.antd.nist.gov/bgpsrx/.
[53] E. Osterweil, T. Manderson, R. White, and D. McPherson. Sizing estimates for a fully deployed rpki. Technical report, Verisign Labs Technical Report, 2012.
[54] D. Piscitello. Guidance for preparing domain name orders, seizures & takedowns. Technical report, ICANN, March 2012.
[55] D. Piscitello. The value of assessing collateral damage before requesting a domain seizure. Technical report, ICANN, January 2013.
[56] I. G. Project. In important case, RIPE-NCC seeks legal clarity on how it responds to foreign court orders, 2011. http://www.internetgovernance.org/2011/11/23/in-important-case-ripe-ncc-seeks-legal-clarity-on-how-it-responds-to-foreign-court-orders/.
[57] Rensys Blog. Pakistan hijacks YouTube. http://www.renesys.com/blog/2008/02/pakistan_hijacks_youtube_1.shtml.
[58] C. Soghoian and S. Stamm. Certified lies: Detecting and defeating government interception attacks against ssl (short paper). In *Financial Cryptography and Data Security*, pages 250–259. Springer, 2012.
[59] S. Steffann. "[address-policy-wg] Legal counsel on 2008-08 (Initial Certification Policy in the RIPE NCC Service Region)". RIPE Archive, may 2011. http://www.ripe.net/ripe/mail/archives/address-policy-wg/2011-May/005858.html.
[60] I. The Measurement Factory. Ipv4 heatmap software, 2007. http://maps.measurement-factory.com/.
[61] The President's National Security Telecommunications Advisory Committee. Nstac report to the president on communications resiliency, 2011.
[62] M. Wählisch, O. Maennel, and T. Schmidt. Towards detecting BGP route hijacking using the RPKI. In *Poster: SIGCOMM'12*, pages 103–104. ACM, 2012.
[63] L. Wang, J. Park, R. Oliveira, and B. Zhang. Internet topology collection. http://irl.cs.ucla.edu/topology/.
[64] R. White. Deployment considerations for secure origin BGP (soBGP). draft-white-sobgp-bgp-deployment-01.txt, June 2003, expired.
[65] C. Wisniewski. Turkish certificate authority screwup leads to attempted google impersonation. Naked Security Blog, January 4 2013.

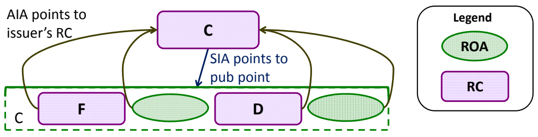


Figure 7: RPKI repository structure.

APPENDIX

A. HOW TO WHACK A ROA

This appendix gives a detailed description of how a manipulator can whack a target ROA below it in the RPKI hierarchy, without causing collateral damage to any other route. We start by providing the necessary background on the RPKI repository structure.

A.1 RPKI Repositories.

The following details will be important for understanding our procedure for whacking a ROA.

RCs and ROAs are placed in a repository [31, Section 2], which is a structure of directories and files, each of which can be identified by a uniform resource identifier (URI). Each RC and ROA is in its own file with its own filename. An updated RC or ROA may have the same filename as the previous one—the old version in the repository simply gets overwritten [31, Section 2.2]. Each RC and ROA (except the root RC, of course) contains the URI for filename of the RC for the authority who issued it, so that certification chains can be traced up the hierarchy (this URI is stored in the so-called "Authority Information Access (AIA)" field; we prefer to call it "parent pointer").

Everything signed by an authority C—that is, RCs, ROAs, and certificate revocation lists (CRLs) issued by C—are placed in a single directory (or as the RFCs call it, *publication point*) for C. The RC of C, which is stored in the directory of his parent, contains the URI of the directory where C will place everything it signs (it is stored in the so-called "Subject Information Access (SIA)" field; we prefer to call it "child pointer"). Thus, all the pointers reflect the tree structure of the certificate hierarchy, with every node pointing to the file of its parent and to the directory containing its children.

These details are illustrated in Figure 7, which shows how RC C's child pointer (SIA) points to C's directory (denoted by the rectangle drawn with dotted lines). All objects issued by C (in this case, RC D and F, as well as two (unlabeled) ROAs) are stored in C's directory, and have parent pointers (AIA) that point to C's RC. To simplify the presentation, our subsequent figures will not depict the parent pointer (AIA); Figure 8 depicts C's RC and repository with AIA pointers removed.

A.2 Generic procedure to whack a target ROA without creating collateral damage.

We now present the generic procedure that a manipulator can use to whack *any* target ROA below it in the RPKI hierarchy (*i.e.,* a grandchild ROA, a great-grandchild ROA, *etc.*). The following procedure does not create collateral damage, in that all prefix-to-AS mappings provided by the RPKI remain valid, except for the one in the target ROA. We used this procedure to obtain the examples discussed in Section 3.1.

We explain the procedure using Figure 8. The manipulator in this example holds RC A in Figure 8, and wants to whack target ROA t that is below RC A in the RPKI hierarchy (bottom right corner). Note that A can only revoke, delete or overwrite objects in its own repository (*i.e.,* its child RCs B and G and three other unlabeled ROAs). Thus, to make t invalid, A will modify B by removing some portion p of the addresses in t from B. This modification, however, will make every RC on the path between B and t invalid (in our example, those RCs are C, D, and E), and therefore will also make subtrees rooted at those RCs invalid. Moreover, it may make other descendants of B invalid, if their address space happens to overlap p. To fix this problem and prevent collateral damage, A will re-issue all the objects that are made invalid by the change (except t, of course) under its own signature in its own directory. In fact, in will issue all those objects before modifying B, in a make-before-break manner.

To make the manipulation as easy to execute and as hard to detect as possible, A may judiciously choose p (which can be as small as a single address) so as to invalidate as few objects as possible. In particular, if t is the grandchild of A, it is likely that no objects will need to be reissued by A, because t will likely have address space that overlaps no other RC or ROA, except t's parent B.

More precisely, the manipulator A proceeds as follows:

1. Prepare a new child RC. Pick some subset p of the address space covered by the target ROA t. This subset should, in particular, minimize the number of objects that need to be issued in Step 2. Prepare, but do not yet publish, a new RC for B that excludes this portion p of the address space, but is otherwise identical to the current RC for B.

2. Re-issue the impacted descendant objects. Find all the objects (except t and B) in the subtree rooted at B that overlap with p. (Note that such objects are limited to the path between B and t, except in the unusual case of overlapping coverage by sibling RCs or ROAs. In particular, if t is the grandchild of A, then there may be no such objects at all, in which case there is nothing to do in this step—see, for example, Side Effect 3 in Section 3.1.) For each such object, reissue it

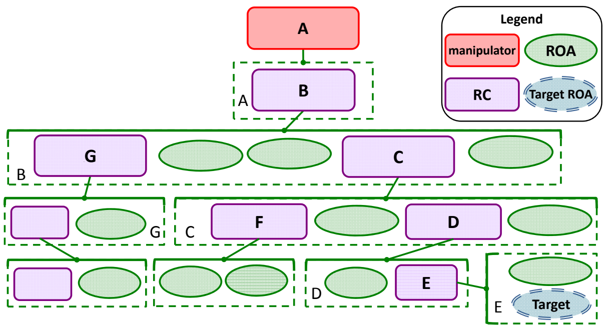


Figure 8: Sample RPKI for Appendix A.2.

and all its descendants as children of A. The newly issued RCs and ROAs will have the same attributes as the original ones, including the same public keys, address space, and child pointers to their subjects' directories (in case of RCs); however, because these RCs and ROAs are issued by A, they will have parent pointers to A's RC, and will be placed in A's directory.

3. Overwrite the child RC. Replace B's original RC with the one created in step 1.

Analysis. Note that in the procedure above, the manipulator never actually revokes any certificates; instead, it *overwrites* an old RC and (sometimes) issues a few new ROAs and RCs. We'd naturally expect to always be able to detect revocations by referring to the certificate revocation list (CRL); the procedure above suggests that this is not always possible. Moreover, all (IP prefix, origin AS) mappings, apart from the those of the target ROA, remain valid according to the RPKI. All parties retain the ability to (autonomously) issue RCs and ROAs for the same prefixes as before, except for prefixes containing the the portion of address space p found in step 1.

This procedure may leave a trace, however, because all the reissued RCs and ROAs, except the new RC of the child B, are now signed by the a different issuer and reside in a new directory. Moreover, the RPKI repository now contains pairs of RCs for a single public key published at different directories (the old one and the new one). Both these issues create an unusual and suspicious repository state that may be detectable. To reduce the number of suspiciously-reissued objects, the manipulator might choose not to re-issue any RCs (only ROAs); while this prevents holders of those RCs from issuing future ROAs, it does not impact the validity of any routes.

Notice, however, that these suspicious new objects are not always needed—as mentioned in Step 2, sometimes there is nothing new to issue. Thus, in Section B.3 we use measurement-driven models to determine how often this procedure can be done "cleanly", *i.e.,* without requiring the manipulator to do anything in Step 2.

B. DATA-DRIVEN MODELS & RESULTS

We use data-driven models to determine when cross-country certification occurs in the RPKI (Section 3.2), where power concentrates in the RPKI (Appendix B.2), and the feasibility of detecting and reacting to RPKI manipulations that whack ROAs (Appendix B.3-B.4).

B.1 Data sources and methodology.

We created a model for a future (full) deployment of the RPKI by using routing data for the week starting 2012–05–06. The RIRs sit at the highest layer of the hierarchy, and thus have the most power, but they do not issue ROAs directly. One level deeper are "direct allocations," *i.e.,* IP prefixes directly allocated by the RIRs. In the RPKI, RIRs will issue RCs to recipients of direct allocations, who will then issue ROAs or further suballocate their address space by issuing deeper RCs [11, 14]. We obtained these top two layers of the hierarchy by using files retrieved from the FTP site of each RIR [1–5]. To model the ROAs covered by each direct allocation, we extracted (prefix, origin AS)-tuples from the BGP update feeds provided by Routeviews [8] and routing table dumps from the RIPE RRCs [6] (any tuple not covered by a directly-allocated prefix was discarded as a bogon). We assumed that such tuples would correspond to ROAs in the RPKI. Whenever possible, we combined multiple tuples into a single ROA by using the appropriate maxLength attribute (*e.g.,* (4.4.1.0/23, 42), (4.4.1.0/24, 42), and (4.4.2.0/24, 42) became a single ROA for (4.4.1.0/23, 42) with maxLength 24). Finally, we use an AS-to-country mapping, provided by each RIR, to map each ROA to a country. The resulting dataset was used to obtain the results in Section 3.2.

B.2 Concentration of power.

A fully-deployed RPKI would concentrate power at authorities at the top of the RPKI hierarchy (because they can whack the most ROAs). We consider who those parties might be, and the extent of their power, by using the data described in Section B.1 to quantify the extent of entity's power as: the number of ROAs it can whack, the number of ASes in those ROAs, and the number of BGP-announced (prefix, origin AS) tuples that were combined (with the help of the maxLength attribute) to make those ROAs.

The results for the first layer of the RPKI hierarchy—the RIRs—is summarized in Table 10 (which also includes the number of direct allocations issued per RIR).

For the second layer of the RPKI hierarchy (*i.e.,* recipients of direct allocations), our data shows that the distribution of power is highly skewed. On average, each recipient sits above only 1.5 ASes and can whack 2.0 ROAs corresponding to 4.4 (prefix, origin AS) tuples. But some recipients cover several hundred ROAs and have power comparable to the entire AfriNIC RIR. The

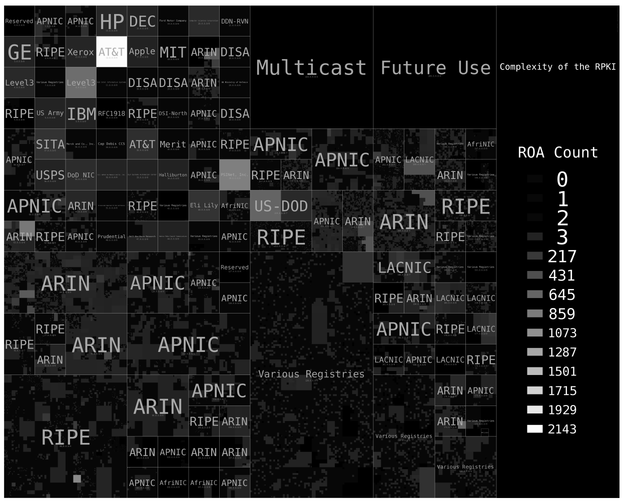


Figure 9: Number of ROAs that can be whacked by each direct allocation.

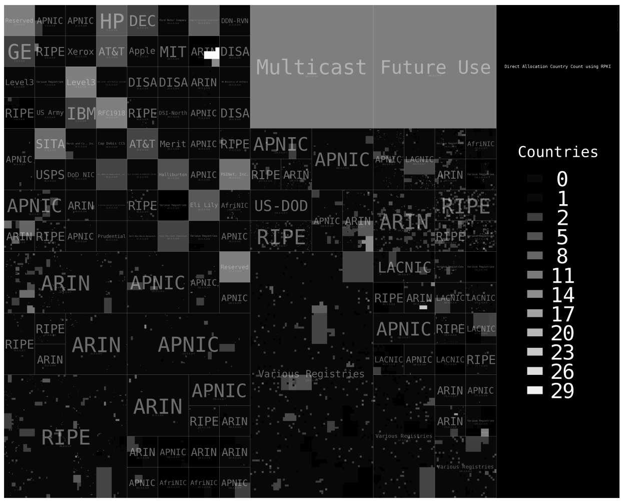


Figure 11: The number of countries that hold ROAs that descend from each direct allocation.

data for the top four most powerful entities in the second layer of the RPKI is shown in Table 12. The overall picture is presented Figures 9 and 13, which show the number of ROAs and ASes, respectively, that descend from each direct allocation using a Hilbert curve of the IPv4 address space. (The Hilbert curve maps 1-dimensional IPv4 address space into a 2-dimensional space; every consecutive group of IP addresses will map onto a single contiguous space on the map. For example, the square on the top left labeled 'AT&T' represents the prefix 12.0.0.0/8, and every pixel on the original 4096 × 4096 image represents a single /24 prefix. See [60] for more details.)

RIR	#Dir. Alloc.	#ASes	#ROAs	#(pfx, AS) tuples
ARIN	44,431	16,063	107,446	198,058
RIPE	45,834	18,509	71,563	136,420
APNIC	20,266	4,814	43,815	117,768
LACNIC	3,825	2,205	10,876	48,623
AfriNIC	2,001	625	3,537	11,755

Table 10: The power of RIRs in the RPKI

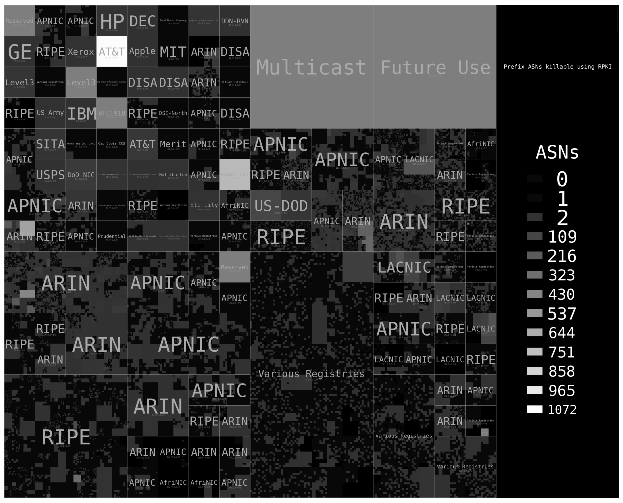


Figure 13: The number of ASes that hold ROAs that descend from each direct allocation.

B.3 When is a whacked ROA suspicious?

In Section 3.1 we discussed the risk that authorities in the RPKI can whack ROAs of their descendants, without causing collateral damage. One way to mitigate this risk is to *detect* them as they occur. RPKI objects are stored in public repositories, which makes detection possible. Recently, systems have been proposed [7, 14, 52] to allow the holder of a ROA to detect that when its ROA has become invalid. Detection by third parties can be more powerful, allowing RPKI relying parties to adjust their routing policies in real-time, without being informed by the holder of a (whacked) ROA. We thus classify RPKI manipulations that whack ROAs without collateral damage in terms of how easily they might be detected by third parties:

Clean: As discussed in Section 3 and Appendix A, if the target ROA (a) is issued by the manipulator or (b) is the grandchild of the manipulator and contains an address that overlaps none of its siblings, then the ROA can be whacked without collateral damage and without issuing suspicious new objects.

Suspicious: In all other cases, whacking a target ROAs without causing collateral damage may require the manipulator to introduce suspicious new objects into the RPKI, which is easier to detect. (The number of these new objects increases with the distance between the manipulator and the target—see Appendix A.)

To understand how effective third-party detection could be, we determine how often a ROA can be whacked cleanly. To do this, we must model future RPKI deployment in more detail than in Section B.2: it's no longer enough to know the top two layers of RCs (i.e., the RIRs and the recipients of direct allocations) and the ROAs at the leaves—we also need to know what's in between. We do this using the data from Section B.1. Note that in all the models below, the top two layers of RCs and the ROAs at the leaves are the same.

Entity	Prefix	#ASes	#ROAs	#(pfx, AS) tuples
AT&T	12.0.0.0/8	1073	2145	2567
Cogent	38.0.0.0/8	721	917	1055
Verizon	63.64.0.0/10	598	842	1020
Level3	8.0.0.0/8	413	740	2014

Table 12: Powerful recipients of direct alloc'ns

B.3.1 Models of the RPKI in full deployment.

Depending on how RPKI is adopted and how business relationships among ASes evolve, organization may choose to issue RCs to their customers or manage their ROAs for them instead.

"Hosted Model." This model has nothing in between the top two layers of RCs and the ROAs at the leaves. It supposes that most of the RPKI operation is outsourced to the RIRs, and is consistent with activity at RIRs like RIPE [14] and ARIN [11] and the expectation that many network operators will prefer not to have to learn how to become authorities [9]. This model presents a lower bound on the complexity and depth of the RPKI.

"RPKI Mania Model." This model is the opposite of the previous one: it assumes RCs are issued hierarchically whenever possible. For every directly-allocated IP prefix P designated as "allocated" in the data we obtained from the RIRs [1–5] ("allocated" means that sub-allocation to another organization is allowed), we built a subtree of RCs, with one RC for every prefix covered by P that had a (prefix, origin AS) pair seen in BGP data [6, 8]. The "ancestor" relationship in this subtree corresponds to the "cover" relationship for prefixes. We collapsed together parent-child pairs of RCs that were generated based on the same AS seen in BGP data. For each RC, there was also a corresponding ROA. When a directly-allocated prefix was marked as "assigned" by the RIRs (*i.e.,* sub-allocation was not allowed), we treated it as in the Hosted Model, issuing an RC for it and only ROAs directly below it.

"Non-stub CA Model." This model is slightly less hierarchical than the previous: it assumes that stubs (found according to [63] on 2013-04-12) will not choose to become authorities, and will thus not be issued RCs to manage their own ROAs. RCs generated because of stub ASes were eliminated from the previous model (if they were below layer two), and their children given to their parents. In addition, any RC issued to the same organization as its parent (inferred using using AS-to-org data [21]) was similarly eliminated. Note that the sample RPKI in Figure 2 is an excerpt of this model.

"Tier 1 CA Model." This model is even less hierarchical: it assumes that only Tier 1 ISPs will become authorities. Thus, all RCs not for a Tier 1 ISP (as classified by [63]) were eliminated from the previous model.

Remark. It is possible that in the future, National Internet Registries (NIRs) for individual countries could

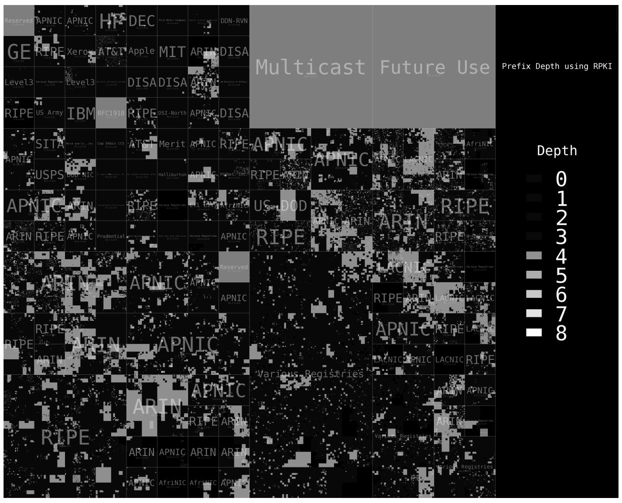


Figure 14: ROA depth in RPKI mania.

act as intermediate authorities between RIRs and direct allocations, or at deeper points in the RPKI hierarchy. Because it is currently unclear how this might be done, we did *not* incorporate NIRs into our models.

B.3.2 Results.

We can use the above models to measure how many RPKI manipulations can be done cleanly.

First, consider the case of manipulation by an RIR. In every model, at least 41% of ROAs (corresponding to 43% of the allocated IPv4 space) can be whacked cleanly by the RIRs.

Next, we consider the case of whacking by the recipient of a direct allocation (*i.e.,* the second layer of the hierarchy). In the Hosted Model, all the ROAs are at layer three, and therefore all of them can be whacked cleanly by recipients of direct allocations. In the Tier-1 CA model, 90% of the ROAs are at layer three; in the Non-Stub CA Model, 82% are at layer three; and in the RPKI Mania Model, 50% are at layer three. Moreover, even ROAs deeper than layer three can be whacked cleanly (*e.g.,* if they are the only children of parents at layer three), and so the numbers above present a conservative estimate of the number of ROAs that can be cleanly whacked by recipients of direct allocations. Summarizing, we see that even in Mania, the model that makes clean manipulations most difficult, the recipients of direct allocations can whack a large fraction of ROAs cleanly; in more realistic models, they can whack almost all the ROAs cleanly. Detection, while important, could be difficult.

B.4 Easy manipulations and little recourse.

Given that ROA whacking is most detectable when the manipulator is a great-grandparent of the target or beyond (see Section 3.1 and Appendix A), we conclude by looking at the depth of the RPKI certificate hierarchy. It turns out that the RPKI is quite shallow in

	Depth						
Model	3	4	5	6	7	8	9
Hosted	237,237	0	0	0	0	0	0
Tier 1 CA	212,503	24,269	464	1	0	0	0
Non-stub CA	193,679	41,652	1,886	20	0	0	0
Mania	118,028	108,043	10,863	293	9	1	0

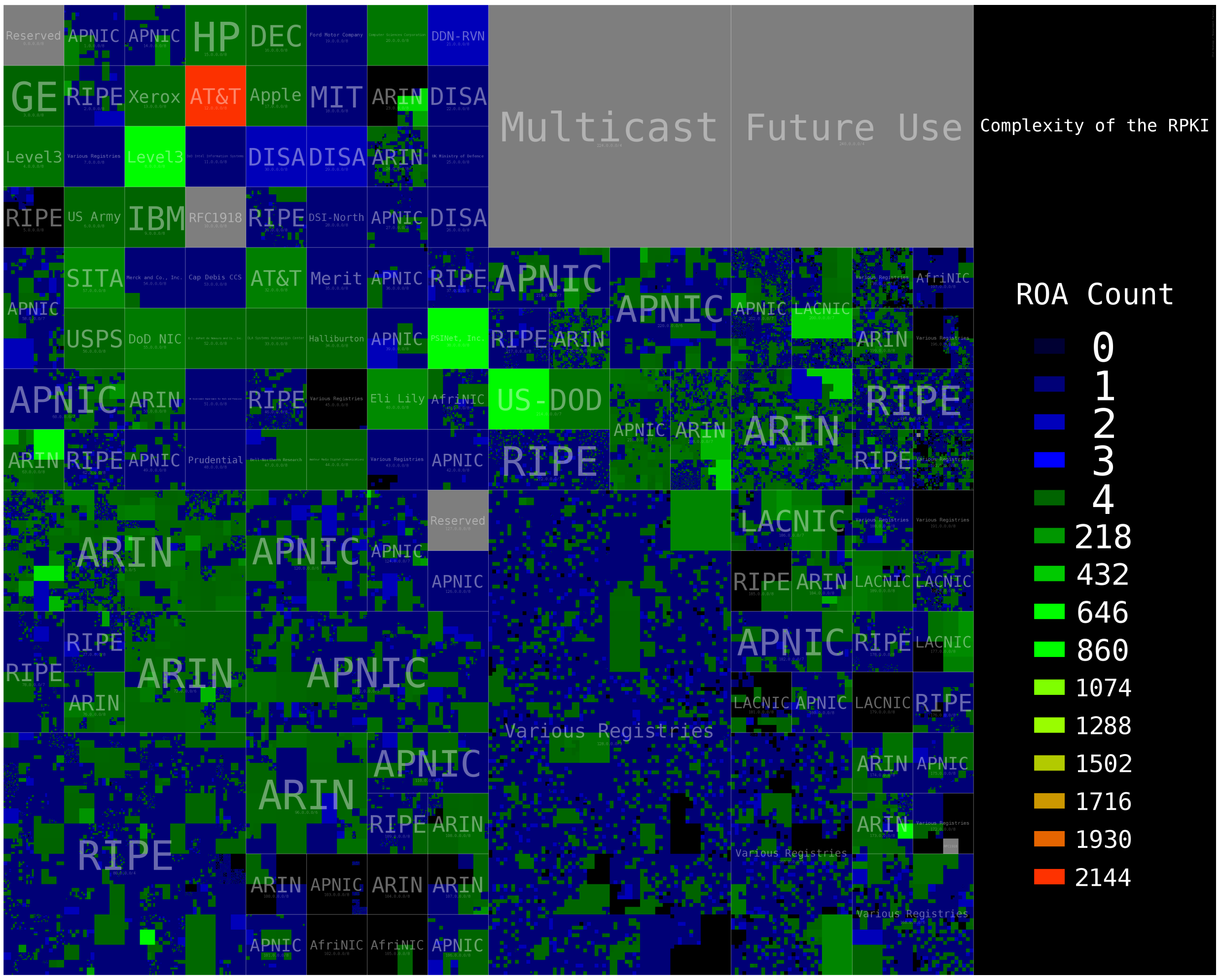
Table 15: ROA depth distributions.

all four models, as shown in Table 15. Even in RPKI mania, which has the deepest objects, a vast majority of ROAs are at layer three or four. This makes manipulations even more difficult to detect. We show depth distribution of ROAs in RPKI Mania the IPv4 space in Figure 14.

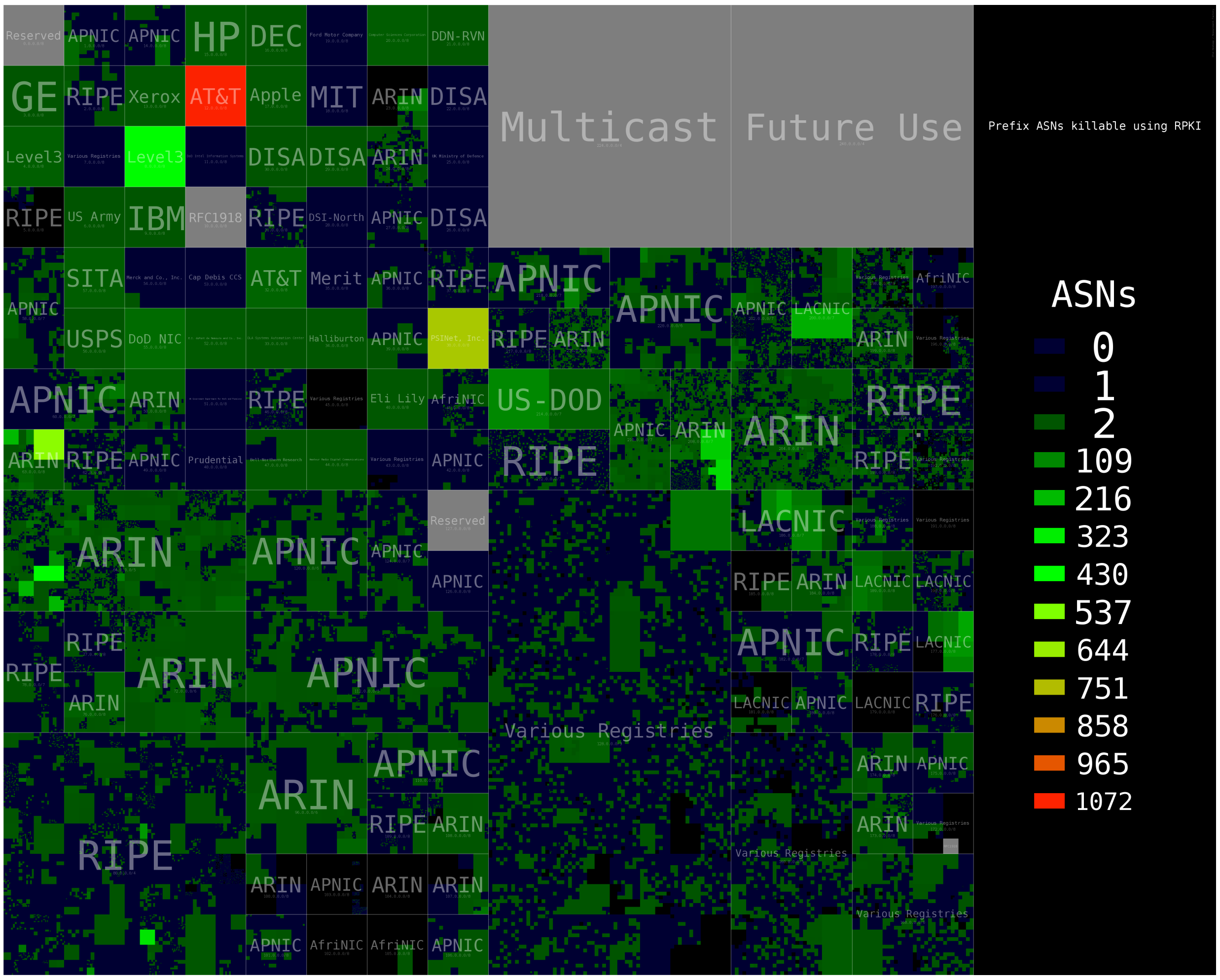
The shallowness of the RPKI has further implications. While more familiar PKIs (*e.g.,* the web PKI) allow an entity to have its certificates issued by *many* authorities, in the RPKI, any party that issues a ROA must hold an RC for superset of that prefix. In all models, most parties have only one authority (the recipient of a direct allocation) who can issue them a ROA; in fact, an overwhelming majority of parties have at most two. This could make it difficult for party C caught in a dispute with its parent authority to find an alternate authority that can issue a new ROA for C, a practice that the RFCs call "grandparenting" [17]. On the other hand, the shallowness of the RPKI also implies that, typically, only 2-3 parties have the ability to whack a given ROA.

C. FULL COLOR IPV4 HILBERT CURVES

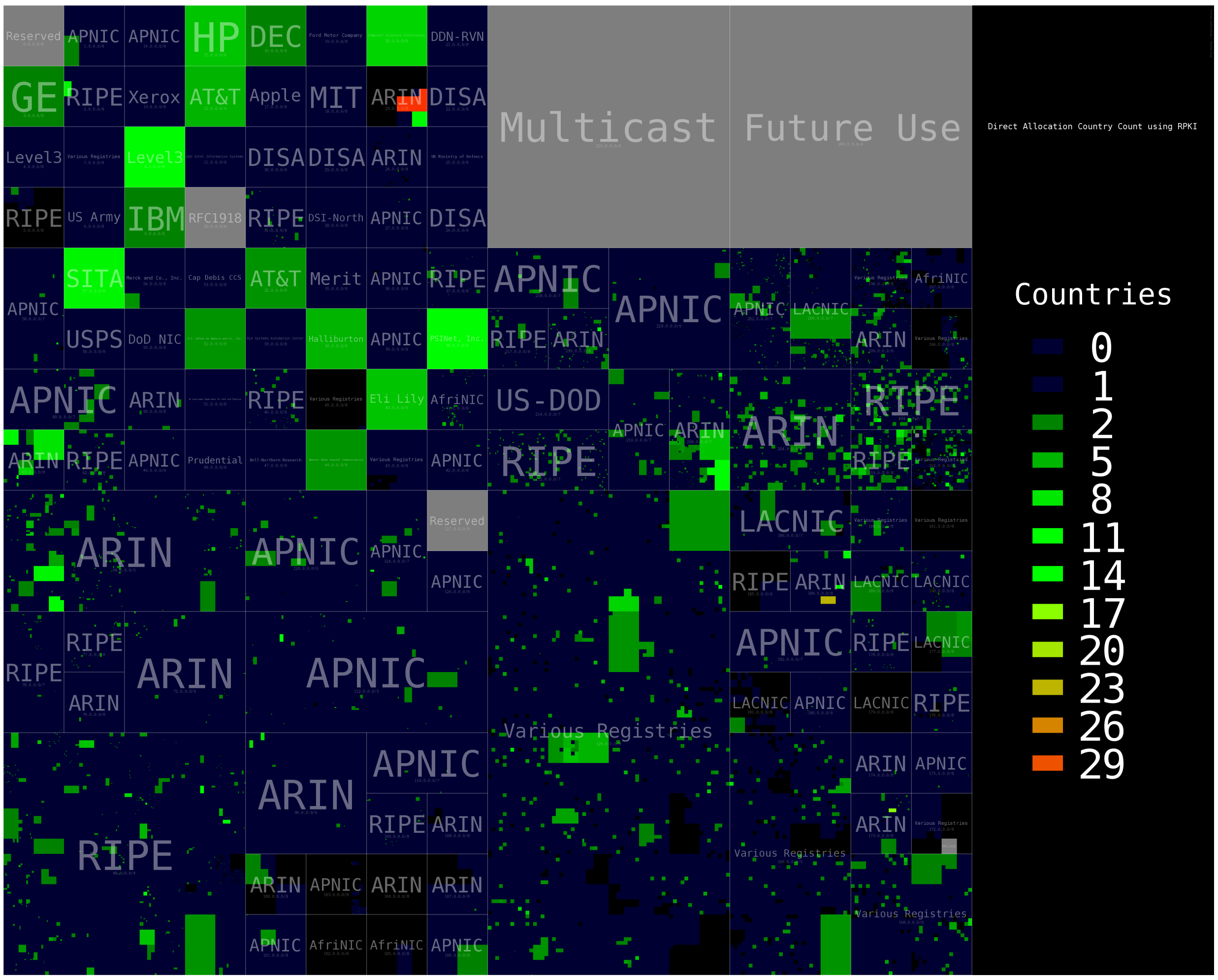
C.1 Number of ROAs that can be whacked by each direct allocation.



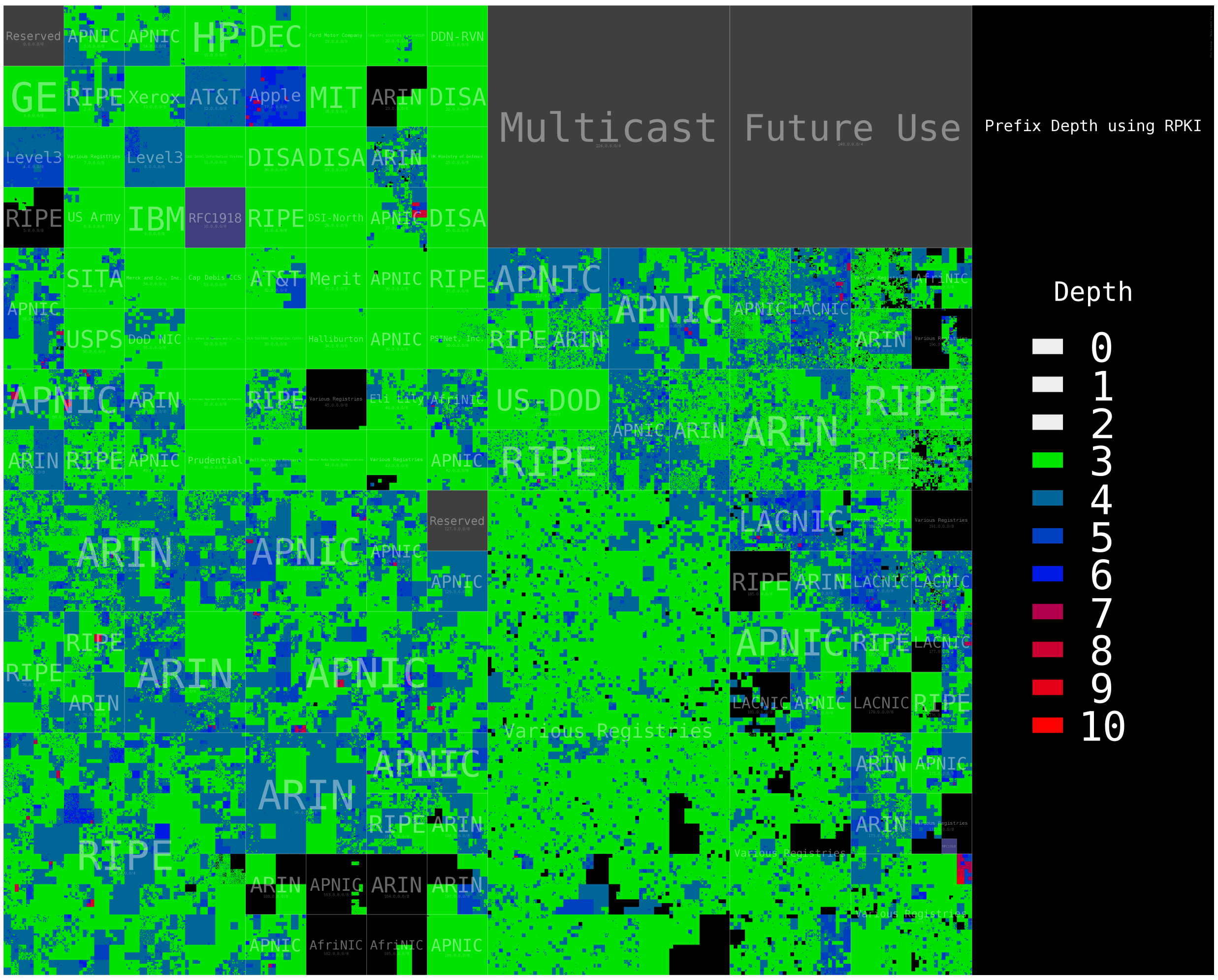
C.2 Number of ASes holding ROAs that can be whacked by each direct allocation.



C.3 The number of countries that hold ROAs that descend from each direct allocation.



C.4 ROA depth in the RPKI mania model


Multicast
Future Use
Prefix Depth using RPKI
Depth
Reserved
APNIC
HP
DEC
DDN-RVN
GE
RIPE
Xerox
AT&T
Apple
MIT
ARIN
DISA
Level3
IBM
RFC1918
US Army
DSI-North
SITA
Merit
USPS
DoD NIC
Halliburton
US-DOD
LACNIC
Prudential
AfriNIC
Various Registries