April 2014
Geoff Huston

A Reappraisal of Validation in the RPKI

I've often heard that security is hard. And good security is very hard. Despite the best of intentions, and the investment of considerable care and attention in the design of a secure system, sometimes it takes the critical gaze of experience to sharpen the focus and understand what's working and what's not. We saw this with the evolution of the security framework in the DNS, where it took multiple iterations over 10 or more years to come up with a DNSSEC framework that was able to gather a critical mass of acceptance. So before we hear cries that the deployed volume of RPKI technology means that its too late to change anything, let's take a deep breath and see what we've learned so far from this initial experience, and see if we can figure out what's working and what's not, and what we may want to reconsider.

RPKI - The Elevator Pitch

A few words to describe the "RPKI": The Resource Public Key Infrastructure (RPKI) is a security framework intended to add security credentials to the Internet's Inter-domain routing protocol, BGP. The intent is to allow a BGP speaker to determine the difference between an authentic advertisement of reachability information and a contrived lie. The basic security credentials are provided by a public key infrastructure that allows an IP address holder to associate a public key with their address holdings, and allows others to validate digitally signed attestations about IP addresses, AS numbers and their use in routing when made by the resource holder. This enables the information in routing advertisements to be digitally signed with an "authority", and for contrived routing information to be detectable through the lack of a necessary signed authority, or through the use of a signature on the authority that cannot be validated by a BGP speaker within the RPKI framework.

A more detailed explanation of the RPKI can be found in an article published in the Internet Protocol Journal (Vol14. No. 2, June 2011, http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_14-2/142_bgp.html). If reading RFCs is your favourite bedtime reading, then I can recommend RFC6480 as a riveting read about the RPKI.

So what's the problem with this technology? It's not as if we are lacking experience in digital cryptography, asymmetric cryptographic algorithms, and various forms of attestation framework that are intended to demonstrate potential trust vectors relating to the use of a public key. So surely we can get this right. Yes? Or maybe no. Let's look at an particular aspect of the RPKI and some of the issues that have arisen, and look at some of the ways that we could address these issues.

Validation and the Semantics of RPKI Certificates

To introduce this topic I should digress for a second and briefly and informally describe a digital signature. A digital signature is formed by taking a message that you want to sign, generating a hash of the message, then using your private key to encrypt this hash. The result is called a "digital signature". Validating a digital signature involves using the public key of the signer to decrypt the hash inside the signature, and then calculating your own hash of the message using the same hashing algorithm. If the hash values match then the receiver can be assured this is the message that was sent by the sender. However, I've glossed over one critical part here - the receiver needs to use the public key of the sender to validate the signature. Normally, the sender's public key would be attached to the message, after all the public key is just that - public. But how can the receiver be assured that it's the right public key? This is where public key certificates can help. If somebody you know and trust is willing to attest that they know the sender and can confirm the value of the sender's public key, then you can validate this public key via this attestation. Maybe the chain of referrals is a little longer, but the idea is still the same. These attestations take the form of "certificates", and the process of validating an entity's digital signature is akin to validating the end entity certificate that attests to the public key needed to validate the digital signature. The process of linking up a chain of attestations from someone you are willing to trust to the party whose digital signature you are validating is analogous to forming a certificate validation path from a trust anchor to the certificate being validated. The validation path is an ordered sequence of certificates starting with a known public key value associated with someone you trust (your "trust anchor"), and ending with the certificate being validated. Along the path, the n^{th}'s certificate's Subject name has to match the $n+1^{th}$ certificate's Issuer name, and of course the n^{th} certificate's Subject Public Key value has to match the $n+1^{th}$ certificate's Issuer Public Key value.

RPKI certificates are conventional public key certificates with one critical addition, in that each certificate contains a field that lists a collection of IP addresses and AS numbers. The intended semantics of this additional field is that the certificate issuer is no longer attesting that the individual named as the Subject of this certificate holds a private key whose matching public key part is recorded in this certificate, but that the issuer has allocated or assigned the IP addresses and AS numbers listed in the certificate to the entity who has the key pair whose public key part is identified in this certificate. This is no longer a conventional attestation relating to the subject's identity, role or some form of authorization. It is attesting a relationship between the holder of an IP address and a public key.

So how should we validate such certificates? What seems logical is to simply add another condition to the pairwise rules about matching Issuer/Subject names and public key values. What should this condition be? The approach used in the RPKI is to use a similar pairwise relationship in the validation path relating to these addresses, namely that the collection of IP addresses and AS numbers in the parent (Issuer's) certificate must be either the same, or fully encompass, the IP addresses and AS numbers in child (Subject's) certificate.

So what this looks like is that for an RPKI certificate to be "valid" the certificate must satisfy a number of criteria:

> The certificate must satisfy syntax correctness, validity dates, etc
> and there must exist an ordered sequence of certificates (1..n) where:
>
> - Certificate 1 is issued by a trust anchor
> - Certificate x's Subject Name value matches Certificate x+1's Issuer Name value
> - The resources in the Address extensions of Certificate x+1 must be "subsumed" by the Address extensions listed in Certificate x
> - Certificate 'n' is the certificate to be validated
> - Certificates 1 through n-1 are also "valid" according to this same criteria

The relationship of the resources between successive certificates in this validation path sequence is one of a sequence of enveloping sets, as shown in Figure 1.

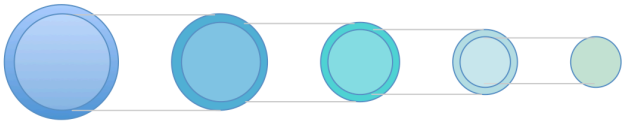

Figure 1 – Sequence of Enveloping Resource Sets

Lets look at the implications of this definition of validation. Figure 2 illustrates the validity of the certificate issued by C, using a validation path of 3 certificates as shown.

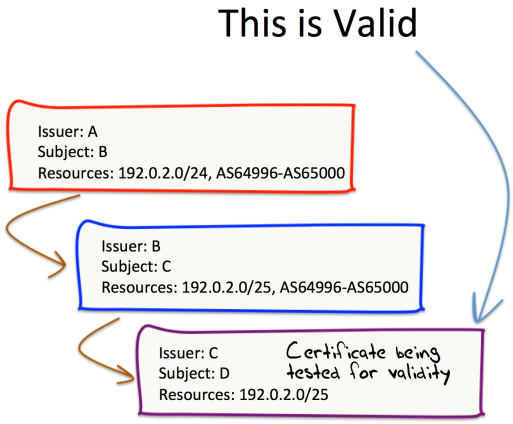

Figure 2 – Sequence of Enveloping Resource Sets

Now lets make a slight change to just one of these certificates. What we will not change is the certificate issued by C. Instead, we'll change the certificate issued by B, and add a couple of extra AS numbers to that certificate, as shown in Figure 3.

According to this specification of validity, the certificate issued by C is now invalid. It's not invalid by virtue of some inability to validate the IPv4 address prefix 192.0.2.0/25 in the chain of certificates from A to B to C, as that relationship fits within the requirements of a sequence of enveloping resources. Its invalid because the certificate issued by B now contains resources that are not contained in that issued by A (namely the range of Autonomous System numbers 65001 to 65011). Consequently, the certificate issued by B invalid and this implies that the certificate issued by C is also invalid. In this example, certificate B is invalid because it includes AS resources that are not found in its parent certificate, but equally this could occur with any type of number resource, be it IPv4 addresses, IPv6 addresses or AS numbers that are found in a certificate's resource extension, but not found in its parent certificate.

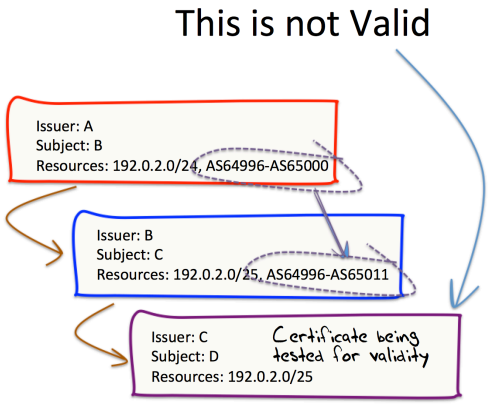

Figure 3 – Broken Sequence of Enveloping Resource Sets

This model of validation, when applied to the RIR model of resource distribution, has lead us to an intricate system where resource holders are required at times to maintain multiple certificates, and where the movement of resources between holders or across registries causes complex transitional states with a high degree of fragility. The consequences of number resource inconsistencies between issuer and subject at points high in the RPKI hierarchy, when applied into BGP for use in secure routing, admits the potential for catastrophically large routing failures though unintentional certificate invalidation. In the previous example, if A is the certificate issued by the IANA close to the root of the RPKI hierarchy, and B is a certificate issued by an RIR, then all subordinate certificates issued below that point in the hierarchy would also be invalid. If this subordinate certificate corresponds to a National Internet Registry, then its possible to envisage that the consequences of a small error at levels close to the apex of the certificate hierarchy could have large scale impact on a nation's routing infrastructure, or even at a regional level.

Clearly, this level of fragility would lead many to question the value of adopting a secure routing system. The tradeoff between additional security and additional fragility is an uneasy one, and often these tradeoffs are resolved in favour of robustness rather than security. Security is often associated with the assessment of risk, while fragility directly affects every actor's operating costs.

What can be done to mitigate such fragility in this system, where small inconsistencies in certificates high in the PKI system can cause widespread failures of validity in the descendant infrastructure? Of course we could simply insist that such errors in registry and certificate management system never happen. But demanding persistent absolute perfection from certificate management systems that are intended to reflect the contents of a diverse set of human-operated registries is a somewhat ludicrous proposition. And here the RPKI does not help, but actually hinders this absolute requirement for continuous perfection in all issued certificates. As an example of this, let's look at one such scenario, associated with the movement of a registration of an address between registries.

In this scenario, shown in Figure 4, an address prefix is being transferred from registry C to registry E.

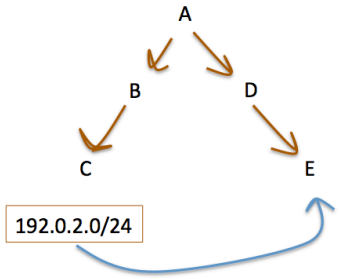

Figure 4 – A Resource Transfer

The objective is to ensure that the resources held by the subordinate registries B, C, D and E are not invalidated at any stage by actions of a superior registry. The required series of steps is to initially reissue certificates on the receiving side, using a top-down issuance sequence, and then perform a bottom-up reissuance sequence on the disposing side. The necessary sequencing of certificate actions is shown in Figure 5.

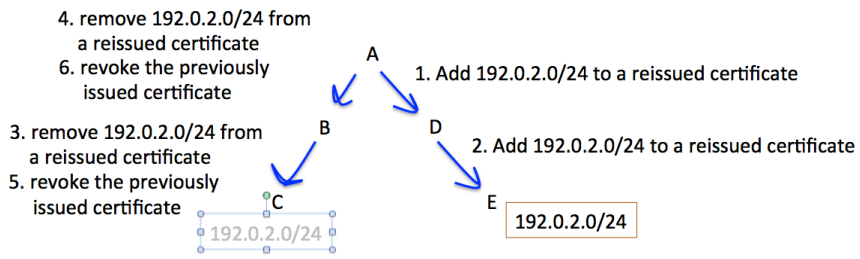

Figure 5 – Certificate Actions to support a Resource Transfer

To preserve the outcomes of the RPKI certificate validity condition its necessary that actions that result in a certificate with an expanded set of number resources be sequenced in a top-down fashion in the certificate hierarchy, and actions that result in the removal of resources be sequenced in a bottom up fashion. The potential for the generation of transient states that invalidate the resources of subordinate registries in this environment is significant, and the certificate provisioning protocol used by the RPKI (RFC6492) does not include specific signaling requirements to support this requirement for top-down and bottom-up sequencing of certificate actions.

One response to these vulnerabilities is to augment the existing set of procedures and mechanisms to address each use case that exposes this vulnerability. However, it may be more productive to look at this situation in more general terms. Is it possible to think about removing some aspects of this complexity within the RPKI framework, and also reducing the scope of consequential damage of certificate address collection mismatch?

Obviously this is a very difficult question to answer. By its very nature, a trust system based on hierarchies imputes a massive level of onus on the correct operation of the system at points close to, and at, the apex of the hierarchy. One possible approach is to remove the hierarchy and contemplate various web of trust models, and look at trust as some statistical process. But this seems less than satisfactory from many view points. Is it feasible to contemplate models that retain a hierarchical model of trust, but at the same time address this concern?

The critical assumption that underpins the RPKI validation algorithm is that the set of resources described in the address extensions in these certificates are an immutable "blob". If we assume that this is an aspect that is fixed for the RPKI, then one way to reduce the extent of impacts of inconsistencies in the certificate hierarchy is to reduce the scope of the resources listed in any individual certificate, and rather than issue a certificate for a subject that describes the entirety of their held address resources, one could envisage a certificate framework where a distinct certificate was issued for every individual AS number, and at its most extreme case, for every individual IP address (Figure 6). Obviously, particularly when we consider IPv6, this thought experiment is completely impractical and the volume of certificates would overwhelm all practical forms of certificate management and local cache operation, and possibly a few major laws of physics as well. However, the impacts of any inconsistency between Issuer and Subject in this completely unaggregated framework would be limited to the particular addresses where there is a difference, and would not generate a large volume of potential collateral damage to any other address resources, where no such inconsistency has occurred.

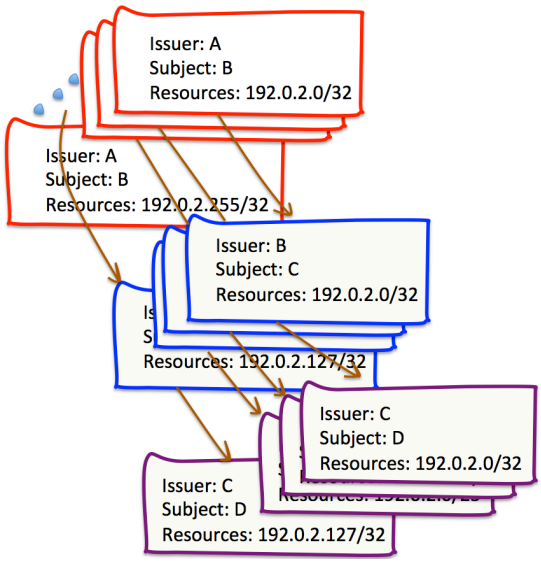

Figure 6 – A "devolved" RPKI Certificate structure

Can we retain the efficiency of using certificates with maximal spanning resources sets in order to reduce the volume of issued certificates in the RPKI system, while at the same time defining a system behavior that is comparable to that of a certificate structure that uses a completely unaggregated set of resources?

One possible response is to redefine a resource certificate as a separable collection of IP addresses and AS numbers that share a common cryptographic set of credentials. In other words, a single certificate with the resource extension that included the IP addresses 192.0.2.0/24 and AS 64500 could be regarded as semantically equivalent to two certificates, one listing the IP addresses of 192.0.2.0/24 and the other listing AS 64500, with all other fields of these two certificates being identical to the fields of the first original certificate. Notionally we could take this thought experiment one step further and breakdown the certificate listing 192.0.2.0/24 into 256 distinct certificates, namely 192.0.2.0/32, 192.0.2.1/32 and so on, again with all other fields of these certificates being identical.

What does this interpretation of an RPKI certificate imply in terms of the RPKI? Most of the RPKI framework is unaltered: An issuer would still issue "aggregate" certificates to entities who hold resources in the Issuer's registry, where, as far as is possible, the resources in the certificate are the complete set of resources held by the subject that have been assigned or allocated by this Issuer. This is the same as the current RPKI interpretation, so the underlying semantics of certificates in the context of the RPKI is unaltered. Indeed most of the RPKI need not be altered in any substantive way whether we consider RPKI certificates as a bundled set of cryptographic information and a "blob" of address resources, or as a common set of cryptographic information that applies to a separable collection of address resources.

However, there is one point in the RPKI framework where this subtle change in semantics is important. The point of change lies in the definition of validity in the RPKI. Rather than asking whether *a certificate is "valid"*, which assumes that what we are asking is whether the certificate is valid for the complete set of address resources listed in the certificate's address resource extension, we can break out of that "all or nothing" nature of the certificate validity test and ask instead *for what address resources can a certificate be considered "valid"?*

What this looks like is that for a resource to be considered as "valid" with respect to a given RPKI certificate, the certificate must satisfy a number of criteria:

The certificate must satisfy syntax correctness, validity dates, etc
and there must exist an ordered sequence of certificates (1..n) where:

- Certificate 1 is issued by a trust anchor
- Certificate x's Subject Name value matches Certificate x+1's Issuer Name value
- The resources in the Address extensions of Certificate x must "subsume" the address resource given in the validity question
- Certificate 'n' is the certificate to be validated
- Certificates 1 through n-1 are also "valid" according to this same criteria

The relationship of the resources between successive certificates in this validation path sequence is one of a sequence of resource sets, each of which encompass the resource in question, as shown in Figure 7.

Figure 7 – Common Intersecting Sequence of Resource Sets

This alteration to the RPKI validity definition offers improved robustness to the RPKI by limiting the impact of mismatches in the address resource extensions within the certificate hierarchy. Looking at the example from Figure 3 above, we can apply this altered validation algorithm to produce the result shown in Figure 7. For the resource 192.0.2.0/25, as the resource can be found in each of the certificates, then, assuming that these certificates are otherwise valid, then we can conclude that certificate C is valid for the resource 192.0.2.0/25.

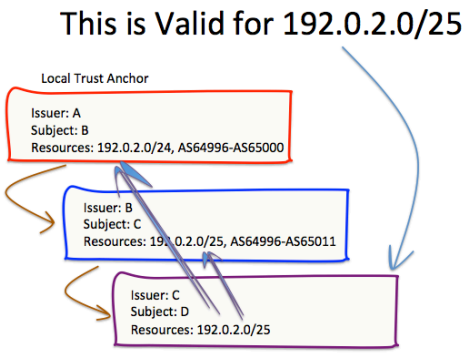

Figure 8 – Revised RPKI Validation Process

Applying this validation model to the example of resource transfer, the strict requirement for top down signaling of the augmented certificate is no longer necessary, nor is the requirement for bottom up processing of certificate re-issuance requests along the path where the resource is being removed. Each of the CAs can re-issue certificates within their own timings, and the only resource that is affected by this uncoordinated set of certificate actions is the resource being transferred. All other resources remain valid throughout this process.

The more general observation is that by using this interpretation of an RPKI certificate as a separable collection of resources with a common set of associated cryptographic material, inconsistencies in the resource sets within any given validation path do not invalidate the entirety of the resources listed in any certificate. Instead, the "damage" arising from such inconsistencies is limited to the set of resources that are listed in a certificate, but are not listed at higher levels in the path from the certificate in question to the peak of the certificate hierarchy. While this does not limit the potential for such inconsistencies to occur, it limits the extent of any collateral damage that may occur when such inconsistencies occur, and goes a long way in reducing the operational fragility that sits within the current RPKI framework.

Taking a Further Step

If a resource in a certificate is to be considered valid if we can establish a certification path from this certificate to a trust anchor where all the certificates along this path contain this resource, as described above, then what can we say about the set of all such resources that can be validated through this certificate?

In particular, the question being asked is: is it strictly necessary for the certification path to be identical in all cases for each of the resources listed in the certificate? Are we necessarily constrained to use the same trust anchor and the same validation path to validate every resource listed in the certificate? If we interpret an RPKI certificate as a collection of resources with a common cryptographic "wrapper", then is it reasonable to allow relying parties to validate various resources listed in this collection using distinct validation paths?

The reason why this question is raised is that the effort to make the certificate system replicate the existing address registry framework has come up with some unforeseen outcomes that seem to increase the complexity of the system.

The original model of address allocation in the RIR system was intended to be quite simple. IANA allocated blocks of numbers to each of the RIRs (a "block" is defined as a /8 in IPv4, a /12 in IPv6 and 1024 AS numbers). IANA recorded these block assignments in its registry, detailing the data of the assignment and the RIR who received this block. The RIR then made assignments from this block, recording the recipients of these allocations and assignments in its registry. So if you were wanting to find a registry entry for a given address, then you should consult the IANA registry to find encompassing block, which would provide the RIR, and then consult the RIR's registry to find the end user. And most of the time this works. However, some of the time the recipient has moved, and the registry entry has been transferred to a different RIR. There are some 8,020 allocations that fall into this category, where the RIR that holds the authoritative registry entry for an allocation or assignment is not the RIR that received the encompassing number block allocation from the IANA (out of a total of some 213,836 registry records, as of March 2014).

This can lead to a situation shown in Figure 9, where a single entity holds a collection of resources that are derived from distinct IANA address blocks that were originally allocated to different RIRs. When resources are transferred from one RIR registry to another, the certification framework needs to reflect this, hence the requirement for the two "bridging" certificates issued by RIRs B and C, that reflect the moment of these addresses into the registry operated by RIR A. However, if the validation paths for these resources need to be distinct, we end up with the end entity holding three different certificates.

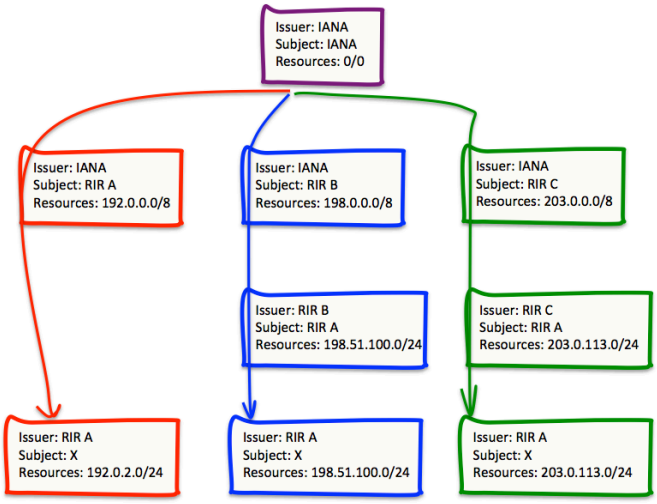

Figure 9 – Diverse Certificate Paths

However, it is feasible to simplify this situation, by using the modified validation algorithm that interprets a resource certificate as a separable collection of resources with a common set of cryptographic credentials.

In this case RIR A uses one key pair to sign distinct Certificate Signing Requests (CSR) that are submitted to its superior RIRs, as well as to the IANA. The RIR operates a certificate authority using this key pair ("RIR A" in Figure 10). The RIR then generates a second key pair to sign a single CSR for the entirety of the resources managed by this RIR, and operates a second certificate authority using this second key pair ("RIR A-Working" in Figure 9). It is then possible for the RIR to issue a single certificate for subordinate entities that will describe the entirety of the resources held by such entities listed in the RIR's registry (Figure 10).

The general principle here is that reducing the number of moving parts in any machine reduces the potential for failure. By reducing the number of terminal certificates within the RPKI framework we can reduce the potential for errors, and restore a simple principle that the end user needs only a single certificate to describe the entirety of their resource holdings.

This framework of "joining" elements in the certification hierarchy is consistent with the RPKI validation process described above. Each of the resources in the certificate with Subject X can be validated by the formation of a certificate path from this certificate to the IANA self-signed certificate (which is, presumably, a Relying Party's trust anchor). The definition of validation does not treat the resources in the certificate as a bound bundle, and this admits the possibility that different resources in the certificate could be validated through distinct certification paths.

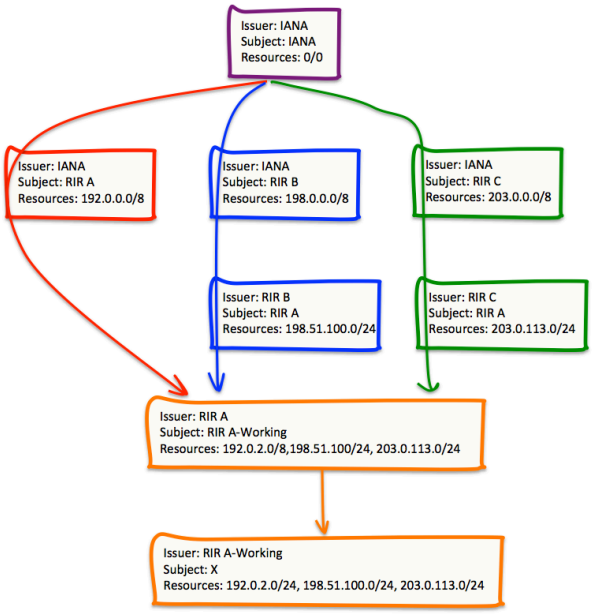

Figure 10 – Amalgamated Certificate Paths

Implementing this form of validation is not a major change to the local cache management algorithms. The same form of top-down certificate management functions is used, but rather than applying a simple "encompassing" rule, the algorithm uses "intersections" instead.

However, this has some elements in certificate structure that start to look a little more like a web of trust structure, as distinct from a strict hierarchy. In Figure 10 the certificate issued by RIR A is constructed using the elements of three distinct certificates issued by IANA, RIR B and RIR C. In effect RIR A has to present the same certificate signing request to multiple CAs, complete with the same subject name as well as the same public key value. The CA will still undertake the same procedure as the current RPKI system, namely that the issued certificate accurately reflects the state of resources held by the subject within the number registry operated by the CA. The "borrowing" of the web of trust concept occurs at the next level of the certificate hierarchy, where the subordinate entity (RIR A in this case) effectively forms a single CA and casts the entirety of its registry holdings as the resources associated with this CA. It can then create a new subordinate CA (RIR A-Working) with a single certificate that includes the entirety of the resources held in RIR A's registry, and holders of resources in RIR A's registry also can receive a single certificate issued by RIR A-Working that reflect the entirety of the holder's resources in the registry, irrespective of their provenance.

A pseudo code implementation of a top-down local repository cache management implementation that illustrates this form of chained intersection is shown at http://www.potaroo.net/ispcol/2014-04/local-cache.txt. It uses a two step pass through the certificate infrastructure, but this is merely a programming convenience, as the same outcome can be achieved in a single pass across a local cache of the RPKI repository.

However, its not clear that this is a step further in improving the robustness of the RPKI system, or whether this represents a step too far. Over time the simplicity of the original resource management model of the IANA, the RIRs and local registries has seen additional complexity added in response to

the movement of registry details between RIRs and potentially between LIRs. In crafting a model of resource certification which is based on the foundations of this registry framework, one view is that it is a necessary outcome that details in the registries are accurately reflected in details of issued certificates, and outcomes of a single entity with a single relationship with one RIR may still be issued with multiple certificates to reflect the entirety of their resource holdings (as illustrated in Figure 9). Another view is that the certificates should reflect the entirety of a relationship of an entity with its registry, and that the complexities of inter-registry resource movement are reflected in the certificate validation process, and not in the certificates per se. This leads to mechanisms such as that shown in Figure 10. As to whether this is a step too far, through re-introducing the potential for new failure states and unintended consequences, then that is a matter for further consideration.

Conclusions

This article has attempted to expose some of the shortcomings in the current RPKI framework, and consider some approaches that can mitigate these shortcomings. It's certainly not the intention here to claim that the RPKI is all so fundamentally misguided that we would be better off looking at an entirely different routing security framework. Not at all. But it is the case that we should be thinking about the level of fragility and unintended exposure to collateral damage that exists in the current hierarchical RPKI framework, and thinking about ways to improve this situation.

The foundation of the RPKI system, that issued certificates are based on mirroring the state of the address distribution framework and its associated registries, where the party who assigned or allocated a resource is the party who is the issuer of an RPKI certificate and the recipient of the address assignation is the certificate's subject, is a sound foundation. The certificate structure is intended to be a replica of the address registry structure, where individual certificates correspond to entries in the address registries, and the registry operator is the certification authority. Nothing in this foundation changes in that respect in these slightly altered certificate validation frameworks.

However, as we've explored above, the use of the currently specified validation algorithm introduces a certain level of fragility into the process, and as one gets close to the apex of the certificate hierarchy, the smallest error in an issued certificate may have widespread consequences in the routing system, and potentially invalidate large collections of route information, and thereby fracture the Internet in possibly catastrophic ways. Obviously this level of fragility and consequent risk is disturbing for operators of these registries.

Perhaps there are other validation algorithms that retain the essential semantics of a certificate, but do not embrace a "all or nothing" validation outcome with its attendant liabilities to the operators of registries high in the certificate hierarchy. While its always preferable to avoid making errors in the first place, perhaps its possible to limit the consequences of certain forms of errors, replacing some aspects of the fragility of this system with measures that limit the scope of the consequence damage to the routing system.

Disclaimer

The above views do not necessarily represent the views or positions of the Asia Pacific Network Information Centre.

Author

Geoff Huston B.Sc., M.Sc., is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region. He has been closely involved with the development of the Internet for many years, particularly within Australia, where he was responsible for the initial build of the Internet within the Australian academic and research sector. He is author of a number of Internet-related books, and was a member of the Internet Architecture Board from 1999 until 2005, and served on the Board of Trustees of the Internet Society from 1992 until 2001.

www.potaroo.net