Internet Engineering Task Force (IETF)
Request for Comments: 8416
Category: Standards Track
ISSN: 2070-1721

D. Ma
ZDNS
D. Mandelberg
Unaffiliated
T. Bruijnzeels
NLnet Labs

August 2018

Simplified Local Internet Number Resource Management with the RPKI (SLURM)

Abstract

The Resource Public Key Infrastructure (RPKI) is a global authorization infrastructure that allows the holder of Internet Number Resources (INRs) to make verifiable statements about those resources. Network operators, e.g., Internet Service Providers (ISPs), can use the RPKI to validate BGP route origin assertions. ISPs can also use the RPKI to validate the path of a BGP route. However, ISPs may want to establish a local view of exceptions to the RPKI data in the form of local filters and additions. The mechanisms described in this document provide a simple way to enable INR holders to establish a local, customized view of the RPKI, overriding global RPKI repository data as needed.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8416.

Table of Contents

1. Introduction 3
   1.1. Terminology 4
2. RP with SLURM 4
3. SLURM Files and Mechanisms 5
   3.1. Use of JSON 5
   3.2. SLURM File Overview 5
   3.3. Validation Output Filters 6
        3.3.1. Validated ROA Prefix Filters 6
        3.3.2. BGPsec Assertion Filters 7
   3.4. Locally Added Assertions 9
        3.4.1. ROA Prefix Assertions 9
        3.4.2. BGPsec Assertions 10
   3.5. Example of a SLURM File with Filters and Assertions 11
4. SLURM File Configuration 13
   4.1. SLURM File Atomicity 13
   4.2. Multiple SLURM Files 13
5. IANA Considerations 14
6. Security Considerations 14
7. References 14
   7.1. Normative References 14
   7.2. Informative References 16
Acknowledgments 17
Authors' Addresses 17

1. Introduction

The Resource Public Key Infrastructure (RPKI) is a global authorization infrastructure that allows the holder of Internet Number Resources (INRs) to make verifiable statements about those resources. For example, the holder of a block of IP(v4 or v6) addresses can issue a Route Origin Authorization (ROA) [RFC6482] to authorize an Autonomous System (AS) to originate routes for that block. Internet Service Providers (ISPs) can then use the RPKI to validate BGP routes. (Validation of the origin of a route is described in [RFC6811], and validation of the path of a route is described in [RFC8205].)

However, an RPKI Relying Party (RP) may want to override some of the information expressed via configured Trust Anchors (TAs) and the certificates downloaded from the RPKI repository system. For instance, [RFC6491] recommends the creation of ROAs that would invalidate public routes for reserved and unallocated address space, yet some ISPs might like to use BGP and the RPKI with private address space (see [RFC1918], [RFC4193], and [RFC6598]) or private AS numbers (see [RFC1930] and [RFC6996]). Local use of private address space and/or AS numbers is consistent with the RFCs cited above, but such use cannot be verified by the global RPKI. This motivates creation of mechanisms that enable a network operator to publish, at its discretion, an exception to the RPKI in the form of filters and additions (for its own use and that of its customers). Additionally, a network operator might wish to make use of a local override capability to protect routes from adverse actions [RFC8211], until the results of such actions have been addressed. The mechanisms developed to provide this capability to network operators are hereby called "Simplified Local Internet Number Resource Management with the RPKI (SLURM)".

SLURM allows an operator to create a local view of the global RPKI by generating sets of assertions. For origin validation [RFC6811], an assertion is a tuple of {IP prefix, prefix length, maximum length, Autonomous System Number (ASN)} as used by the RPKI-Router protocol, version 0 [RFC6810] and version 1 [RFC8210]. For BGPsec [RFC8205], an assertion is a tuple of {ASN, subject key identifier, router public key} as used by version 1 of the RPKI-Router protocol. (For the remainder of this document, these assertions are called "ROA Prefix Assertions" and "BGPsec Assertions", respectively.)

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. RP with SLURM

SLURM provides a simple way to enable an RP to establish a local, customized view of the RPKI, overriding RPKI repository data if needed. To that end, an RP with SLURM filters out (i.e., removes from consideration for routing decisions) any assertions in the RPKI that are overridden by local ROA Prefix Assertions and BGPsec Assertions.

In general, the primary output of an RP is the data it sends to routers over the RPKI-Router protocol [RFC8210]. The RPKI-Router protocol enables routers to query an RP for all assertions it knows about (Reset Query) or for an update of only the changes in assertions (Serial Query). The mechanisms specified in this document are to be applied to the result set for a Reset Query and to both the old and new sets that are compared for a Serial Query. RP software may modify other forms of output in comparable ways, but that is outside the scope of this document.

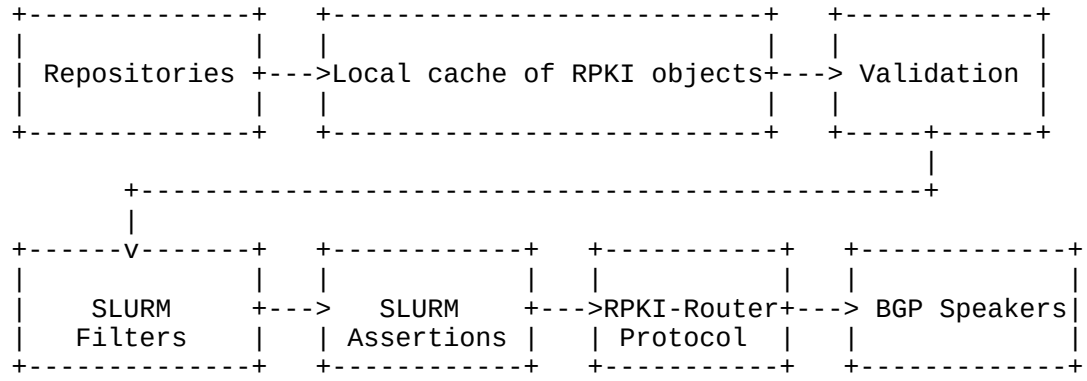

Figure 1: SLURM's Position in the RP Stack

3. SLURM Files and Mechanisms

3.1. Use of JSON

SLURM filters and assertions are specified in JSON format [RFC8259]. JSON members that are not defined here MUST NOT be used in SLURM files. An RP MUST consider any deviations from the specifications to be errors. Future additions to the specifications in this document MUST use an incremented value for the "slurmVersion" member.

3.2. SLURM File Overview

A SLURM file consists of a single JSON object containing the following members:

- A "slurmVersion" member that MUST be set to 1, encoded as a number
- A "validationOutputFilters" member (Section 3.3), whose value is an object. The object MUST contain exactly two members:
  - A "prefixFilters" member, whose value is described in Section 3.3.1.
  - A "bgpsecFilters" member, whose value is described in Section 3.3.2.
- A "locallyAddedAssertions" member (Section 3.4), whose value is an object. The object MUST contain exactly two members:
  - A "prefixAssertions" member, whose value is described in Section 3.4.1.
  - A "bgpsecAssertions" member, whose value is described in Section 3.4.2.

In the envisioned typical use case, an RP uses both Validation Output Filters and Locally Added Assertions. In this case, the resulting assertions MUST be the same as if output filtering were performed before locally adding assertions; that is, Locally Added Assertions MUST NOT be removed by output filtering.

The following JSON structure with JSON members represents a SLURM file that has no filters or assertions:

```
{
  "slurmVersion": 1,
  "validationOutputFilters": {
    "prefixFilters": [],
    "bgpsecFilters": []
  },
  "locallyAddedAssertions": {
    "prefixAssertions": [],
    "bgpsecAssertions": []
  }
}
```

Figure 2: Empty SLURM File

3.3. Validation Output Filters

3.3.1. Validated ROA Prefix Filters

The RP can configure zero or more Validated ROA Prefix Filters ("Prefix Filters" for short). Each Prefix Filter can contain either an IPv4 or IPv6 prefix and/or an ASN. It is RECOMMENDED that an explanatory comment is included with each Prefix Filter so that it can be shown to users of the RP software.

The above is expressed as a value of the "prefixFilters" member, as an array of zero or more objects. Each object MUST contain either 1) one of the following members or 2) one of each of the following members.

- A "prefix" member, whose value is a string representing either an IPv4 prefix (see Section 3.1 of [RFC4632]) or an IPv6 prefix (see [RFC5952]).

- An "asn" member, whose value is a number.

In addition, each object MAY contain one optional "comment" member, whose value is a string.

The following example JSON structure represents a "prefixFilters" member with an array of example objects for each use case listed above:

```
"prefixFilters": [
  {
    "prefix": "192.0.2.0/24",
    "comment": "All VRPs encompassed by prefix"
  },
  {
    "asn": 64496,
    "comment": "All VRPs matching ASN"
  },
  {
    "prefix": "198.51.100.0/24",
    "asn": 64497,
    "comment": "All VRPs encompassed by prefix, matching ASN"
  }
]
```

Figure 3: "prefixFilters" Examples

Any Validated ROA Payload (VRP) [RFC6811] that matches any configured Prefix Filter MUST be removed from the RP's output.

A VRP is considered to match with a Prefix Filter if one of the following cases applies:

1. If the Prefix Filter only contains an IPv4 or IPv6 prefix, the VRP is considered to match the filter if the VRP prefix is equal to or covered by the Prefix Filter prefix.

2. If the Prefix Filter only contains an ASN, the VRP is considered to match the filter if the VRP ASN matches the Prefix Filter ASN.

3. If the Prefix Filter contains both an IPv4 or IPv6 prefix and an ASN, the VRP is considered to match if the VRP prefix is equal to or covered by the Prefix Filter prefix and the VRP ASN matches the Prefix Filter ASN.

3.3.2. BGPsec Assertion Filters

The RP can configure zero or more BGPsec Assertion Filters ("BGPsec Filters" for short). Each BGPsec Filter can contain an ASN and/or the Base64 [RFC4648] encoding of a Router Subject Key Identifier (SKI), as described in [RFC8209] and [RFC6487]. It is RECOMMENDED that an explanatory comment is also included with each BGPsec Filter, so that it can be shown to users of the RP software.

The above is expressed as a value of the "bgpsecFilters" member, as an array of zero or more objects. Each object MUST contain one of either, or one each of both following members:

- An "asn" member, whose value is a number

- An "SKI" member, whose value is the Base64 encoding without trailing '=' (Section 5 of [RFC4648]) of the certificate's Subject Key Identifier as described in Section 4.8.2 of [RFC6487]. (This is the value of the ASN.1 OCTET STRING without the ASN.1 tag or length fields.)

In addition, each object MAY contain one optional "comment" member, whose value is a string.

The following example JSON structure represents a "bgpsecFilters" member with an array of example objects for each use case listed above:

```
"bgpsecFilters": [
  {
    "asn": 64496,
    "comment": "All keys for ASN"
  },
  {
    "SKI": "<Base 64 of some SKI>",
    "comment": "Key matching Router SKI"
  },
  {
    "asn": 64497,
    "SKI": "<Base 64 of some SKI>",
    "comment": "Key for ASN 64497 matching Router SKI"
  }
]
```

Figure 4: "bgpsecFilters" Examples

Any BGPsec Assertion that matches any configured BGPsec Filter MUST be removed from the RP's output. A BGPsec Assertion is considered to match with a BGPsec Filter if one of the following cases applies:

1. If the BGPsec Filter only contains an ASN, a BGPsec Assertion is considered to match if the Assertion ASN matches the Filter ASN.

2. If the BGPsec Filter only contains an SKI, a BGPsec Assertion is considered to match if the Assertion Router SKI matches the Filter SKI.

3. If the BGPsec Filter contains both an ASN and a Router SKI, then a BGPsec Assertion is considered to match if both the Assertion ASN matches the Filter ASN and the Assertion Router SKI matches the Filter SKI.

3.4. Locally Added Assertions

3.4.1. ROA Prefix Assertions

Each RP is locally configured with a (possibly empty) array of ROA Prefix Assertions ("Prefix Assertions" for short). Each ROA Prefix Assertion MUST contain an IPv4 or IPv6 prefix and an ASN. It MAY include a value for the maximum length. It is RECOMMENDED that an explanatory comment is also included with each so that it can be shown to users of the RP software.

The above is expressed as a value of the "prefixAssertions" member, as an array of zero or more objects. Each object MUST contain one of each of the following members:

- A "prefix" member, whose value is a string representing either an IPv4 prefix (see Section 3.1 of [RFC4632]) or an IPv6 prefix (see [RFC5952]).
- An "asn" member, whose value is a number.

In addition, each object MAY contain one of each of the following members:

- A "maxPrefixLength" member, whose value is a number.
- A "comment" member, whose value is a string.

The following example JSON structure represents a "prefixAssertions" member with an array of example objects for each use case listed above:

```
  "prefixAssertions": [
    {
      "asn": 64496,
      "prefix": "198.51.100.0/24",
      "comment": "My other important route"
    },
    {
      "asn": 64496,
      "prefix": "2001:DB8::/32",
      "maxPrefixLength": 48,
      "comment": "My other important de-aggregated routes"
    }
  ]
```

Figure 5: "prefixAssertions" Examples

Note that the combination of the prefix, ASN, and optional maximum length describes a VRP as described in [RFC6811]. The RP MUST add all Prefix Assertions found this way to the VRP found through RPKI validation and ensure that it sends the complete set of Protocol Data Units (PDUs), excluding duplicates when using the RPKI-Router protocol (see Sections 5.6 and 5.7 of [RFC8210]).

3.4.2. BGPsec Assertions

Each RP is locally configured with a (possibly empty) array of BGPsec Assertions. Each BGPsec Assertion MUST contain an AS number, a Router SKI, and the router public key. It is RECOMMENDED that an explanatory comment is also included so that it can be shown to users of the RP software.

The above is expressed as a value of the "bgpsecAssertions" member, as an array of zero or more objects. Each object MUST contain one each of all of the following members:

- An "asn" member, whose value is a number.
- An "SKI" member, whose value is the Base64 encoding without trailing '=' (Section 5 of [RFC4648]) of the certificate's Subject Key Identifier as described in Section 4.8.2 of [RFC6487] (This is the value of the ASN.1 OCTET STRING without the ASN.1 tag or length fields.)
- A "routerPublicKey" member, whose value is the Base64 encoding without trailing '=' (Section 5 of [RFC4648]) of the equivalent to the subjectPublicKeyInfo value of the router certificate's public key, as described in [RFC8208]. This is the full ASN.1 DER encoding of the subjectPublicKeyInfo, including the ASN.1 tag and length values of the subjectPublicKeyInfo SEQUENCE.

The following example JSON structure represents a "bgpsecAssertions" member with one object as described above:

```
"bgpsecAssertions": [
  {
    "asn": 64496,
    "SKI": "<some base64 SKI>",
    "routerPublicKey": "<some base64 public key>",
    "comment": "My known key for my important ASN"
  }
]
```

Figure 6: "bgpsecAssertions" Examples

Note that a "bgpsecAssertions" member matches the syntax of the Router Key PDU described in Section 5.10 of [RFC8210]. Relying Parties MUST add any "bgpsecAssertions" member thus found to the set of Router Key PDUs, excluding duplicates, when using the RPKI-Router protocol [RFC8210].

3.5. Example of a SLURM File with Filters and Assertions

The following JSON structure represents an example of a SLURM file that uses all the elements described in the previous sections:

```
{
  "slurmVersion": 1,
  "validationOutputFilters": {
    "prefixFilters": [
      {
        "prefix": "192.0.2.0/24",
        "comment": "All VRPs encompassed by prefix"
      },
      {
        "asn": 64496,
        "comment": "All VRPs matching ASN"
      },
      {
        "prefix": "198.51.100.0/24",
        "asn": 64497,
        "comment": "All VRPs encompassed by prefix, matching ASN"
      }
    ],
    "bgpsecFilters": [
      {
        "asn": 64496,
        "comment": "All keys for ASN"
      },
      {
        "SKI": "Zm9v",
        "comment": "Key matching Router SKI"
      },
      {
        "asn": 64497,
        "SKI": "YmFy",
        "comment": "Key for ASN 64497 matching Router SKI"
      }
    ]
  },
  "locallyAddedAssertions": {
    "prefixAssertions": [
      {
```

```
            "asn": 64496,
            "prefix": "198.51.100.0/24",
            "comment": "My other important route"
          },
          {
            "asn": 64496,
            "prefix": "2001:DB8::/32",
            "maxPrefixLength": 48,
            "comment": "My other important de-aggregated routes"
          }
        ],
        "bgpsecAssertions": [
          {
            "asn": 64496,
            "comment" : "My known key for my important ASN",
            "SKI": "<some base64 SKI>",
            "routerPublicKey": "<some base64 public key>"
          }
        ]
      }
    }
```

Figure 7: Example of Full SLURM File

4. SLURM File Configuration

4.1. SLURM File Atomicity

To ensure local consistency, the effect of SLURM MUST be atomic. That is, the output of the RP either MUST be the same as if a SLURM file were not used or MUST reflect the entire SLURM configuration. For an example of why this is required, consider the case of two local routes for the same prefix but different origin ASNs. Both routes are configured with Locally Added Assertions. If neither addition occurs, then both routes could be in the NotFound state [RFC6811]. If both additions occur, then both routes would be in the Valid state. However, if one addition occurs and the other does not, then one could be Invalid while the other is Valid.

4.2. Multiple SLURM Files

An implementation MAY support the concurrent use of multiple SLURM files. In this case, the resulting inputs to Validation Output Filters and Locally Added Assertions are the respective unions of the inputs from each file. The envisioned typical use case for multiple files is when the files have distinct scopes. For instance, operators of two distinct networks may resort to one RP system to frame routing decisions. As such, they probably deliver SLURM files to this RP independently. Before an RP configures SLURM files from different sources, it MUST make sure there is no internal conflict among the INR assertions in these SLURM files. To do so, the RP SHOULD check the entries of each SLURM file with regard to overlaps of the INR assertions and report errors to the sources that created the SLURM files in question. The RP gets multiple SLURM files as a set, and the whole set MUST be rejected in case of any overlaps among the SLURM files.

If a problem is detected with the INR assertions in these SLURM files, the RP MUST NOT use them and SHOULD issue a warning as error report in the following cases:

1. There may be conflicting changes to ROA Prefix Assertions if an IP address X and distinct SLURM files Y and Z exist such that X is contained by any prefix in any "prefixAssertions" or "prefixFilters" in file Y and X is contained by any prefix in any "prefixAssertions" or "prefixFilters" in file Z.

2. There may be conflicting changes to BGPsec Assertions if an ASN X and distinct SLURM files Y and Z exist such that X is used in any "bgpsecAssertions" or "bgpsecFilters" in file Y and X is used in any "bgpsecAssertions" or "bgpsecFilters" in file Z.

5. IANA Considerations

This document has no IANA actions.

6. Security Considerations

The mechanisms described in this document provide a network operator with additional ways to control use of RPKI data while preserving autonomy in address space and ASN management. These mechanisms are only applied locally; they do not influence how other network operators interpret RPKI data. Nonetheless, care should be taken in how these mechanisms are employed. Note that it also is possible to use SLURM to (locally) manipulate assertions about non-private INRs, e.g., allocated address space that is globally routed. For example, a SLURM file may be used to override RPKI data that a network operator believes has been corrupted by an adverse action. Network operators who elect to use SLURM in this fashion should use extreme caution.

The goal of the mechanisms described in this document is to enable an RP to create its own view of the RPKI, which is intrinsically a security function. An RP using a SLURM file is trusting the assertions made in that file. Errors in the SLURM file used by an RP can undermine the security offered to that RP by the RPKI. A SLURM file could declare as invalid ROAs that would otherwise be valid, and vice versa. As a result, an RP MUST carefully consider the security implications of the SLURM file being used, especially if the file is provided by a third party.

Additionally, each RP using SLURM MUST ensure the authenticity and integrity of any SLURM file that it uses. Initially, the SLURM file may be preconfigured out of band, but if the RP updates its SLURM file over the network, it MUST verify the authenticity and integrity of the updated SLURM file. The mechanism to update the SLURM file to guarantee authenticity and integrity is out of the scope of this document.

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4632] Fuller, V. and T. Li, "Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", BCP 122, RFC 4632, DOI 10.17487/RFC4632, August 2006, <https://www.rfc-editor.org/info/rfc4632>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <https://www.rfc-editor.org/info/rfc5952>.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, DOI 10.17487/RFC6487, February 2012, <https://www.rfc-editor.org/info/rfc6487>.

[RFC6811] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein, "BGP Prefix Origin Validation", RFC 6811, DOI 10.17487/RFC6811, January 2013, <https://www.rfc-editor.org/info/rfc6811>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8205] Lepinski, M., Ed. and K. Sriram, Ed., "BGPsec Protocol Specification", RFC 8205, DOI 10.17487/RFC8205, September 2017, <https://www.rfc-editor.org/info/rfc8205>.

[RFC8208] Turner, S. and O. Borchert, "BGPsec Algorithms, Key Formats, and Signature Formats", RFC 8208, DOI 10.17487/RFC8208, September 2017, <https://www.rfc-editor.org/info/rfc8208>.

[RFC8209] Reynolds, M., Turner, S., and S. Kent, "A Profile for BGPsec Router Certificates, Certificate Revocation Lists, and Certification Requests", RFC 8209, DOI 10.17487/RFC8209, September 2017, <https://www.rfc-editor.org/info/rfc8209>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

7.2. Informative References

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <https://www.rfc-editor.org/info/rfc1918>.

[RFC1930] Hawkinson, J. and T. Bates, "Guidelines for creation, selection, and registration of an Autonomous System (AS)", BCP 6, RFC 1930, DOI 10.17487/RFC1930, March 1996, <https://www.rfc-editor.org/info/rfc1930>.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <https://www.rfc-editor.org/info/rfc4193>.

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, DOI 10.17487/RFC6482, February 2012, <https://www.rfc-editor.org/info/rfc6482>.

[RFC6491] Manderson, T., Vegoda, L., and S. Kent, "Resource Public Key Infrastructure (RPKI) Objects Issued by IANA", RFC 6491, DOI 10.17487/RFC6491, February 2012, <https://www.rfc-editor.org/info/rfc6491>.

[RFC6598] Weil, J., Kuarsingh, V., Donley, C., Liljenstolpe, C., and M. Azinger, "IANA-Reserved IPv4 Prefix for Shared Address Space", BCP 153, RFC 6598, DOI 10.17487/RFC6598, April 2012, <https://www.rfc-editor.org/info/rfc6598>.

[RFC6810] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol", RFC 6810, DOI 10.17487/RFC6810, January 2013, <https://www.rfc-editor.org/info/rfc6810>.

[RFC6996] Mitchell, J., "Autonomous System (AS) Reservation for Private Use", BCP 6, RFC 6996, DOI 10.17487/RFC6996, July 2013, <https://www.rfc-editor.org/info/rfc6996>.

[RFC8210] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol, Version 1", RFC 8210, DOI 10.17487/RFC8210, September 2017, <https://www.rfc-editor.org/info/rfc8210>.

[RFC8211] Kent, S. and D. Ma, "Adverse Actions by a Certification Authority (CA) or Repository Manager in the Resource Public Key Infrastructure (RPKI)", RFC 8211, DOI 10.17487/RFC8211, September 2017, <https://www.rfc-editor.org/info/rfc8211>.

Acknowledgments

The authors would like to thank Stephen Kent for his guidance and detailed reviews of this document. The authors would also like to thank Richard Hansen for his careful reviews and Hui Zou and Chunlin An for their editorial assistance.

Authors' Addresses

Di Ma
ZDNS
4 South 4th St. Zhongguancun
Haidian, Beijing 100190
China

Email: madi@zdns.cn

David Mandelberg
Unaffiliated

Email: david@mandelberg.org
URI: https://david.mandelberg.org

Tim Bruijnzeels
NLnet Labs
Science Park 400
Amsterdam 1098 XH
The Netherlands

Email: tim@nlnetlabs.nl