Network Working Group R. Reddy
Internet-Draft National Security Agency
Intended status: Informational C. Wallace
Expires: March 15, 2008 Cygnacom Solutions
September 12, 2007

Trust Anchor Management Problem Statement
draft-wallace-ta-mgmt-problem-statement-02

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on March 15, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2007).

Abstract

This document provides a problem statement for the Trust Anchor Management Birds of a Feather (BOF). A trust anchor represents an authoritative entity via a public key and associated data. The public key is used to verify digital signatures and the associated data is used to constrain the types of information for which the trust anchor is authoritative. Relying parties use trust anchors to determine if digitally signed objects are valid by verifying digital signatures using the trust anchor's public key and by enforcing the constraints expressed in the associated data. This document describes some of the problems associated with the lack of a standard trust anchor management mechanism as well as problems that must be addressed by such a mechanism.

Table of Contents

1. Introduction 3
   1.1. Terminology 4
2. Problem Statement 5
3. Functional Properties 7
4. Security Considerations 9
5. IANA Considerations 11
6. References 12
   6.1. Normative References 12
   6.2. Informative References 12
Authors' Addresses 13
Intellectual Property and Copyright Statements 14

1. Introduction

Digital signatures are used in many applications. For digital signatures to provide integrity and authentication, the public key used to verify the digital signature must be trusted. A public key used to verify a signature must be configured as a trust anchor or contained in a certificate that can be transitively verified by a certification path terminating at a trust anchor. Directly trusted public keys are known as trust anchors. A Trust Anchor is a public key and associated data used by a relying party (RP) to validate a signature on a signed object where the object is either:

- a public key certificate that begins a certification path terminated by a signature certificate or encryption certificate

- a non-public key certificate object that cannot be validated via use of a certification path

Trust anchors have local significance, i.e., each RP is configured with a set of trust anchors, either by the RP or by an entity that manages TAs in the context in which the RP operates. The associated data often is used to define the scope of a trust anchor, by imposing constraints on the signatures it may be used to verify. For example, if a trust anchor is used to verify signatures on X.509 certificates, these constraints may include a combination of name spaces, certificate policies, or application/usage types. Whenever a signature is verified, a trust anchor must be used, either by verifying the signature directly or by validating a certification path.

One particular use of digital signatures is the verification of signatures on firmware packages loaded into hardware modules, such as cryptographic modules, cable boxes, routers, etc. Since such devices are often managed remotely, the devices must be able to authenticate the source of management interactions and can use trust anchors to perform this authentication. However, trust anchors require management as well.

All applications that rely upon digital signatures must have some means of managing one or more sets of trust anchors. These sets of trust anchors are referred to in this document as trust anchor stores. Often, the means of managing trust anchor stores are application-specific and rely upon out-of-band means to establish and maintain trustworthiness. Applications may use multiple trust anchor stores and a given trust anchor store may be used by multiple applications. Trust anchor stores are managed by trust anchor managers.

In some cases, a hardware device may have a single trust anchor that is hard-wired or managed only through physical access to the device. However, to support the ability to delegate different functions to different authorities, the device may require multiple trust anchors. It is desirable to manage those trust anchors using similar means as software updates, certificate requests, etc. to enable code reuse.

1.1. Terminology

The following terms are defined in order to provide a vocabulary for describing requirements for trust anchor management.

Trust Anchor: A trust anchor represents an authoritative entity via a public key and associated data. The public key is used to verify digital signatures and the associated data is used to constrain the types of information for which the trust anchor is authoritative. Relying parties use trust anchors to determine if digitally signed objects are valid by verifying digital signatures using the trust anchor's public key and by enforcing the constraints expressed in the associated data.

Trust Anchor Manager: A trust anchor manager is responsible for managing the contents of a trust anchor store.

Trust Anchor Store: A trust anchor store is a set of one or more trust anchors.

2. Problem Statement

Trust anchors are used to support many application scenarios. Most Internet browsers and email clients use trust anchors to authenticate TLS sessions, to verify signed email and to generate encrypted email. Many software distributions are digitally signed to enable authentication of the originator to be performed prior to installation. Trust anchors that support these applications are typically installed as part of the operating system or application, installed using an enterprise configuration management system or installed directly by the application user.

In some devices, trust anchors are initially installed in the device in a secure manner with no means of managing the trust anchor store in a non-secure environment.

Trust anchors are typically stored in application- or operating system- specific trust anchor stores. Often, a single machine may have a number of different trust anchor stores that may or may not be synchronized (or need to be synchronized). Reviewing the contents of a particular trust anchor store typically involves the use of a proprietary tool that interacts with a particular type of trust store.

The mere presence of a trust anchor in a particular store often conveys implicit authorization to validate signatures for any contexts from which the store is accessed. For example, the public key of a timestamp authority (TSA) may be installed in a trust anchor store to validate signatures on timestamps. However, if the trust anchor store is used by multiple applications that serve different purposes, the same key may be used to validate other types of objects such as certificates or OCSP responses. There is currently no standard means of limiting the applicability (scope) of a trust anchor.

Trust relationships between PKIs are negotiated by policy authorities. Negotiations frequently require significant time to ensure all participating parties' requirements are satisfied. These requirements are expressed, to some extent, in public key certificates. In order for these requirements to be enforced, trust anchor stores must be managed in accord with policy authority intentions.

Trust anchors are often represented as self-signed certificates, which provide no useful means of establishing the validity of the information contained in the certificate. Confidence in the integrity of a trust anchor is typically established through out-of-band means, often by checking the "fingerprint" of the self-signed

certificate with an authoritative source. Routine trust anchor re-key operations typically require similar out-of-band checks. Ideally, only the initial set of trust anchors installed in a particular trust anchor store should require out-of-band verification, particularly when the costs of performing out-of-band checks commensurate with the security requirements of applications using the trust anchor store are high.

Despite the prevalent use of trust anchors, there is neither a standard means for reporting which trust anchors installed in a particular trust anchor store nor a standard means of managing those trust anchors. The remainder of this document describes some of the functional characteristics a solution to this problem should exhibit along with some security considerations.

3. Functional Properties

A general-purpose solution for the management of trust anchors must be transport independent in order to apply to a range of device communications environments. It should also be applicable in both session-oriented and store-and-forward contexts. At a minimum, it must enable a trust anchor manager to add trust anchors to, remove trust anchors from and determine which trust anchors are installed in a particular trust anchor store.

Trust anchor configurations may be uniform across an enterprise, or they may be unique to a single application or small set of applications. Management transactions, therefore, may be generic, targeted to groups of trust anchor stores, or targeted to individual trust anchor stores.

Once installed into a trust anchor store, a trust anchor represents an entity with authority recognized by applications that use that store. It is important to be able to define the scope of authority assigned to each trust anchor, which may be very specific (e.g., a trust anchor public key may be limited to verification of firmware updates only), or more general (such as to validate certification paths for certificates issued to users or devices). It should be possible to authorize a trust anchor to delegate authority and to prevent delegation.

Trust anchor managers have significant control over a device or application due to the ability to control what other authorities are recognized. As such, trust anchor managers are likely to be associated with the legal owner of the device or application in an enterprise setting or an agency authority for government devices. The trust anchor manager may be static over the life of a device, or it may change as legal ownership or other factors change. A trust anchor management protocol should enable secure transfer of a device from one trust anchor manager to another as well as delegation over specific aspects of the device without delegation of the overall trust anchor management capability itself. Trust anchor re-key is one type of transfer that must be supported.

A trust anchor management protocol must be capable of managing trust anchors that can be used to validate certification paths in accordance with [RFC3280]. Minimally, the definition of a trust anchor must include a public key, a public key algorithm and, if necessary, public key parameters. When the public key is used to validate certification paths, a distinguished name also must be included. A public key identifier should be included to enable other applications of the trust anchor, for example, verification of data signed using the Cryptographic Message Syntax SignedData structure

[RFC3852].

In some scenarios, a public key may be explicitly trusted for some purposes, but not trusted for use in validating certification paths. A trust anchor management protocol must enable the management of trust anchors that do not serve as trust anchors for certification path validation. For example, a public key may be used only for verification of signed firmware packages [RFC4108].

Connections between PKIs can be accomplished using different means. Unilateral or bilateral cross-certification can be performed, or a community may simply elect to explicitly trust the trust anchor from another community. Typically, these decisions occur at the enterprise level. In some scenarios, it can be useful to establish these connections for a small community within an enterprise. Enterprise-wide mechanisms such as cross-certificates are ill-suited for this purpose since certificate revocation or expiration affects the entire enterprise. A trust anchor management protocol can address this issue by supporting managed installation of trust anchors, or more tightly controlled trust list management capabilities within the enterprise. Managed installation requires the ability to identify the members of the community that are authorized to rely upon a particular trust anchor, as well as the ability to query and report on the contents of trust anchor stores.

There is no common format for trust anchors. A trust anchor management protocol should support various representations of trust anchors to simplify management across a range of application scenarios. Examples of trust anchor formats include self-signed X.509 certificates, Open PGP certificates [RFC2440] or DNSSEC trust anchors. [RFC3280] does not mandate a particular trust anchor representation, and requires only that a trust anchor public key information and a distinguished name be available during certification path validation.

A trust anchor manager must be able to authenticate which device produced a report listing the trust anchors that comprise a trust anchor store and be able to confirm the contents of the report have not been subsequently altered. Undetectable replay of old reports must not be possible.

A trust anchor definition should enable the representation of constraints that influence certification path validation or otherwise establish the scope of usage of the trust anchor public key. Examples of such constraints are name constraints, certificate policies and key usage. Trust anchor managers must be able to establish the constraints associated with any particular trust anchor.

4. Security Considerations

The integrity of trust anchor management transactions must be assured and it must be possible to authenticate the originator of a transactions and confirm the originator is authorized for that transaction.

Traditionally, trust anchors are distributed out-of-band with integrity mechanisms checked manually prior to installing a trust anchor. Installation is performed by anyone with sufficient administrative privilege on the system receiving the trust anchor. A trust anchor management protocol should enable integrity to be checked automatically by relying upon a public key that is resident on the client system participating in the protocol. The ability to manage the trust anchor store can be transformed into the ability to engage in the trust anchor management protocol with remote control of the trust anchor store contents being possible.

The public key used to authenticate the trust anchor management transactions may have been placed on the client as the result of an earlier transaction or during an initial bootstrap configuration operation. In most scenarios, at least one public key authorized for trust anchor management must be placed in each trust store to be managed during the initial configuration of the trust store. This public key may be transported and checked using traditional out-of-band means. In all scenarios, regardless of the authentication mechanism, at least one trust anchor manager must be established for each trust store during the initial configuration of the trust store.

An entity receiving trust anchor information must be able to authenticate the party providing the information and be able to confirm the party is authorized to provide trust anchor information. A trust anchor may be authorized to participate in trust anchor management protocol exchanges but limited to managing trust anchors within a particular scope. Alternatively, a trust anchor may be authorized to participate in trust anchor management protocol exchanges without any constraints on the types of trust anchors that may be managed. Clear subordination rules must be defined.

Some devices that utilize trust anchors have no access to a reliable source of time. Trust anchor management transactions should enable such devices to obtain trust anchor information without being subject to replay attacks that could add old or no-longer-trusted trust anchors to a trust anchor store.

Compromise of a private key corresponding to a trust anchor can have significant negative consequences. A trust anchor management protocol must include strategies to enable recovery from the

compromise of a trust anchor private key, including the private key authorized to serve as a source of trust anchor information.

Reliance on unauthorized trust anchors is the primary threat that must be countered by a trust anchor management protocol.

5. IANA Considerations

None. Please remove this section prior to publication as an RFC.

6. References

6.1. Normative References

[RFC3280] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.

6.2. Informative References

[RFC2440] Callas, J., Donnerhacke, L., Finney, H., and R. Thayer, "OpenPGP Message Format", RFC 2440, November 1998.

[RFC3852] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 3852, July 2004.

[RFC4108] Housley, R., "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages", RFC 4108, August 2005.

Authors' Addresses

Raksha Reddy
National Security Agency

Email: r (dot) reddy (at) radium (dot) ncsc (dot) mil

Carl Wallace
Cygnacom Solutions
Suite 5200
7925 Jones Branch Drive
McLean, VA 22102

Email: cwallace@cygnacom.com

Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).