Abstract

This specification defines an optimal format for pregenerated Online Certificate Status Protocol (OCSP) response database, that can be generated by a keyed OCSP responder and used by a keyless OCSP responder to serve OCSP queries.
1. Introduction

The Online Certificate Status Protocol [2] specifies a mechanism used to determine the status of digital certificates, without requiring CRLs. Since its definition in 1999, it has been deployed in a variety of environments and has proven to be a useful certificate status checking mechanism.

In addition, OCSP has been deployed in caching mode in large sized PKIs to address scalability issues. In such an architecture, a small number of keyed servers continually pregenerate large databases of OCSP responses. These OCSP response databases are then transported and used by a larger number of keyless responders to serve OCSP requests.

Due to lack of a common OCSP response database format specification, signing servers that presigned OCSP responses cannot interoperate with caching servers that make use of presigned OCSP responses, unless they agree on a proprietary format.

This specification defines a format for pregenerated OCSP response databases, that can be generated by a signing server and used by a caching server.

Since the count of pregenerated OCSP responses for a CA may run into millions, a goal for this format is to be efficient in terms of space requirements.

1.1  Definitional Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [1].

1.2  General Terminology

It is hoped that the implementers of this specification will adopt the following terminology in describing offerings to potential users or customers.

OCSP Responder  A server that provides OCSP responses in reply to OCSP queries. An OCSP responder can be keyless or keyed. A keyless server provides cached responses. A keyed responder provides freshly signed responses.
Signing Server
A keyed OCSP responder. It can provide live signed responses and can also create a database of signed OCSP responses.

Caching Server
A keyless OCSP responder. It provides OCSP responses signed by a keyed responder. It can either provide a response by a lookup in a database or by relaying the request to a signing server.

OCSP Database
A normalized representation of a collection of OCSP responses.
2. OCSP Database structure

The OCSP database structure is an optimal representation of a
database of basic OCSP responses. It contains a template OCSP
response that contains the common fields of all OCSP responses in the
database. The fields that differ are represented by
basicResponseInfoList.

Responders pregenerating OCSP databases MUST generate one OCSP
database per Certificate Authority (CA). An OCSP database must NOT
contain responses for certificates issued by multiple CAs.

The OCSP database MUST have the following syntax:

OCSPDatabase ::= SEQUENCE {
  templateOCSPResponse     BasicOCSPResponse,
  basicResponseInfoList    SEQUENCE OF BasicOCSPResponseInfo
}

The fields in the OCSPDatabase structure are defined in the following
sections.

2.1 templateResponse

The templateResponse item specifies the common elements in all basic
OCSP responses in the database. The definition of BasicOCSPResponse
is imported from [2].

2.1.1 responseExtensions

Responders compliant to this specification MUST not include the nonce
extension.

2.1.2 thisUpdate, nextUpdate and producedAt

When pre-producing OCSPResponse messages, the responder MUST set the
thisUpdate, nextUpdate and producedAt times as follows:

thisUpdate:          The time at which the status being indicated is
                    known to be correct.
nextUpdate:         The time at or before which newer information
                    will be available about the status of the
certificate. Responders MUST always include this
                    value.
producedAt:         The time at which the OCSP response is signed.
The basicResponseInfoList defines the list of BasicOCSPResponseInfo object. The basicResponseInfoList contains one BasicOCSPResponseInfo item for every BasicOCSPResponse contained in the database. BasicOCSPResponseInfo object combined with the CommonOCSPData creates a complete basic OCSP Response.

The basicResponseInfoList uses the BasicOCSPResponseInfo type, which has the following syntax:

\[
\text{OcspBasicResponseInfo ::= SEQUENCE }
\begin{array}{l}
\text{serialStatusList } \text{SEQUENCE OF SerialStatus,}
\text{signature } \text{BIT STRING}
\end{array}
\]

The basicResponseInfoList must be sorted in ascending order based on the serialNumber of the first SerialStatus item in serialStatusList.

The fields in BasicOCSPResponseInfo structure are defined in the following sections.

2.2.1 serialStatusList

serialStatusList uses the SerialStatus type, which has the following syntax:

\[
\text{SerialStatus ::= SEQUENCE }
\begin{array}{l}
\text{serialNumber } \text{CertificateSerialNumber,}
\text{certStatus } \text{CertStatus}
\end{array}
\]

The serialStatusList contains one SerialStatus item for every SingleResponse contained in the BasicOCSPResponse.

The responder sets the fields in SerialStatus as follows:

- serialNumber: This is set to the value of SingleResponse.certID.serialNumber. The definition of SingleResponse is imported from [2].
- certStatus: The value of certStatus is set to the value of SingleResponse.certStatus.

The SerialStatus structures contained in the serialStatusList MUST be
sorted in ascending order of serialNumber value.

2.2.2 signature

The value of signature is set to the value of signature item from the BasicOCSPResponse represented by this BasicOCSPResponseInfo structure.

3. References


Author’s Address

Piyush Jain
Tumbleweed Communications Corp.
700 Saginaw Drive
Redwood City, CA  94063
US

Phone: +1 650 216 2547
Email: piyush.jain@tumbleweed.com
Intellectual Property Statement

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.

Disclaimer of Validity

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 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.

Copyright Statement

Copyright (C) The Internet Society (2005). 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.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.