International

Virtual

Observatory

Alliance

A Proposal for Digital Signatures in VOEvent Messages
Version 1.1

IVOA Note 2008 May 14

Previous version(s):
Version 1.0 2008 March 13
Author(s):
Robert B. Denny

Abstract

The advantages of digitally signing VOEvent messages are well understood. This paper presents a proposal for accomplishing this with standards-based technology and having no impact on VOEvent message formats, minimal programming requirements, and an adaptable trust model.

Status of this Document

This is a Note. The first release of this document was 2008 March 13. The 1.1 release contains clarifications of key usage for signing and validation.

This is an IVOA Note expressing suggestions from and opinions of the authors. It is intended to share best practices, possible approaches, or other perspectives on interoperability with the Virtual Observatory. It should not be referenced or otherwise interpreted as a standard specification.

A list of current IVOA Recommendations and other technical documents can be found at http://www.ivoa.net/Documents/.

Acknowledgments

The author would like to thank Alasdair Alan of the eStar project (University of Exeter, UK) for his assistance in getting started with VOEvent, as well as Roy Williams of Caltech and Rob Seaman of NOAO, without whose encouragement this Note would not have been published.

Contents

---

1. Introduction

At the Hot Wiring the Transient Universe workshop in July, 2008 (attended by the author) there was some discussion about security. S. L. Allen gave a talk VOEvent Authentication via XML digital signatures (since published as a paper, see References). Attendees generally agreed that there is a need to at least authenticate VOEvent messages in order to verify their source and integrity. Encryption was not a significant consideration. Allen's paper covers the issues and technology well. Advantages of using signed VOEvent messages include assurances that

- no changes have occurred (or been made) to the message,
- the message truly came from the originator, and thus
- no 3rd party can forge a message from the originator.

Since the workshop, discussions have continued, focusing mainly on enveloped versus detached signatures, canonicalization, aggregation of VOEvent messages to reduce traffic, and whether to sign individual messages in an aggregate. Also being discussed is the usefulness of nested signatures, with some being applied during transit of the message.

Throughout those discussions, it has been assumed that XML Signature and the X.509 (PKIX) certificate and trust model would be used. The author feels that, for completeness, at least one alternate technology and method should be presented

and considered. A prototype implementation has been made (to validate practicality and determine speed), and is presented herein.

2. Objectives and Characteristics

This paper presents a proposal for authenticating VOEvent messages with PGP digital signatures. The proposed system has the following objectives and characteristics:

- **Separate:** no changes to the VOEvent XML schema, no canonicalization issues, etc.
- **Elective:** no impact on existing VOEvent message parsers
- **Flexible:** simple yet adaptable trust model, centralized hierarchy possible but not required
- **Simple:** minimal programming requirements (in contrast to XML Signature)
- **Fast Start** : keys and certificates can be generated in minutes, coding of sign/verify in days or less
- **Cross-platform:** can be implemented on virtually any operating system
- **Strong:** well known algorithms, sufficient strength for message value
- **Standards:** Internet standards based

3. PGP Trust in VOEvent

The PGP trust model lends itself well to the VOEvent environment. The following narrative assumes the reader understands the PGP “web of trust” model and digital signature concepts. PGP (like X.509) uses a *key-pair* system, in which the secret key is possessed only by the originator and the public key is freely distributed to recipients. In short, a message is signed using the originator's *secret* key and validated using the originator's *public key.* In order to prevent forgeries, the originator signs his public key as well, before distributing it. Recipients must be certain that they have a genuine copy of the originator's (signed) public key. Without that certainty ("trust"), they can't be certain that signatures were truly applied by the originator. In PGP, trust is added via countersignature.

Herein, the term *key* will refer to a signed public key (also known as a certificate). With PGP, there is no need for a hierarchy of trust with a master “root”. Instead, trust is distributed. In this proposal, given that the presence of a digital signature in a VOEvent message has no impact on receivers, a VOEvent message originator and a VOEvent message receiver pair can decide to use signed messages without action on anyone else’s part.

3.1 Initial Low Volume Scenario

The originator first provides his (signed public) key to the receiver such that the receiver is 100% certain that it is authentic[1]. The receiver then countersigns the originator’s key with *his* secret key, indicating that he trusts the key enough to use it for signature validation. At this point, both parties have everything they need to use signed VOEvent messages.

The originator starts sending his VOEvents with digital signatures. Again, this has no impact on other receivers. However, the receiver with whom he established a relationship can immediately begin to validate the incoming messages using the originator's key. As time goes on, other receivers may become interested in validating the signatures on the originator's messages. The originator can provide his key to anyone. The receiver just has to be 100% certain that it is authentic[1]. Anyone can validate the signatures on incoming messages. There is no need for involving a Certification Authority to whom the originator must prove his identity and pay money.

3.2 Scaling with Growth

While this simple scheme should suffice at least in the short term, it does not scale. If the number of interested receivers gets "too large" the burden on originators to prove that their key is genuine could become too great. In this case, PGP's "web of trust" feature is useful. Originators could collect signatures on their key by having others countersign them and return the key with the new signature. As the originator's key is passed around and signed, it eventually gets a "large" number of countersignatures on it.

At this point, other parties can decide that one or more of the parties that countersigned the originator's key are "trusted introducers", and that the presence of their countersignature(s) on the originator's key is/are enough to prove that it is genuine. This relieves the originator from having to provide first-hand proof of key authenticity to everyone.

3.3 Identity in Public Key

The public key contains its owner's identification. Normally this is constructed from a name, an email address, and a comment. In this application, it is suggested that the name be set to the author IVORN and the other fields left blank. Thus the key is identified by the IVORN of the originator as registered with IVOA. This permits the receiver to match up the identification of the key with the AuthorIVORN element (if present) in the VOEvent message itself, a further assurance of authenticity. See the example in Figure 3.

[1] Face to face contact is best, but the key can also be validated with some out-of-band check of the key fingerprint. For example, the originator's key could be downloaded from a web site, and then a phone call or email message could cross-check its fingerprint.

4. Format of Signed and Validated VOEvent Messages

In order to avoid impacting either the VOEvent schema or existing parsers, both the digital signature and validation results are enclosed in XML comments. Thus, parsers that do not support signed messages are unaware of the signature's presence. Including validation results in a validated received message provides a record of validation in event messages stored by the receiver.

The digital signature header is placed between the `xml` declaration and the root `VOEvent` tag, as shown in Figure 1. The hyphens in the signature header are

converted to equal signs, as XML comments cannot contain more than one hyphen in a row. These must be converted back to hyphens before validating by the receiver. The digital signature block is placed following the closing `/VOEvent` tag, as shown in Figure 2.

5. Prototype System

In the prototype system, PGP signing and validation services are provided by the open-source Gnu Privacy Guard (GnuPG) tool. This tool is freely available on virtually all OS platforms and it is actively maintained. Two versions are available. Version 2 (the latest) is much larger and more complex since it contains support for both OpenPGP and S.MIME (and thus X.509 PKI). Version 1.4.8 is sufficient for this application, and it is mature. Thus, GnuPG version 1.4.8 is recommended for this application, and was used in the demonstration system. A 1024 bit DSS (signing) key-pair was generated with a bogus IVORN as the key ID, and then used for signing messages. The Perl scripts, public/secret keys, and other bits used for the prototype are available at http://solo.dc3.com/~rdenny/VoDigiSig.zip. The passphrase for the secret key is "test".

Perl scripts are used to sign and validate VOEvent messages. Perl seems to be widely used for VOEvent processing and it lends itself to this sort of work, so it is a natural choice. However, virtually any language could have been used. Figure 4 shows the script for signing a message and Figure 5 shows the script for validating a signed message. The validation script leaves the validation results in the message for future reference as shown in Figure 3. It can be seen that the required code for signing and validating is minimal.

5.1 Validation of Integrity

In order to verify that no changes are made to the VOEvent message, including spurious line endings, the following pipeline was used to sign then validate a VOEvent message:

```
send.pl < orig.xml | recv.pl > rcvd.xml
```

The results of the test are shown as a diff listing for `orig.xml` versus `rcvd.xml` in Figure 6. It can be seen that there are no differences apart from the addition of the signature validation results in XML comments in `rcvd.xml`. Note the IVORN is displayed, as a result of using it as the signing key ID (as suggested in section 3.3).

5.2 Signing and Validating Rates

Scaling of this system is clearly affected by the time it takes to sign or validate a VOEvent message. For the demonstration system, the Perl benchmark package was used to test the rates of signing and validation. 100 cycles each of signing and validation were run and the timing results were encouraging. Output was redirected to the null device, so if it were to go to a real file, the time would increase. It took an average of 260 milliseconds to sign a message, and 190

milliseconds to validate. This was done on Windows XP running on a 2GHz Pentium CPU. Windows has a slower process activation time than most Unix-derivatives, so this was a worst-case test. In any case, this simple test shows that a modest system can sign or validate hundreds of thousands of message per day.

6. Conclusions and Discussion Points

It has been shown that there is a simple alternative to XML Signature (and related canonicalization issues) which provides the protection and authentication afforded by digitally signed VOEvent messages. However, the method presented diverges from the views of the IVOA and uses a cryptographic system that is considered by some to be somehow substandard. Is PGP (as described by RFC 4880 and implemented in GnuPG) truly substandard? If the use of PGP becomes a roadblock, should we consider some PKIX/X.509 method that still embeds the signature info in XML comments, outside of the VOEvent document? Most of the important advantages would remain.

The VOEvent Working Group has several major tasks ahead, and digital signatures is one of them. If XML Signature is used, digital signatures become cross-correlated with VOEvent 2.0 and canonicalization issues, creating one larger task out of two separable ones. Is XML Signature that important? Or is something akin to the system presented here "good enough"?

The science environment places a high value on breadth of thought and rigor. The engineering environment places a high value on finding solutions that are "good enough" and have the lowest risk/reward quotient. These two views during a design process can at times be orthogonal. But in the end, the design must be implemented. It is always better to have at least one complete working *production* implementation in hand *before* adopting a standard. In a sense, this is engineering rigor. Lessons are always learned when fielding systems based on a proposed standard. Often, the first prototype is (or should be!) thrown away. Design-and-decree standards have a history of high failure rates. Are we headed down that path with XML Signature? Will we end up with an ad-hoc "practical subset "? Is that even possible? Will we have done a large amount of work and made compromise decisions in order to accommodate XML Signature in VOEvent 2.0? Should we even set ourselves up for that?

Finally, can we afford to wait for (maybe) years before having the benefits of signed VOEvent messages? Or is it essential that we not make a misstep by adopting something rashly just to get going much sooner? If the reader takes anything away from this Note, the author hopes it will be the desire to find *technical* problems in this proposal and raise them on the VOEvent mailing list.

Appendix A: Figures

Figure 1: PGP digital signature header between xml declaration and document root, with hyphen to equal sign translation

```
<?xml version="1.0"  encoding="UTF-8"?>
<!--
=====BEGIN PGP SIGNED MESSAGE=====
Hash: SHA1

- --><VOEvent xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  ..
       <Who>
              <AuthorIVORN>ivo://bso.megaparsecs.abc/photons/</AuthorIV
              <Date>2008-02-18T16:58:27</Date>
       </Who>
       <What>
```

Figure 2: PGP digital signature block following the document root closing tag, with hyphen to equal sign translation

```
</VOEvent>
<!--
=====BEGIN PGP SIGNATURE=====
Version: GnuPG v1.4.8 (MingW32)

iEYEARECAAYFAkfTAB0ACgkQBJrAMSgAu9KH5ACeMFhuwynvp9GvaKfWAjbOsksv
zFsAoItRT0oBVzOG33gYUjHRrttvrovF
=gtsV
=====END PGP SIGNATURE=====
-->
```

Figure 3. Digital signature validation results between the xml declaration and the VOEvent document root, and with matching key ID and Author IVORN

```
<?xml version="1.0"  encoding="UTF-8"?>
<!--
Signature made 03/08/08 14:07:41 using DSA key ID  2800BBD2
Good signature from  "ivo://bso.megaparsecs.abc/photons/"
-->
<VOEvent  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" ...>
       <Citations/>
       <Who>
              <AuthorIVORN>ivo://bso.megaparsecs.abc/photons/</AuthorIV
              <Date>2008-02-18T16:58:27</Date>
       </Who>
       <What>
```

Figure 4. Perl script for applying the PGP digital signature (simplified error handling)

```
#!/usr/bin/perl
#
use strict;
while(<>) {
       if(substr($_,  '<?xml')) { last; }
        }
open TO, "| gpg --batch --passphrase-file  passph.txt --clearsign --no-
       or die "gpg failed: $!";
print TO '-->';
while (<>) { print TO }
print TO '<!--';
close TO
       or  die "pipe to gpg failed: $! ";
open FR, '~stmp~';
print "<?xml version=\"1.0\"  encoding=\"UTF-8\"?>\n<!--\n";
while (<FR>) {
       s/^-----([A-Z  ]+)-----/=====$1=====/g;
       print;
}
close FR;
print "-->\n";
unlink '~stmp~';
```

Figure 5. Perl script for validating and removing the PGP digital signature (simplified error handling)

```
#!/usr/bin/perl
#
use strict;


my $xml = '';
open TO, '>~vtmp~';
while (<>) {
       s/^=====([A-Z ]+)=====/-----$1-----/g;
       $xml .= $_;
       print TO;
}
close TO;
open VR, "gpg --verify ~vtmp~ 2>&1 |"
       or die "gpg failed: $!";
my $vr = '';
while(<VR>) {
       s/^gpg: //sg;
       $vr .= $_;
}
close VR
       or  die "pipe from gpg failed: $! ";
unlink '~vtmp~';
$xml =~ s/<!--.*- -->/<!--\n$vr-->\n/s;
```

```
$xml =~ s/Event>\n<!--.*-->/Event>/s;
print $xml;
```

Figure 6. Differences between the original and validated VOEvent messages

```
*** orig.xml  Wed  Mar 5 21:28:28 2008 UTC
--- rcvd.xml  Sat  Mar 8 21:09:03 2008 UTC
***************
*** 1,4 ****
  <?xml  version="1.0" encoding="UTF-8"?>
  <VOEvent  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" ..>
       <Citations/>
       <Who>
--- 1,8 ----
  <?xml  version="1.0" encoding="UTF-8"?>
+ <!--
+ Signature made 03/08/08 14:07:41 using DSA key  ID 2800BBD2
+ Good signature from  "ivo://bso.megaparsecs.abc/photons/"
+ -->
  <VOEvent  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" ..>
       <Citations/>
       <Who>
```

References

Alfarez, A-R, 1996, The PGP Trust Model (PDF)
http://netresearch.ics.uci.edu/Previous_research_projects/agentos/related/secu
-rahman-pgp-trust.pdf

Allen, S.L., 2007, VOEvent Authentication via XML digital signature, (informal, given at HTU Workshop)
http://www.cacr.caltech.edu/hotwired/program/presentations/xmldsig.pdf

Allen, S.L., 2008, VOEvent Authentication via XML digital signature, *Astron. Nach.*, 329, 298-300 (2008).
http://www3.interscience.wiley.com/cgi-bin/fulltext/117927641/PDFSTART

Ashley, J.M. et al., 1999, The Gnu Privacy Handbook (PDF)
http://gnupg.org/gph/en/manual.pdf

Callas, J. et al., 2007, OpenPGP Message Format (RFC 4880) (PDF)
http://tools.ietf.org/html/rfc4880

Eastlake, D. et al., XML Signature Syntax and Processing (RFC 3275)
http://www.ietf.org/rfc/rfc3275.txt

Howes, E., 2004, GnuPG Command Reference (unofficial, but excellent)
http://www.spywarewarrior.com/uiuc/gpg/gpg-com-0.htm

Koch, W. et al., Gnu Privacy Guard
http://gnupg.org/

Zimmerman, P., 2001, Why OpenPGP's PKI is better than an X.509 PKI (editorial)
http://www.openpgp.org/technical/whybetter.shtml