

# VOEvent Digital Signatures with PGP

Robert B. Denny

DC-3 Dreams SP, Mesa, Arizona

**Abstract:** The advantages of digitally signing VOEvent messages are well understood. This paper presents a proposal for accomplishing this with no impact on VOEvent message formats, minimal programming requirements, and no cost for certificates. This is an informal and unpublished engineering white paper. It is intended for electronic distribution as a PDF document, thus it contains embedded hyperlinks to references. Footnotes with the referenced hyperlink have been provided for users of printed copies.

**DRAFT 2 of March 8, 2008**

© 2008, Robert B. Denny, Mesa, AZ (unpublished)

## 1 Introduction

At the [Hotwiring the Transient Universe](#) workshop in July, 2008 (attended by the author) there was some discussion about security. S. L. Allen presented a paper *VOEvent Authentication via XML digital signatures*. 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 3<sup>rd</sup> party can forge a message from the originator.

Since the workshop, discussions have continued, focusing mainly on enveloped versus detached signatures, 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 the discussions, it has been assumed that [XML Digital Signature](#)<sup>1</sup> and the [X.509 \(PKIX\)](#)<sup>2</sup> certificate and trust model would be used. The author felt that, for completeness, at least one alternate approach should be presented and considered. A demonstration design and implementation have been made, and are presented herein.

## 2 Objectives

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

- No changes to the VOEvent XML schema
- No impact on existing VOEvent message parsers
- Simple trust model
- Minimal programming requirements
- Zero cost to originators and receivers of signed messages

## 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. 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. Given that the presence of a digital signature in a VOEvent message has no impact on receivers, a VOEvent message originator ("originator") and a VOEvent message receiver ("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<sup>3</sup>. The receiver then signs the originator's key with

---

<sup>1</sup> <http://www.w3.org/TR/xmlsig-core/>

<sup>2</sup> <http://www.ietf.org/html.charters/pkix-charter.html>

---

<sup>3</sup> 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.

his secret key. 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. Anyone can validate the signatures on incoming messages. There is no need for involving a third party (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 sign 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 signatures on it.

At this point, other parties can decide that one or more of the parties that signed the originator's key are "trusted introducers", and that the presence of their signature on the originator's key is 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.

## 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 Demonstration System

In the demonstration system, PGP signing and validation services are provided by the open-source [Gnu Privacy Guard](http://gnupg.org/)<sup>4</sup> (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 1.4.1 is sufficient for this application, and it is mature. Thus, GnuPG version 1.4.1 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.

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. 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 also affected by the time it takes to sign and 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

---

<sup>4</sup> <http://gnupg.org/>

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 References

- Alfarez, A-R, 1996, [The PGP Trust Model \(PDF\)](http://netresearch.ics.uci.edu/Previous_research_project/s/agentos/related/security/abdul-rahman-pgp-trust.pdf)  
[http://netresearch.ics.uci.edu/Previous\\_research\\_project/s/agentos/related/security/abdul-rahman-pgp-trust.pdf](http://netresearch.ics.uci.edu/Previous_research_project/s/agentos/related/security/abdul-rahman-pgp-trust.pdf)
- Allen, S.L., *Astron. Nach.*, **329**, 298-300 (2008).
- Ashley, J.M. et al., 1999, [The Gnu Privacy Handbook](http://gnupg.org/gph/en/manual.pdf)  
(PDF) <http://gnupg.org/gph/en/manual.pdf>
- Callas, J. et al., 2007, [OpenPGP Message Format \(RFC 4880\)](http://tools.ietf.org/html/rfc4880) (PDF) <http://tools.ietf.org/html/rfc4880>
- Howes, E., 2004, [GnuPG Command Reference](http://tools.ietf.org/html/rfc4880)  
<http://tools.ietf.org/html/rfc4880>
- Zimmerman, P., 2001, [Why OpenPGP's PKI is better than an X.509 PKI](http://www.openpgp.org/technical/whybetter.shtml) (editorial)  
<http://www.openpgp.org/technical/whybetter.shtml>

## VOEvent Digital Signatures with PGP (unpublished)

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

- --><VOEvent xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.0"
xmlns="http://www.ihoa.net/xml/VOEvent/v1.1" role="observation ... >
  <Who>
    <AuthorI VORN>i vo://gcn.gsfc/swi ft/</AuthorI VORN>
    <Date>2008-02-18T16:58:27</Date>
  </Who>
  <What>
```

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

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

i EYEARECAAYFAkftAB0ACgkQBjRAMSgAu9KH5ACeMFhuwynvp9GvaKfWAj b0sksv
zFsAoI tRT0oBVz0G33gYUj HRrttvrovF
=gtsV
=====END PGP SIGNATURE=====
-->
```

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

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
Signature made 03/08/08 14:07:41 using DSA key ID 2800BBD2
Good signature from "i vo://bso.megaparsecs.abc/photons/"
-->
<VOEvent xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.0"
xmlns="http://www.ihoa.net/xml/VOEvent/v1.1" role="observation" ... >
  <Citations/>
  <Who>
    <AuthorI VORN>i vo://gcn.gsfc/swi ft/</AuthorI VORN>
    <Date>2008-02-18T16:58:27</Date>
  </Who>
  <What>
```

Figure 3. Digital signature validation results between the xml declaration and the VOEvent document root

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

while(<>) {
  if(substr($_, '<?xml') ) { last; }
}
open TO, "| gpg --batch --passphrase-file passph.txt --clearsign --no-tty > ~stmp-";
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 4. Perl script for applying the PGP digital signature (simplified error handling)

## VOEvent Digital Signatures with PGP (unpublished)

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

while(<>) {
    if(substr($_, '<?xml') { last; }
}
open TO, "| gpg --batch --passphrase-file passph.txt --clearsign --no-tty > -stmp-"
    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)

```
*** 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>
```

Figure 6. Differences between the original and validated VOEvent messages

VOEvent Digital Signatures with PGP (unpublished)

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.8 (Mingw32)

mQGi BEFS9WERBACNj QFi hohZac6+sFrAKOnUzoKzs6UMUoI CI XV+nzwl 33j OJ2Ux
Ta2yI JzOr+mOPi ZwQZsg8AoBS2Lvz4zMDr87F96MM48vKOKhXyZST4UWUEaBUHZO
8qAMMF0zWwJ 1bStARXOj yh6k5vaD4L+s2gw7ei 2dNou2dntbm2BYSpsvgwCgk7gB
1MvwtKBFMUyYMHAI wVOMi pcD/j FJed0YdVYj uTI j nWcsa/09u3Lgl gOrybLnkboW
6mWscJecFN2hkZQCWFYI QsbgBr8VLUBvYXv2C2qTZwqxwBLmUVskuhMWHLv+QHMM
QTVcpcznKwTEKY2t1zsSi l fGxi CUUI 71DFuZL8auZbJaAeGrwEKf/ZvB8fZ5/Q8F
81K2A/4t+RGyi yn05TF5/MAi q4B5y+2GW9Kr6Tmj kE/MKXv8KCyL08y21Bbc/knl
Oyj ybvnFDEQtMzgHuZAU4Y0dj nGYNKbgyD0JF1U1kTVm6Z871F3n/pLQ51FTGyF
lpxrg+NmorRGP0Yzj nwXI 6FOVfVmtLnG6/48nTCG6hi pr0Z3qbQi aXZv0i 8vYnNv
Lm1l Z2FwYXJzZWNzLmFi Yy9waG90b25zL4hgBBMRAGAgBQJH0vVhAhsDBgsJCACD
AgQVAggDBBYCAwECHgECFAACGkQBjRmSgAu9Jz+ACfRhe7JZy0uURzF1 JAp9rH
Vd2aEHQAn2cAX4KqZGRkBVWAL9wcl x+ewl Mqi OEgBBABAgAKBQJH0v0i AwUBeAAK
CRBv7mXi i Geo52A8CACxp7d3H7Jfcaj R94sI goRhP6Q2Qcmp9mFei BKJ4Y5JNql
f1 fh2XrAJLGNsXNtmuP3di htZDKhqI wgf3+0Sbm83TF6Xa061Fm7cNeM8CymdZmd
HSquKXj R5NXQcj g+i Utm0/KW5k/9KX6gLbkNoUd1WI cbFC5h8Tmi Y264PA/LQfRm
2j AmVi qZk+M+L f7hRk3fAVZvmZ8LTD9f9PMGDo8yX5f9nCCdF7EvNxzxAnFRqf1/
RpV2K26MdMBAZm8UXl MqS2QGsFogw1WkphyLF4qVcNoWu00Mo0RU7cy0bH7y6JLk
fq74ohXLJ5pj Gj I 85/AwH2Zl wAWPQI heSzW5Q7qhuQI NBEFS9WEQCADE81ACoEi /
Nwpwi ZzJvFZKnJI GJl 1dBV6NI Yi GtCPu+JyJDbkFri mBNHwmMi PvWGGdXdI QHoXw
LAEwFJ4hDmcj DBI SW7V0pQkyXXcdM1LnFMkeLcDGRd0kk0o3vni JBUUC/f/bYg
7P8URaGf2dUvl YKPQkI uq5dEWj mdOHGb13kc/LSL04Ya698j zDbI j Xbo4RMg6b86
NVaXsGzFmMvaM52KwCEo6K1SdwsHyI YNBo0YTKqfn/KVMEDnUJdS1DrDQJl FVkb7
USMXWR+Qb0ptSW4A3y7nL/O8dkmLX45HzW2oS91pX+ZJAfS+Nzv/2DHki ONhYAF
KJtWfhl 3vJs/AAMFCACcyadLkCgbcnUaH+j j HDgXLUi 4l VvGV+BX+z4Z8cJ/2f9M
j 5+Uw5qy2x/Wkhmj ar8tRpp8Q0Vma9fU/sj 043HzbJ+qLb/koqwi 6l HmkPEfyUq0
qN5C+BQeSMrJVLUUGe97+z5yVuhN9mxh1j nAwnqvI A7EKv1nuR00Ng++Y4l GSroK
b2l kMkYH9mDq/ERQH7Qn7G3cwCgu8tGf+73/WGFEX4FK5bzkcnEnj HHwtHgPALYx
zn7edUTqEstaaC6RevoAyj 9OV0wozv23B1xY1RN0tqNvu7HTI LXF8l dKQEDJFr44
Rvgk+x/Wj bYThshxwCgHoRs5gCi See3yJXI l V6l Ei EkEGBECAAkFAkFS9WECGwwA
CgkQBjRmSgAu9KTrwCeLbbregygi eqj f4Gstht3r3nyS8Anj ODMsQ3PvuFI Nj 4
pmQG90l GxGSc
=/Hnh
-----END PGP PUBLIC KEY BLOCK-----
```

Figure 7. 1024 bit DSS key-pair used for signing in test (passphrase “test”)