Amazon SNS Message Signing Release Note

1/25/2011

Amazon SNS messages are signed using a cryptographic mechanism (private-public key pair) which can be used to verify that messages received from Amazon SNS have not been altered. Amazon SNS users can verify the authenticity of messages received from Amazon SNS by verifying the signature of the messages they receive against the Amazon SNS message signing X.509 certificate.

All Amazon SNS messages in machine readable JSON format will include a URL reference to the new Amazon SNS X.509 (public key infrastructure) certificate starting in January 2011. This URL reference to the Amazon SNS message signing certificate will automatically be updated to accommodate the annual expiration of X.509 certificates used for SNS message signing and validation. This ensures that the message signature encoding and subsequent verification steps for each message are always done using the same X.509 certificate and annual certificate rotation will not be impactful to applications that depend on message signature verification.

The current certificate can always be obtained from the following URL in each region:

US – N. Virginia: http://sns.us-east-1.amazonaws.com/SimpleNotificationService.pem

US – N. California: http://sns.us-west-1.amazonaws.com/SimpleNotificationService.pem

EU – Ireland: http://sns.eu-west-1.amazonaws.com/SimpleNotificationService.pem

APAC – Singapore: http://sns.ap-southeast-1.amazonaws.com/SimpleNotificationService.pem

All previous certificates will be retained with their explicit certificate ID and can be accessed at the following URL:

http://sns.[region].amazonaws.com/SimpleNotificationService-[certSerialNumberInHex].pem

A new property will be added to outgoing SNS notifications in JSON format which provides a pointer to the certificate that was used to sign the message. The Amazon SNS message verification certificate will exclusively be available from the domain 'amazonaws.com' as shown in the examples below and developers should consider restricting any verification logic to certificates that originate from amazonaws.com.

There will be no change to SNS notifications which are delivered in plain 'Email' format (human readable). Please note that additional message fields may be added to the Amazon SNS JSON message format in the future.

Old JSON message format example:

```
{
 "Type" : "Notification",
 "MessageId" : "d2a28afb-d617-40bc-8ce4-dfd99b179103",
 "TopicArn" : "arn:aws:sns:us-east-1:123456789012:babar1",
 "Message" : "foo",
 "Timestamp" : "2010-12-22T19:58:54.383Z",
 "SignatureVersion" : "1",
 "Signature" :
"PA0cpKrAicc43mify7VcUula0344oWo6lZu73H9c9pAs+ZcO+RGI5/0K5ur/hnG7g6r1OvgRDOlq
gtojApqVfRRR9HTl7gsANx8SPxdFgePVxAMb5ltzHOwEssv87RTMUOxttXSh+B1FVmU70RpP9SWBs
MKKBYP838oCLg2Ea2Y=",
```

```
 "UnsubscribeURL" : "https://sns.us-east-
1.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-east-1:
123456789012:babar1:a20c1582-19dd-4ef9-adfb-df858debcffc"
}
```

New JSON message format example:

```
{
 "Type" : "Notification",
 "MessageId" : "d2a28afb-d617-40bc-8ce4-dfd99b179103",
 "TopicArn" : "arn:aws:sns:us-east-1:123456789012:babar1",
 "Message" : "foo",
 "Timestamp" : "2010-12-22T19:58:54.383Z",
 "SignatureVersion" : "1",
 "Signature" :
"PA0cpKrAicc43mify7VcUula0344oWo6lZu73H9c9pAs+ZcO+RGI5/0K5ur/hnG7g6r1OvgRDOlq
gtojApqVfRRR9HTl7gsANx8SPxdFgePVxAMb5ltzHOwEssv87RTMUOxttXSh+B1FVmU70RpP9SWBs
MKKBYP838oCLg2Ea2Y=",
 "SigningCertURL" : "https://sns.us-east-
1.amazonaws.com/SimpleNotificationService-
6fe9ca67684fd989c58c4d2f7c7d0a90.pem",
 "UnsubscribeURL" : "https://sns.us-east-
1.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-east-1:
123456789012:babar1:a20c1582-19dd-4ef9-adfb-df858debcffc"
}
```

Details on Computing a Notification Message Signature

The first step in computing an Amazon SNS notification message signature is to extract the needed fields from the JSON formatted SNS message. It is recommended that you use a well-behaved JSON parser to extract the fields needed below. One reason it is important to use a JSON parser is that some control characters are escaped (such as newlines converted to '\n') when text is converted to JSON. In extracting the text the reverse conversion must be done.

The notification message signature is computed using the SHA1withRSA algorithm on a "canonical string" – a UTF-8 string which observes certain conventions including the sort order of included fields. (Please note that any deviation in the construction of the message string described below such as excluding a field, including an extra space or changing sort order will result in a different validation signature which will not match the pre-computed message signature.)

Using the JSON message fields contained in an Amazon SNS notification, the canonical string required for message validation can be constructed with the following function:

```
StringBuilder sb = new StringBuilder();
sb.append("Message\n");
sb.append(message).append("\n");
sb.append("MessageId\n");
sb.append(messageId).append("\n");
if (subject != null) {
sb.append("Subject\n");
sb.append(subject).append("\n");
}
sb.append("Timestamp\n");
sb.append(timestamp).append("\n");
sb.append("TopicArn\n");
sb.append(topicarn).append("\n");
sb.append("Type\n");
```

```
sb.append(type).append("\n");
return sb.toString().getBytes("UTF-8");
```

To verify the signature, first get the notifications service certificate (e.g. http://sns.us-east-1.amazonaws.com/SimpleNotificationService.pem). Extract the public key, decode the base64-encoded signature, and use a tool like openssl to verify the signature cryptographically. For example, given files CERT, SIG and MESS which contain the certificate, signature and the canonical string:

```
> openssl x509 -in CERT -pubkey -noout > pub
> base64 -i -d SIG > sigraw
> openssl dgst -sha1 -verify pub -signature sigraw MESS
```

Details on Computing a Subscribe or Unsubscribe Message Signature

The first step in computing an Amazon SNS subscribe or unsubscribe message signature is to extract the needed fields of the message from the JSON format. One should use a well-behaved JSON parser to extract the fields needed below. It is important to use a JSON parser as some control characters are escaped (such as newlines converted to '\n') during the conversion of text to JSON. In extracting the text the reverse conversion must be done.

The subscribe message signature is computed using the SHA1withRSA algorithm on a "canonical string" - a UTF-8 string constructed via the following function:

```
StringBuilder sb = new StringBuilder();
sb.append("Message\n");
sb.append(message).append("\n");
sb.append("MessageId\n");
sb.append(messageId).append("\n");
sb.append("SubscribeURL\n");
sb.append(subscribeurl).append("\n");
sb.append("Timestamp\n");
sb.append(timestamp).append("\n");
sb.append("Token\n");
sb.append(Token).append("\n");
sb.append("TopicArn\n");
sb.append(topicarn).append("\n");
sb.append("Type\n");
sb.append(type).append("\n");
return sb.toString().getBytes("UTF-8");
```