MAC Bypass in ESAPI Symmetric Encryption

Summary

Category:	Symmetric cryptography
Module:	ESAPI (Encryptor interface)
Announced:	2013-08-23 via ESAPI-Dev mailing list (http://lists.owasp.org/pipermail/esapi-dev/2013-August/002285.html)
Credits:	Philippe Arteau
Affects:	ESAPI 2.0GA, ESAPI 2.0.1
Corrected:	2013-09-02 in ESAPI 2.1.0 (cryptographic version 20130830)
Google Issue #:	306 (http://code.google.com/p/owasp-esapi-java/issues/detail?id=306)
CWE:	CWE-310 (Cryptographic Issues)
CVE Identifier:	CVE-2013-5679
CVSS Severity (version 2.0)	CVSS v2 Base Score: 5.6 Impact Subscore: 7.8 Exploitability Subscore: 3.9 CVSS Temporal Score: 4.4 CVSS Environmental Score: 4.4 Modified Impact Subscore: 7.8 Overall CVSS Score: 4.4 CVSS v2 Vector (AV:L/AC:L/Au:N/C:P/I:C/A:N/E:POC/RL:OF/RC:C/CDP:N/TD:ND/CR:M/IR:M/AR:L) **NOTE**: Because ESAPI is an API, the context of *how* it is used greatly affects the Exploitability metrics scoring component. You are encouraged to calculate a score that reflects you use of ESAPI encryption. The above CVSSv2 vector reflects what appears to be the most common use of ESAPI encryption which is to encrypt Java properties via EncryptedProperties.

Background

OWASP ESAPI (The OWASP Enterprise Security API) is a free, open source, web application security control library that makes it easier for programmers to write lower-risk

applications. The ESAPI for Java library is designed to make it easier for programmers to retrofit security into existing applications. ESAPI for Java also serves as a solid foundation for new development.

One of the sets of security controls provided by ESAPI for Java is wrappers around cryptographic primitives to provide authenticated encryption even when the underlying cryptographic primitives in the JCE provider that you are using does not supply these directly.

Problem Description

If you are using ESAPI symmetric crypto with CBC mode, PKCS#7 padding (called PKCS5Padding in Java), and an HMAC for authenticity, then it may be possible for an adversary *to bypass the authenticity check* by setting the MAC length to 0 and the MAC to null. The configuration could result in an exploitable vulnerability depending on the context for encryption and what degree an attacker can tamper with the serialized ciphertext. CBC mode with PKCS#7 padding and HMAC is the default configuration for ESAPI Java.

Only some ESAPI systems installations are exploitable when using the default configuration. To be exploitable, an attacker would require the ability to modify ciphertext either at rest or in transit. For example, consider the use of ESAPI encryption for encrypting Java properties via EncryptedProperties. Because properties files are generally not externally accessible, padding oracle attacks against encrypted property values would have to come from an insider. Additionally, the insider would require read access to the file containing the encrypted properties. On the other hand, use of ESAPI Encryptor to encrypt HTTP query parameters or HTTP cookies would be directly accessible to attackers (or at least authenticated attackers) thus increasing the risk of a successful attack. Because of these variations based on the context of how ESAPI encryption might be used, the accuracy of the CVSSv2 scores may not accurately reflect your use of ESAPI; you are encouraged to calculate your own scores based on your specific application’s usage using a CVSSv2 calculator.

Impact

Because a MAC bypass is possible, authenticity of the ciphertext cannot be guaranteed with the default ESAPI configuration. Consequently, this exposure may allow a successful padding oracle attack against the default ESAPI cryptosystem configuration and hence result in an exploitable vulnerability that could result in a loss of confidentiality or a bypass of the authentication or authorization system.

Workaround

Switch to an authenticated encryption cipher mode such as CCM or GCM, which are supplied by the SunJCE provider in Oracle JDK 7. For JDK 6 and earlier, you will need to use an alternate JCE provider such as Bouncy Castle.

Solution

Upgrade your ESAPI for Java library to version 2.1.0 or higher.

1. Download the relevant patch from the location below from:

   http://code.google.com/p/owasp-esapi-java/downloads/detail?name=esapi-2.1.0-dist.zip&can=2&q=#makechanges

2. Verify the SHA1 checksum is cfc2604798fbc11fdd0758ffcc931e324693a162. (Checksum listed at the bottom of the page of the above URL.)
3. Install the patch.
   a. Unzip the “esapi-2.1.0-dist.zip” file
   b. Drop the new ESAPI jar, esapi-2.1.0.jar, into your WEB-INF/lib directory or place it somewhere else in your CLASSPATH.

Correction Details

The details of the correction are in the ESAPI 2.1.0 release notes (http://owasp-esapi-java.googlecode.com/svn/trunk/documentation/esapi4java-core-2.1-release-notes.txt) and more details are in Google Issue #306 (http://code.google.com/p/owasp-esapi-java/issues/detail?can=1&q=306&colspec=ID%20Type%20Status%20Priority%20Milestone%20Component%20Owner%20Summary&id=306) including a JUnit test based on the Philippe Arteau’s original proof-of-concept code and discussion of how to tell if the security of your system may have been compromised as a result of this exposure.

Note that ESAPI release 2.1.0 only addresses Arteau’s first scenario, the MAC bypass, in his original disclosure. Depending on your configuration in ESAPI.properties, it may also be possible for an attacker to manipulate the cipher transformation (e.g., changing the cipher mode from CBC to OFB or padding scheme) to adverse effect. Note that attempts to change the cipher mode would not affect the default ESAPI configuration as the only supported non-authenticated cipher mode in the default ESAPI configuration is CBC. (See the ESAPI property “Encryptor.cipher_modes.additional_allowed” in ESAPI.properties for further details.) A future ESAPI 2.x release will address these scenarios. (Note: CVE identifier CVE-2013-5960 has been reserved for this issue. A separate security bulletin will be published to discuss it when a fix is available.) Note that switching to Google Tink (https://github.com/google/tink) for your encryption needs is highly recommended instead though. It is being actively developed and supported by Google, and it already explicitly supports important concepts like automatic key rotation and popular cloud HSMs for managing your encryption keys.

Additional Precautions

Once you have downloaded and installed the fixed version of the ESAPI jar, check if your system had been compromised.

To check if your system had been compromised from results of this exposure, locate your ESAPI logs and look for the following security event string (which will all be on a single line) in your ESAPI log file:

> Cannot validate MAC as it was never computed and stored. Decryption result may be garbage even when decryption succeeds.

Note that this log event can originate from two different causes so its presence is not 100% conclusive of a security breach. One possible cause is from a client using the deprecated Encryptor.decrypt(String) method (that was in ESAPI 2.0GA and 2.0.1 and carried over from ESAPI 1.4, now removed in ESAPI 2.1.0) and the method was used with the wrong encryption key. The other way that this security event could appear in the ESAPI logs is if the MAC length was set to 0 bytes and the MAC itself was set to null as demonstrated in the proof-of-concept code. It is advised that you look back through your ESAPI logs at least back to August 21, 2013 which is when Philippe Arteau first posted an announcement of this vulnerability to the public.

If you conclude that your system has been compromised via this issue, you should take the extra precaution of generating all new symmetric encryption keys that you have been using, including the Encryptor.MasterKey in your ESAPI.properties file if you have been using that. (Of course, if you plan to do so, if you have persisted any encrypted data using the old keys, you will first want to decrypt that data so it could be re-encrypted with the replacement keys just as if you were doing a key change operation.)

The reason that you ***might*** want to consider taking this additonal precaution is that this vulnerability could lead to a padding oracle attack against your system using the ESAPI 2.0 symmetric encryption—but only if you were using CBC mode and a MAC, which are the default configuration. As Duong and Rizzo showed us, it is possible to use a padding oracle attack to eventually divulge the secret encryption key.

References

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-5679

http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2013-5679

http://blog.h3xstream.com/2013/08/esapi-when-authenticated-encryption.html

Google Issue #: 306 (http://code.google.com/p/owasp-esapi-java/issues/detail?id=306)

Contact details: Kevin W. Wall <kevin.w.wall@gmail.com>