DELMIA Quintiq investigation into CVE-2021-44228 and CVE-2021-45046 Apache Log4j2 (revision 12)

Revision history

Revision	Date	Description
1	2021-12-13	Original Document
2	2021-12-14	Added information about PTV, added alternative self-patch instructions, added FAQ
3	2021-12-17	Updated plans for older version of Quintiq based on the release of log4j 2.12.2 and log4j 2.16 revised state and vulnerability
4	2021-12-20	Added details for Quintiq 5.4 and earlier on the self-patch strategy, extended the zero code change description. Updated FAQ.
5	2021-12-21	Added PTV patches will become available, explained 2.12.2 status
6	2021-12-22	Added released versions 5.4-6.2 to the table
7	2021-12-23	Minor updates.
8	2021-12-24	Marked final releases 5.2-5.3 as available.
9	2022-01-05	Updated information on QRServer vulnerability and minor description updates
10	2022-01-06	Revised self-patching instructions for earlier DELMIA Quintiq versions. Fixed the new CVE and fixed the order in the table of released versions
11	2022-01-24	Added more information on log4j 2.17.1 and 2.12.4 releases
12	2022-02-21	Added information on the vulnerability in Chainsaw, that was delivered as part of log4j 1.x

Background

On December 9 2021 a security vulnerability was identified in the Apache Log4j 2 libraries. This vulnerability (as described in https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-44228 can lead to remote execution if an attacker can influence the content of messages logged through the log4j library.

On December 14 2021 it was clear that the first attempt at fixing the issue by Apache left in another vulnerability (as described it https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-45046). On December 17 this vulnerability was upgraded from moderate to critical. This vulnerability will also be covered by this document.

In December 2021 A new version of log4j (2.17) became available. This addresses the following issues

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-45105 and

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-44832. The Quintiq software is not configured in such a way that we are vulnerable for this denial of service attack. We have therefore deliver hotfixes based on 2.16 or 2.12 as applicable and will update to 2.17 in our normal release cadence.

As a result of multiple requests from our customers we have revised the policy on 2.17. Instead of only releasing for the planned releases we will do another update to make sure that it is possible to have a 2.17.1 or 2.12.4 version of the Quintiq software. This will manage discussions with corporate IT departments that have a blanket policy of not allowing older versions of log4j.

In February 2022 a vulnerability in Chainsaw was detected. Chainsaw is distributed as part of the Log4J 1.x library. Chainsaw is an included tool that is delivered in the Log4J library and can be run from the jar file using the correct command line options. As the Quintiq log file format is not compatible with this tool, and we do not use this tool from our own dedicated logfile tooling, the risk of abuse is negligible. Also the Log4J 1.x library is only used in very old versions of Quintiq and we highly encourage customers who are nevertheless concerned about the inclusion of this tool to upgrade to a later version of Quintiq.

Quintiq Software Versions Affected

Quintiq 4.x – 5.1 → Uses log4j 1.x, which is not impacted directly

Quintiq 5.2 – 5.3 → Uses log4j version 2.2, which is impacted

Quintiq 5.4 – 5.6 → uses log4J version 2.6, which is impacted

Quintiq 6.0, Quintiq 2020, Quintiq 2021, Quintiq 2022 → uses log4J version 2.6, which is impacted

The affected parts of the Quintiq software are:

Server Side

QIntegrator – Integration tool consuming messages from external parties

QGateway – internet facing communication management tool

QJava Server – component used for BIRT reporting and server side rendering of gauges

Client Side

QThinClient – front end UI for all Quintiq users

Tooling

The tools are used during configuration and development. Affected tools are:

Configuration Utility – used to configure the Quintiq components

Log file viewer – used to view log files

Log File Analyzer – used to analyze log files

Log Performance Analyzer (versions 5.3 and earlier) – used to analyze model performance

Diagnostic Report Tool – used to collect logfiles for analysis

System Health Checker – used to send reports from LFA and DRT to sysadmins

BIRT Designer – used to create report templates

Others

PTV – 3rd party software used along with DELMIA Quintiq Software

Components not mentioned

Components not mentioned are not affected by the log4j issues.

Impact

The impact on the different components is dependent on the attack surface that is given by each of these components and therefore differs per component. Most affected are the server side components that have an open connection to the internet, being the QIntegrator and the QGateway.

Client side components and tooling are less vulnerable as they are not running open communication to the external world, they only communicate to other Quintiq services.

Mitigations

Summary

In order to prevent abuse of the vulnerability there are a number of opportunities. They need to be performed by system administrators of the DELMIA Quintiq systems. For DELMIA Hosted Solutions this is managed by DELMIA Quintiq.

- Zero code change mitigation: The vulnerability can be mitigated by firewall settings.
- Self-patching mitigation: The vulnerability can be removed by deleting a problematic class file.
- Hot fix from DELMIA Quintiq: Hotfixes will be released for the affected software versions.

Hot fix is the preferred mitigation plan for DELMIA Quintiq systems as customers and partners can conveniently address the vulnerability with a single installation. For customers or partners who does not prefer to proceed with a hot fix, you may consider the other mitigation plans. Note that each mitigation plans are equally capable of addressing this vulnerability.

Zero Code Change Mitigation

By using a firewall that blocks outgoing traffic for components that do not need to initiate outgoing traffic to the internet (e.g QGateway) it is possible to guard these processes fully. This is true for QGateway, QJavaServer, QRServer. The QIntegrator is more difficult to protect, but could also be restricted to the intended target hosts.

In order to use this approach a firewall, running on the local machine, or a firewall running on the DMZ where this machine is located should be configured to allow only outgoing connections to know host:port combinations. This means that the QGateway should be configured to only have an outgoing connection to the QTCE's that are configured for use, the QJavaServer and QRServer only to the QServer. For the QIntegrator this means that an analysis has to be done on all endpoints and whitelist all the configured endpoints. All other outgoing connections for the QIntegrator process should be blocked.

Self-Patching

The vulnerability can be removed by deleting a problematic class file. The objective is to remove all JndiLookup class file available in the bin\lib folder (inclusive of subfolders). Note that a full restart of DELMIA Quintiq components is necessary to apply the fix. The log files for some components may show a warning upon startup, but this does not affect the behavior of DELMIA Quintiq software except QSystemHealthChecker in certain version(s).

The sequence to run the self-patching mitigation plan is described below followed by the detailed instructions.

1. Run batch script provided in the detailed instructions to remove JndiLookup class file.
2. Perform additional patching instructions to specified log4j locations if you are using the DELMIA Quintiq earlier versions.

Note that the additional patching instructions are inclusive. Example if you are using DELMIA Quintiq version 5.3.0.0 you will also need to run the additional patching as mentioned for DELMIA Quintiq 5.4.2.3 and DELMIA Quintiq 5.4.0.0.

JndiLookup.class Removal Batch Script

This procedure is meant to be executed in all DELMIA Quintiq versions. The script will run recursively in your targeted root folder to remove JndiLookup.class file that matches the location path …\org\apache\logging\log4j\core\lookup\JndiLookup.class. As the procedure is based on 7Zip software, please make sure to install 7Zip (64 bit version) as a prerequisite.

1) Create a batch file with the content as shown in the text box below. Please ensure there is no unnecessary line breaks in the command during your batch file creation.

```
FOR /R %%f in (log4j-core-*.jar,QDiagnosticReport*.jar,QLogFileAnalyzer*.jar,QLogFileViewer*.jar,QSystemHealthChecker*.jar,
QLogPerformanceAnalyzer*.jar) DO (

echo %%f

start cmd.exe /c ""C:\Program Files\7-Zip\7z.exe" d "%%f" org\apache\logging\log4j\core\lookup\JndiLookup.class -r"

)
```

2) Run the batch file in all DELMIA Quintiq installation folders.

Note: You may refer to APPENDIX A for alternative self-patching instructions.

Additional Patching Instruction

Following the above batch script instructions you should have a patched log4j-core-*.jar. For the following versions there are additional folder locations containing log4j files. You will need to additionally replace the log4j-core-*.jar files based on your DELMIA Quintiq version.

<u>DELMIA Quintiq version 5.4.2.3 and earlier</u>

- QLogFileAnalyzer
- QLogFileViewer
- QDiagnosticReport
- com.quintiq.commons.logging
- QSystemHealthChecker
- BIRTDesigner\plugins\com.quintiq.commons.logging

<u>DELMIA Quintiq version 5.4.0.0 and earlier</u>

- QSystemHealthChecker\com.quintiq.commons.logging

<u>DELMIA Quintiq version 5.3.0.0 and earlier</u>

- QLogPerformanceAnalyzer

Below is a detailed steps to remove the additional log4j files present in earlier DELMIA Quintiq versions. Note that this must be executed after the existing self-patching instructions. The below steps are using com.quintiq.commons.logging as use case:

1. Go to. Quintiq installation lib folder and look for log4j jar file (…\Lib log4j-core-*.jar).
2. Perform the self-patching (i.e. via batch script or manually locating the JndiLookup.class file).
3. Create a copy of the patched log4j-core-*.jar file.

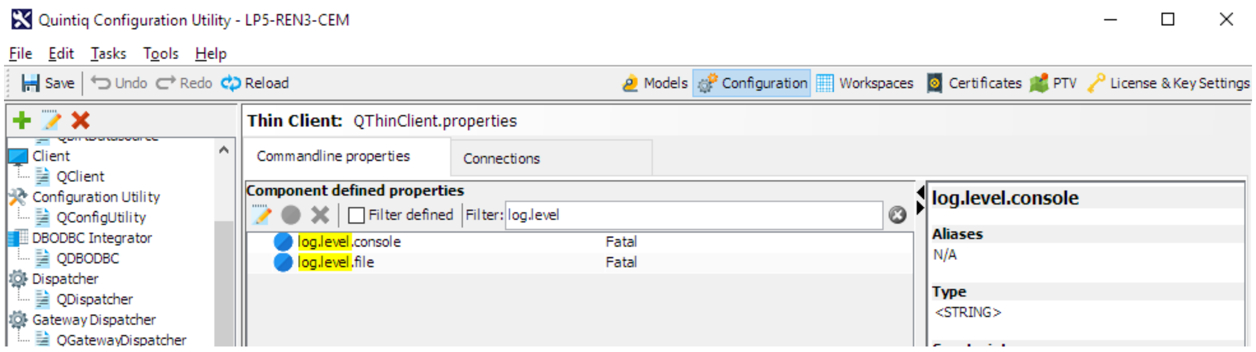
4. Go to com.quintiq.commons.logging-*.jar and open archive.
5. Replace the entire log4j-core-*.jar file with the patched version. Note that you may drag and drop the patched version inside the root folder (... \Lib\com.quintiq.commons.logging-*.jar\).

QThinClient running via JNLP/QJLauncher

Notice that the self-patching does not patch the QThinClient when it is run via JNLP or QJLauncher. Both launch protocols require the jar files to be signed, which can only be done by R&D. In order to use this procedure it is needed to use the .exe version of the Quintiq Thin Client.

Thin Client (TC) is a client process which does not open external management points. This means that from this perspective the attack surface is already extremely limited and basically limited to actions taken explicitly by the end user. Additional recommendation is to apply the zero code change on the customer machines which run QJLauncher/JNLP. If the TC process is only allowed to connect to hosting server via a firewall rule then the TC will not be able to connect to a malicious site.

In order to further mitigate the risks the TC may be configured to only log FATAL level by configuring this in the Configuration utility for log.level.console and log.level.file (see screenshot). In this way there no longer is a vulnerability in the TC. Note that for QJlauncher TC, you may need to redeploy in order to save your configurations. For JNLP TC, there should not be any additional redeployment required.



log.level.console	Fatal
log.level.file	Fatal

Patches from DS DELMIA Quintiq

Installing the official patch version from DS DELMIA Quintiq will ensure all DELMIA Quintiq software files are official again, making sure the client files that are signed by DELMIA Quintiq are also free of vulnerable versions. This will help for situations where a blanket scan is done to ensure that the network is safe.

Currently we have completed the patch delivery for all actively supported versions of Quintiq. Quintiq versions that are based on JAVA 8 or higher will be updated to log4j version 2.17.1 and for older versions of Quintiq we have created a Hotfix of these versions that uses log4j 2.12.4.

These patches will be based on the *latest* patch of the line that has been released until now, so all major minor versions will get a patch on their latest patch. For example, we will release a hotfix for version 6.0.10, but not for version 6.0.9.

Versions in active maintenance will be the first to receive this patch. Patches will be made available through the normal channels. Please refer to the FAQ for the patches availability.

PTV Patching

The PTV patched version is now available in our official download page. As for short-term mitigation plan, please refer to PTV blog for latest updates.

For any questions please don't hesitate to contact DELMIA Quintiq Support as per usual process (see https://www.3ds.com/support/contact/call-us/submit-a-request/).

FAQ

1. What is the timeline for the patches/hotfixes release from DS DELMIA Quintiq?
   R&D is working on a patch timeline and will communicate the plan ASAP. As of December 23 the following patches/hotfixes are available

Version	Log4j version	Availability
2021 (6.2) Refresh4	2.16.0	Available
2020 (6.1) Refresh9	2.16.0	Available
6.0.10	2.16.0	Available
5.6.2.5	2.16.0	Available
5.5.2.5	2.12.2	Available
5.4.2.6	2.12.2	Available
5.3.2.5	2.12.2	Available
5.2.2.5	2.12.2	Available

Note that on 3DS software download platform, a banner has been added if you have not selected the version with the log4j vulnerability patch. That is to ensure that customers download the latest patch containing the vulnerability correction.

The following future releases are planned to include 2.17.1/2.12.4

Version	Log4j version	Availability
2022 (6.3) Golden	2.17.0	Available
2021 (6.2) Refresh5	2.17.1	Available
2020 (6.1) Refresh 9	2.17.1	Available

6.0.10	2.17.1	Planned
5.6.2.5	2.17.1	Planned
5.5.2.5	2.12.4	Planned
5.4.2.6	2.12.4	Available
5.3.2.5	2.12.4	Planned
5.2.2.5	2.12.4	Available
5.3.2.5	2.12.4	Planned

2. I see that older versions of Quintiq are patched using log4j version 2.12. Why?
   These older versions of Quintiq are compatible with JAVA 7. Log4j version 2.12 is the official patched version from log4j for JAVA applications that are compatible with JAVA7.

3. What are the impact(s) of removing JndiLookup class file to our Quintiq software?
   There should not be any functional impact as JndiLookup should never be used during logging, the class is removed from the logging library. You may receive warning mentioning the missing files however this should not affect the component's functionality except QSystemHealthChecker in certain version(s). Note that for version 5.4.2.3 and earlier, there are additional self-patching instructions to ensure DELMIA Quintiq components can be started as usual.

4. In the zero code mitigation instruction, would blocking outbound traffic from the gateway affect any web socket traffic?
   No, it would not affect any web socket traffic.

5. Can we use both Self-Patching and Zero Code Change mitigation plan together?
   Yes.

6. Is self-patching sufficient to address the vulnerability?
   The self-patch strategy is sufficient to resolve the issue on all server side components. On the client side component self-patching can only be executed for installations that use the ThinClient.exe directly. For JNLP/QJLauncher based deployments the mechanisms that prevent deployment of unsigned code will prevent self-patching. Having said that, the QThinClient vulnerability is limited to (deliberate) end user actions and can be reduced further by limiting the log level as described in the latest revision of the knowledge base article. With this configuration, QThinClient will not be vulnerable anymore.

7. Do we need to take any additional steps after performing self-patching mitigation plans to ensure DELMIA Quintiq software is working?

Self-patching affects the JAVA based components of the DELMIA Quintiq suite as described in the knowledge base article. After self-patching, being able to start all services again is sufficient as all of these components will log their startup and therefore will clearly be able to execute.

8. Can we use the external tools that offers similar patching functionality to patch DELMIA Quintiq Software?
   For DELMIA Quintiq software, we highly encourage to use the self-patching instructions mentioned in this document as this has been tested and verified by our team.

9. Can we confirm that zero code mitigation plan is sufficient to be executed in local machine if I am also using VPN?
   Yes.

10. I do not see the QJLauncher marked as vulnerable. Is this correct?
    Yes this is correct the QJLauncher uses another logging mechanism and has no dependency on log4j. It is therefore not affected by the current log4j CVE's. The potential vulnerability lies on TC itself (inclusive TC running on QJLauncher or JNLP) which can be addressed with the above mitigation plans.

11. Is QMC affected by this log4j vulnerability?
    No.

12. How do we install the official patches from DELMIA Quintiq?
    Please treat the patches as any other patches and follow the existing upgrade procedure.

13. Is QRServer affected by the log4j vulnerability?
    Our team has further analyzed and found that RServer is using log4j v1 hence is not impacted by this vulnerability.

14. We noticed after executing the self-patching mitigation plan, the file owner under Windows properties is now updated to the user who made the change. Is there any impact to DELMIA Quintiq software?
    No there will not be any functionality impact. However only during uninstallation process, you may noticed some of the files might not be completely removed. You may need to do additional cleanup on the folder location if there is a need to uninstall Quintiq installation in the future.

15. We noticed QSystemHealthChecker is able to startup successfully after self-patching however we are not able to generate a report from it. What is the expectation and workaround for this?
    Unfortunately during our investigation and findings, we noticed that QSystemHealthChecker may not function properly in certain DELMIA Quintiq version(s). The long-term solution is to move towards the official log4j patch release by our DELMIA Quintiq R&D team. Alternatively, you may choose to download only the QSystemHealthChecker tool from 2020 Refresh9 HF2 official patch and reconfigure to use the QSystemHealthChecker to read logs and files from your existing version.

16. In Quintiq version 5.4, I noticed that there is a cache folder for QIntegrator (i.e. windows\temp folder) which also contains some vulnerable log4j file. Is it safe to remove the vulnerable files?
    Yes if you notice there are vulnerable files, it is safe to remove them.

17. What if I still have questions on this topic?
    Please contact DELMIA Quintiq Support so that we can assist you accordingly.

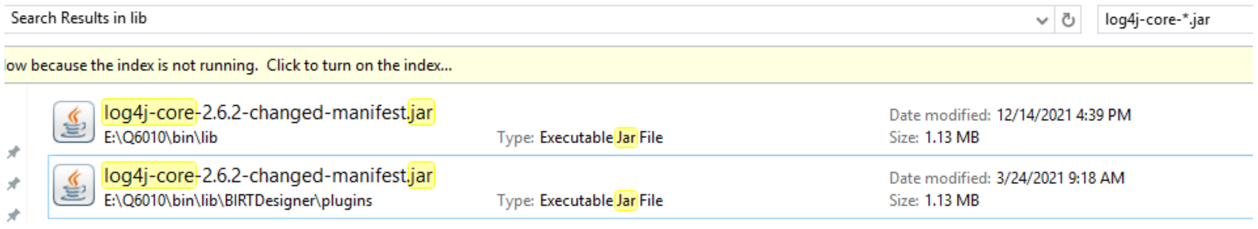
APPENDIX A

Alternative to Batch Script Self-Patching

The following instructions are meant for customer who prefers patching via the UI dialog. This alternate self-patching will have the same end results as batch script instructions. Note that for customer on DELMIA Quintiq version 5.4.2.3 and earlier, the **additional patching instructions is still mandatory** to be executed along with this UI self-patching. Please refer to the additional self-patching instructions for DELMIA Quintiq version 5.4.2.3 and earlier under Self-Patching.

Quintiq version 6.0 and higher

1. Navigate to DELMIA Quintiq Installation …\bin\lib subfolder.
2. In the Windows explorer search box, enter “log4j-core-*.jar” and click Enter.
3. It will return all the log4j-core related results as shown below:

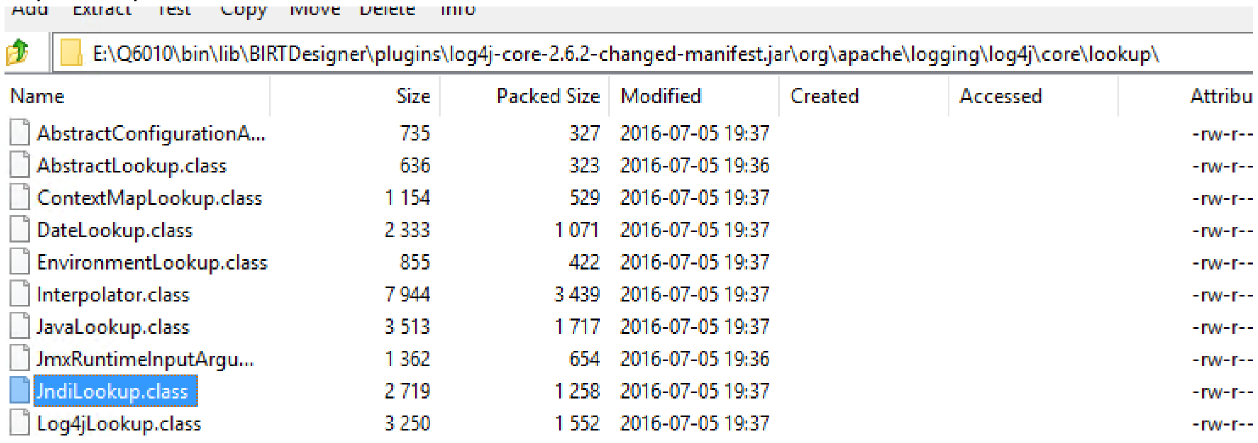


```
Search Results in lib                                              log4j-core-*.jar
iow because the index is not running.  Click to turn on the index...

log4j-core-2.6.2-changed-manifest.jar                                  Date modified: 12/14/2021 4:39 PM
E:\Q6010\bin\lib                       Type: Executable Jar File       Size: 1.13 MB

log4j-core-2.6.2-changed-manifest.jar                                  Date modified: 3/24/2021 9:18 AM
E:\Q6010\bin\lib\BIRTDesigner\plugins  Type: Executable Jar File       Size: 1.13 MB
```

4. Right click on the first file and select 7Zip > Open archive.
5. Navigate to “…\org\apache\logging\log4j\core\lookup\”
6. Look for “JndiLookup.class” and right click to delete the file.
7. Repeat step 4-6 for all other instances.



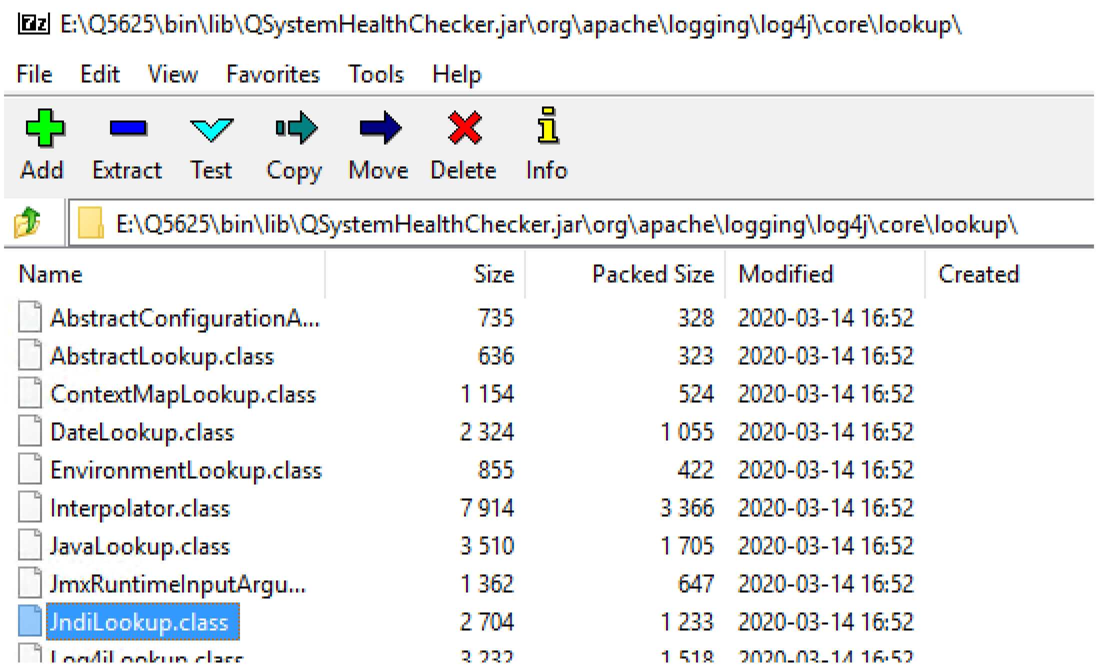
E:\Q6010\bin\lib\BIRTDesigner\plugins\log4j-core-2.6.2-changed-manifest.jar\org\apache\logging\log4j\core\lookup\

Name	Size	Packed Size	Modified	Created	Accessed	Attribu
AbstractConfigurationA...	735	327	2016-07-05 19:37			-rw-r--
AbstractLookup.class	636	323	2016-07-05 19:36			-rw-r--
ContextMapLookup.class	1 154	529	2016-07-05 19:37			-rw-r--
DateLookup.class	2 333	1 071	2016-07-05 19:37			-rw-r--
EnvironmentLookup.class	855	422	2016-07-05 19:37			-rw-r--
Interpolator.class	7 944	3 439	2016-07-05 19:37			-rw-r--
JavaLookup.class	3 513	1 717	2016-07-05 19:37			-rw-r--
JmxRuntimeInputArgu...	1 362	654	2016-07-05 19:36			-rw-r--
JndiLookup.class	2 719	1 258	2016-07-05 19:37			-rw-r--
Log4jLookup.class	3 250	1 552	2016-07-05 19:37			-rw-r--

Earlier Quintiq versions:

The same procedure above is applied but it has to be executed additionally for QLogFileAnalyzer.jar, QLogFileViewer.jar, QDiagnosticReport.jar, QSystemHealthChecker.jar, QLogPerformanceAnalyzer.jar and BirtDesigner folder. Note that lib folder path may differ in earlier DELMIA Quintiq versions.

Screenshot below is using QSystemHealthChecker.jar as an example


E:\Q5625\bin\lib\QSystemHealthChecker.jar\org\apache\logging\log4j\core\lookup\

File Edit View Favorites Tools Help
Add Extract Test Copy Move Delete Info

E:\Q5625\bin\lib\QSystemHealthChecker.jar\org\apache\logging\log4j\core\lookup\

Name	Size	Packed Size	Modified	Created
AbstractConfigurationA...	735	328	2020-03-14 16:52	
AbstractLookup.class	636	323	2020-03-14 16:52	
ContextMapLookup.class	1 154	524	2020-03-14 16:52	
DateLookup.class	2 324	1 055	2020-03-14 16:52	
EnvironmentLookup.class	855	422	2020-03-14 16:52	
Interpolator.class	7 914	3 366	2020-03-14 16:52	
JavaLookup.class	3 510	1 705	2020-03-14 16:52	
JmxRuntimeInputArgu...	1 362	647	2020-03-14 16:52	
JndiLookup.class	2 704	1 233	2020-03-14 16:52	
Log4jLookup.class	3 232	1 518	2020-03-14 16:52	