Oracle® Healthcare Data Repository

Secure Development Guide

Release 8.1.3

F52479-01

July 2022

Oracle Healthcare Data Repository Secure Development Guide, Release 8.1.3

F52479-01

Contents

Preface

Documentation accessibility iv
Diversity and Inclusion iv

1 Secure Development Guide

OWASP Top 10 Security Vulnerabilities 2017 1-1
- Injection 1-1
- Broken Authentication 1-1
- Sensitive Data Exposure 1-2
- XML External Entities (XXE) 1-2
- Broken Access Control 1-2
- Security Misconfiguration 1-2
- Cross Site Scripting (XXS) 1-3
- Insecure Deserialization 1-3
- Using Components with Known Vulnerabilities 1-3
- Insufficient Logging and Monitoring 1-3

Preface

This preface contains the following sections:

Documentation accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1

Secure Development Guide

The *Secure Development Guide* provides an overview of the security options for customers who will use Healthcare Data Repository (HDR) user database accounts and middle tier WebLogic user accounts to access the HDR APIs. Note that the set of recommendations in this document is not exhaustive and that no guarantee is given that implementing all the suggestions in this document provides sufficient protection for all security threats. The reason for this disclaimer is that you cannot delegate responsibility for secure application development to a third party or a single document. This document is to help developers be aware of the security tools and features that they can use to implement application security. This document does not replace a formal code review process.
Guidelines are presented here to assist in mitigating common security risks when customers are using the HDR APIs. The Open Web Application Security Project (OWASP) publishes the OWASP Top 10 to identify some of the most critical application security risks. This document briefly describes each Top 10 risk, provides the HDR mitigation strategies, and encourages our users to extend these strategies to secure their own applications and environments that use our APIs. Some of the web-specific Top 10 items don't apply to HDR; these are marked as **Not Applicable**.

OWASP Top 10 Security Vulnerabilities 2017

This paper discusses the practices and strategies used by the HDR application to mitigate risks posed by the security vulnerabilities documented in the OWASP Top 10 – 2017. Customers using the HDR APIs should be aware of and protect against these threats as well.

For the OWASP Foundation's description of the OWASP Top 10 Application Security Risks see the OWASP Top 10 - 2017 document (https://owasp.org/www-pdf-archive/OWASP_Top_10-2017_%28en%29.pdf.pdf).

Injection

SQL Injection can occur when untrusted data is used in a command or query. If an attacker sends hostile data, this can result in executing harmful commands or in unauthorized access of data.

The HDR APIs accept query parameters which could potentially contain hostile data. HDR code adheres to the recommended standards to avoid SQL Injection possibilities by using bind variables and by checking and validating the user input.

Customers using the HDR APIs to build applications should code carefully, using proper data validations and checking user input where needed.

Broken Authentication

The risk of insecure login credentials increases with custom built authentication schemes.

The HDR APIs are secured using the standard WebLogic authentication mechanism. Follow the WebLogic recommendations on creating and configuring a suitable authentication

provider where the HDR WebLogic user accounts will be stored. Refer to https://docs.oracle.com/middleware/12213/wls/SECMG/toc.htm.

Avoid building a custom authentication provider for using the HDR APIs. Use strong passwords and maintain security of account credentials.

Sensitive Data Exposure

Attackers may obtain unauthorized access to poorly protected sensitive data. Caution should be used to hide sensitive information from unauthorized users.

User authorization is not part of the HDR application itself and we strongly recommend that the applications developed using the HDR APIs have their own user management, authorization, and access control mechanisms to allow controlled access to data stored in HDR.

Also, transmission of data between HDR and client applications is always encrypted using TLS protocol.

XML External Entities (XXE)

Attackers can exploit vulnerable XML processors if they can upload XML or include hostile content in an XML document, exploiting vulnerable code, dependencies, or integrations.

In HDR, incoming XML is validated using XSD and a server-side data validation is implemented to prevent hostile (vulnerable) data coming from XML documents, headers, and nodes. HDR SOAP services use the SOAP 1.2 standard. HDR FHIR uses JSON data for data exchange.

The HDR build process is integrated with Fortify (static code analysis tool) to identify security vulnerabilities.

REST end points are tested using WebInspect tool to identify vulnerabilities within the rest API web layer.

Broken Access Control

Access control enforces policy such that users cannot act outside of their intended permissions.

APIs or resources in HDR are protected resources. Only valid users and those requests that contain proper access tokens and scopes can access protected resources. The application delegates the authorization decision to the WebLogic Server OPSS security framework. HDR also uses OAuth 2.0 authorization.

Security Misconfiguration

Attackers can take advantage when the security configuration is incorrect or incomplete and obtain unauthorized access to an application. The entire technology stack must be configured properly, and processes should be in place to detect misconfigurations including missing patches, accounts with default passwords, insecure settings in frameworks or libraries, and so forth.

The HDR, being an on-premise application, must be deployed in a secure WebLogic instance.

Ensure that your WebLogic configuration where HDR is deployed does not contain opportunities for an attacker to gain unauthorized access to the system. Take care to secure default accounts, files or directories, servers, the network, and other data access channels of the HDR deployment and APIs.

Cross Site Scripting (XXS)

Not applicable for access to HDR APIs.

Insecure Deserialization

Applications and APIs will be vulnerable if they deserialize hostile or tampered objects supplied by an attacker.

HDR does not accept serialized data from external client applications.

Using Components with Known Vulnerabilities

If an older version of a component with a known vulnerability is deployed in an environment, then an attacker who is aware of the vulnerability could take advantage and obtain unauthorized access.

HDR is built on a technology stack where patches and new releases offer improvements, including security-related modifications. The HDR application is an on-premise application and users should download and apply recommended security patches for the respective versions of components like WebLogic server and Oracle Database.

HDR users should follow the same policy of applying patches and updating to the latest versions of components being used, especially if security vulnerabilities have been reported in older versions.

Insufficient Logging and Monitoring

Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack business systems.

HDR API access is logged as part of an audit framework. The audit log contains key details such as user ID, resource name, timestamp, request URL, request IP address, and so forth.