hard.

***CVE-2015-0205 – Impact analysis on WR SSL and test approach***

**Prepared For RICOH**

**CONFIDENTIAL**

Version: 1.00

Date : May 8, 2015

Version: 1.00

Date: May 8, 2015

Wind River Systems, Inc.

9855 Scranton Road, Building 5

San Diego, CA 92121

858-824-3100 phone

858-824-3198 fax

[www.windriver.com](http://www.windriver.com)

Copyright

Unless otherwise agreed in writing, all copyright and intellectual property rights embodied in this document are and shall remain the property of Wind River Systems, Inc. This document is provided solely for the purposes of evaluating the work proposed and no other rights whatsoever to use the information herein are granted. The contents of this document may not be disclosed to any third party without the prior written consent of Wind River Systems, Inc.

Trademarks

Wind River, the Wind River logo, Tornado, and VxWorks are registered trademarks of Wind River Systems, Inc. Any third party trademarks referenced are the property of their respective owners. For further information regarding Wind River trademarks, please see <http://www.windriver.com/company/terms/trademark.html>.

| Revision History | | | |
| --- | --- | --- | --- |
| Date | Version | By | Description of Change |
| May 8, 2015 | 1.00 | Wind River | Official Release. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1 Overview 5](#_Toc418858928)

[1.1 Purpose and Scope 5](#_Toc418858929)

[1.2 Applicable Documents 5](#_Toc418858930)

[1.3 Glossary 5](#_Toc418858931)

[2 Impact analysis and Test Approach 6](#_Toc418858932)

[2.1 Vulnerability Summary 6](#_Toc418858933)

[2.2 Details of Vulnerability & Method of Exploit 6](#_Toc418858934)

[2.2.1 Background 6](#_Toc418858935)

[2.2.2 Method of Exploit 7](#_Toc418858936)

[2.3 Implementation Behavior – Openssl & Wind River SSL 7](#_Toc418858937)

[2.3.1 OpenSSL 1.0.1j 7](#_Toc418858938)

[2.3.2 Wind River – SSL 9](#_Toc418858939)

[2.4 Code changes & Impact of OpenSSL Fix 9](#_Toc418858940)

[2.5 Required Actions on WR-SSL 11](#_Toc418858941)

[2.5.1 Code Changes 11](#_Toc418858942)

[2.5.2 Test/verification 11](#_Toc418858943)

[2.6 Summary/Conclusion 11](#_Toc418858944)

[2.7 Attachments 11](#_Toc418858945)

# Overview

## Purpose and Scope

The Purpose of this Document is to explain details of impact analysis carried out on Wind River SSL stack used RICOH for security vulnerabilities as described in CVE-2015-0205.

Document also includes details of fix released by OpenSSL community, applicability and methods of test and verification of same under Wind River SSL.

## Applicable Documents

The following documents are referenced within:

| No. | Document | Version | Scope |
| --- | --- | --- | --- |
| 1. 1 | CVE-2015-0205\_Test\_Report1 | 0.01 | Test plan and different Scenarios  OpenSSL 1.0.1j client authentication with DH certificate |
| 1. 2 | CVE-2015-0205\_Test\_Report2 | 0.01 | Test plan and different Scenarios  OpenSSL 1.0.1k client authentication with DH certificate |
| 1. 3 | CVE-2015-0205\_Test\_Report3 | 0.01 | Test plan and different Scenarios  WR SSL without fix - client authentication with DH certificate |
| 1. 4 | CVE-2015-0205\_Test\_Report4 | 0.01 | Test plan and different Scenarios  WR SSL with fix - client authentication with RSA certificate |

## Glossary

|  |  |
| --- | --- |
| Term | Definition |
| WR | Wind River |
| SSL | Secure Sockets Layer |
| TLS | Transport Layer Security |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Impact analysis and Test Approach

## Vulnerability Summary

CVE-2015-0205 is a security vulnerability discovered in the OpenSSL implementation in Jan 2015 timeframe. Security threat uncovered is that an SSL server can potentially accept a DH certificate for client authentication even if client does not send a CertificateVerify message.

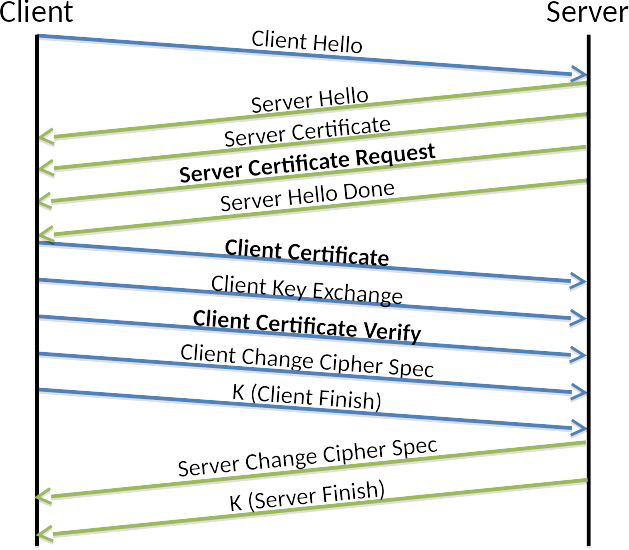
Details of the vulnerability are described in following section.

## Details of Vulnerability & Method of Exploit

### Background

Message exchanges between a client and server in an SSL handshake with mutual authentication (Client and server authenticate to each other) are as shown below.

[From <http://www.allanbank.com/blog/security/tls/x.509/2014/10/13/tls-x509-and-mongodb/> ]



It may be noted that Client certificate is sent only when server requests for one. Reason for server to request a certificate is to authenticate the client. i.e. to verify that client is really the party that is claiming as client.

It may also be noted that CertificateVerify message is not mandatory in all cases. In cases where client presents a certificate with signing capability, CertificateVerify must be sent after ClientKeyExchange, otherwise not.

Note: A certificate has signing capability if the private key corresponding to the public key in certificate can be used to digitally sign. While this is always the case with RSA certificates, it is NOT the case certificates containing fixed DH parameters.

A CertificateVerify contains a signed hash of all handshake messages up to this point. The hash is signed (encrypted) using client’s private key. Client’s public was sent in the certificate earlier to server. Server should be able to verify (decrypt) the signature using client’s public key (from certificate) and compare the decrypted hash value with its own computation of hash of all handshake messages and check if they match.

The mechanism of CertificateVerify allows server to verify that client indeed is in possession of the private key. In other words it assures the server that it is not talking to a client that has forged or stolen the certificate.

### Method of Exploit

A client starts SSL handshake and proceeds as explained in previous section until the Client certificate stage. At this time, client presents a certificate with no signing capability (i.e. Certificate with fixed DH values – referred to as DH certificate from here on). Then after that, Client chooses NOT to send CertificateVerify after ClientKeyExchange and instead sends finished message.

Even though it is expected that server does not proceed with handshake, versions OpenSSL 1.0.1j and many before, in fact proceed to a successful handshake.

Following section explain the behavior of various OpenSSL versions and WR-SSL version used by customer under the above scenario of attack.

## Implementation Behavior – Openssl & Wind River SSL

### OpenSSL 1.0.1j

OpenSSL Server uses state machine to implement the handshake part. This logic works following way.

*Note: state machine transitions and actions are simplified for easier understanding.*

Server initiates state machine and performs handshake message exchange upto ServerCertificate. After sending its certificate, server examines if there is a need to request for client certificate. If so, a CertificateRequest is prepared, sent and state is changed to SSL3\_ST\_SR\_CERT\_A (Server Receive CERTificate). A flag cert\_request is set to 0 or 1 to indicate if a certificate is expected from client side or not.

In SSL3\_ST\_SR\_CERT\_A state, if the flag is set to 1 (which will be the case for CVE-2015-0205), server invokes ssl3\_get\_client\_certificate() and then state is changed to SSL3\_ST\_SR\_KEY\_EXCH\_A (Server Receive KEY EXCHange) state.

In the SR\_KEYEXCH state, function ssl3\_get\_client\_key\_exchange is invoked to receive and process ClientKeyExchange packet. After that state is set to CERT\_VRFY\_A and function ssl3\_get\_cert\_verify() is called. At this point server also calculates hash values so that a comparison can be done after decrypting received CertificateVerify.

*Note: if the flag cert\_request is set to 0, function ssl3\_get\_client\_certificate () is not invoked but state is changed to SSL3\_ST\_SR\_KEY\_EXCH\_A and eventually to CERT\_VRFY\_A thus invoking ssl3\_get\_cert\_verify(). But in this case ssl3\_get\_cert\_verify() returns no error and state is changed to SR\_FINISHED.*

Relevant part of ssl3\_get\_cert\_verify function() in ssl/s3\_srvr.c, is explained below.

|  |  |  |
| --- | --- | --- |
| Line | Source code | Explanation |
| 2950 | type = type=X509\_certificate\_type(peer,publickey); | Type of X.509 certificate is stored in type variable. X509\_certificate\_type returns EVP\_PKT\_SIGN for sign capable certificates and EVP\_PKT\_EXCH for not sign-capable like DH certificate. |
|  | …. |  |
| 2958 | if (s->s3->tmp.message\_type != SSL3\_MT\_CERTIFICATE\_VERIFY) | Check if messagetype is CertificateVerify or not. Note that condition is true only if received message is NOT CertificateVerify. (i.e. received Finished instead of CertificateVerify as explained above) |
| 2959 | {  s->s3->tmp.reuse\_message=1; |  |
| 2961 | if ((peer != NULL) && (type | EVP\_PKT\_SIGN)) | Check if certificate received earlier from peer is valid and if the certificate is sign capable |
| 2962 | {  al=SSL\_AD\_UNEXPECTED\_MESSAGE;  SSLerr(SSL\_F\_SSL3\_GET\_CERT\_VERIFY, SSL\_R\_MISSING\_VERIFY\_MESSAGE);  goto f\_err;  } | **Control is reached here ONLY IF peer certificate is valid and sign capable. Handshake is aborted here.** |
| 2967 | ret=1;  goto end;  } | Handshake continues if control reaches here |

### Wind River – SSL

WR SSL is developed from OpenSSL 0.9.7e base. Implementation related to CertificateVerify processing of SSL server is same as OpenSSL 1.0.1j explained above.

However, WR-SSL version does not support DH certificates. This is same with OpenSSL 0.9.8.

## Code changes & Impact of OpenSSL Fix

Code change to fix the vulnerability is in ssl3\_get\_cert\_verify function() in ssl/s3\_srvr.c file. Following table explains the fix.

|  |  |  |
| --- | --- | --- |
| Line | Source code | Explanation |
| 2950 | type = type=X509\_certificate\_type(peer,publickey); | Type of X.509 certificate is stored in type variable. X509\_certificate\_type returns EVP\_PKT\_SIGN for sign capable certificates and EVP\_PKT\_EXCH for not sign-capable like DH certificate. |
|  | …. |  |
| 2958 | if (s->s3->tmp.message\_type != SSL3\_MT\_CERTIFICATE\_VERIFY) | Check if messagetype is CertificateVerify or not. Note that condition is true only if received message is NOT CertificateVerify. (i.e. received Finished instead of CertificateVerify as explained above) |
| 2959 | {  s->s3->tmp.reuse\_message=1; |  |
| 2961 | if ~~(~~(peer != NULL) ~~&& (type | EVP\_PKT\_SIGN))~~ | Check if certificate received earlier from peer is valid |
| 2962 | {  al=SSL\_AD\_UNEXPECTED\_MESSAGE;  SSLerr(SSL\_F\_SSL3\_GET\_CERT\_VERIFY, SSL\_R\_MISSING\_VERIFY\_MESSAGE);  goto f\_err;  } | **Control is reached here if peer certificate (any) is valid and handshake is aborted here.** |
| 2967 | ret=1;  goto end;  } | Handshake continues if control reaches here |

Following is the impact of code change under different scenarios

|  |  |  |
| --- | --- | --- |
| No | Scenario | Applicability |
| 1 | Client sends Sign capable CERT and send CertificateVerify | Fix not applicable |
| 2 | Client sends Sign capable CERT and not send CertificateVerify | **Applicable. Handshake fails if cert type is sign capable before fix. Same behavior after fix too.**  *NOTE: There is a possibility that client does not send CertificateVerify because it uses its certificate private key to encrypt pre-master-secret. However in such case OpenSSL server implementation does not invoke ssl3\_get\_cert\_verify() function at all and handshake will succeed). [the function ssl3\_get\_client\_key\_exchange returns 2]* |
| 3 | Client sends NON Sign capable CERT and send CertificateVerify | Fix Not applicable |
| 4 | Client sends NON Sign capable CERT and does NOT send CertificateVerify | **Applicable. Handshake success before fix, handshake fail after fix** |
| 5 | Client does NOT send CERT but send CertificateVerify | Fix Not applicable |
| 6 | Client does not send CERT and no CertificateVerify | Fix Not applicable |

*Note: meaning of applicable or not applicable is that modified part of source code is executed or not in that particular scenario.*

## Required Actions on WR-SSL

### Code Changes

Code changes are applicable to Wind River SSL version. However since WR SSL does not support DH Certificate, vulnerability is NOT APPLICABLE. WR SSL can become vulnerable if DH certificate support gets added in future.

### Test/verification

Since vulnerability is not applicable to WR SSL, following is the test/verification approach.

1. Create test method and files, verify that test files are correct by testing them with 1.0.1j and 1.0.1k

Report 1 – 1.0.1j client authentication with DH certificate

Report 2 – 1.0.1k client authentication with DH certificate

1. Apply the same test bed, certificates and files to WR SSL. Verify that DH certificates are not accepted and confirm that WR SSL is not vulnerable.

Report 3 – WR SSL without fix

1. Apply code changes to WR-SSL. To ensure that code change does not affect effect certificates with signing capability, test with RSA certificates and ensure it is working fine.

Report 4 – WR SSL with fix

## Summary/Conclusion

It has been found that current version of WR-SSL used by customer is not vulnerable to the threat described in CVE-2015-0205. However code changes are still applicable. Those changes are made and tests are performed to confirm that vulnerability is not applicable and certificate related functionalities in the current version are not affected.

## Attachments

1. Report 1 – OpenSSL 1.0.1j client authentication with DH certificate
2. Report 2 – OpenSSL 1.0.1k client authentication with DH certificate
3. Report 3 – WR SSL without fix - client authentication with DH certificate
4. Report 4 – WR SSL with fix - client authentication with RSA certificate