Assurance Activity Report for Nubo Client Version 3.2
Version 1.10, 18 December 2023

Nubo Client Version 3.2 Security Target
Version 1.18, 15 December 2023

Nubo Client Version 3.2 Guidance Document
15 December 2023

***Protection Profile for Application Software*, Version 1.4**
***Functional Package for Transport Layer Security (TLS)*, Version 1.1**

Evaluated by:

intertek
acumen
security

2400 Research Blvd, Suite 395
Rockville, MD 20850

Prepared for:

NIAP

National Information Assurance Partnership
Common Criteria Evaluation and Validation Scheme

The Developer of the TOE:
Nubo Software LTD.

The Author of the Security Target:
Nubo Software LTD.

The TOE Evaluation was Sponsored by:
Nubo Software LTD.

Evaluation Personnel:
George Kumi
Joan Marshall
Shaina Rae

Common Criteria Version
Common Criteria Version 3.1 Revision 5

Common Evaluation Methodology Version
CEM Version 3.1 Revision 5

Revision History

VERSION	DATE	CHANGES
1.0	06/21/2023	Initial Release
1.0a	7/21/2023	Testing Updates
1.1	7/27/2023	TSS Related Updates
1.2	8/03/2023	SAR updates and Combined Test Cases
1.3	8/15/2023	Updates
1.4	08/29/2023	Updates
1.5	10/02/2023	Added TD0779 and archived TD0588
1.6	10/20/2023	Reviewed
1.7	08/12/2023	Updated based on ECR comments
1.8	12/08/2023	Updated based on ECR comments
1.9	12/15/2023	Updated based on ECR comments.
1.10	12/15/2023	Updated based on ECR comments.

Contents

1 TOE Overview 10
2 Assurance Activities Identification 11
3 Test Equivalency Justification 12
4 Test Bed Descriptions 13
4.1 Test Bed 13
4.2 Configuration Information 13
4.3 Test Time and Location 14
5 Detailed Test Cases (TSS and AGD Activities) 15
5.1 Mandatory Requirements 15
5.1.1 Cryptographic Support (FCS) 15
5.1.1.1 FCS_CKM_EXT.1 Cryptographic Key Generation Services (Applied TD0717) 15
5.1.1.1.1 FCS_CKM_EXT.1.1 TSS 15
5.1.1.1.2 FCS_CKM_EXT.1.1 Guidance 15
5.1.1.2 FCS_RBG_EXT.1 Random Bit Generation Services 15
5.1.1.2.1 FCS_RBG_EXT.1.1 TSS 15
5.1.1.2.2 FCS_RBG_EXT.1.1 Guidance 16
5.1.1.3 FCS_STO_EXT.1 Storage of Credentials 16
5.1.1.3.1 FCS_STO_EXT.1 TSS 16
5.1.1.3.2 FCS_STO_EXT.1.1 Guidance 16
5.1.1.4 FCS_TLS_EXT.1 TLS Protocol 16
5.1.1.4.1 FCS_TLS_EXT.1.1 TSS 16
5.1.1.4.2 FCS_TLS_EXT.1.1 Guidance 17
5.1.2 User Data Protection (FDP) 17
5.1.2.1 FDP_DAR_EXT.1 Encryption Of Sensitive Application Data 17
5.1.2.1.1 FDP_DAR_EXT.1.1 TSS 17
5.1.2.1.2 FDP_DAR_EXT.1.1 Guidance 17
5.1.2.2 FDP_DEC_EXT.1 Access to Platform Resources 17
5.1.2.2.1 FDP_DEC_EXT.1.1 TSS 18
5.1.2.2.2 FDP_DEC_EXT.1.1 Guidance 18
5.1.2.2.3 FDP_DEC_EXT.1.2 TSS 18
5.1.2.2.4 FDP_DEC_EXT.1.2 Guidance 18
5.1.2.3 FDP_NET_EXT.1 Network Communications 19
5.1.2.3.1 FDP_NET_EXT.1.1 TSS 19
5.1.2.3.2 FDP_NET_EXT.1.1 Guidance 19
5.1.3 Security Management (FMT) 19
5.1.3.1 FMT_CFG_EXT.1 Secure by Default Configuration 19
5.1.3.1.1 FMT_CFG_EXT.1.1 TSS 19
5.1.3.1.2 FMT_CFG_EXT.1.1 Guidance 19
5.1.3.1.3 FMT_CFG_EXT.1.2 TSS 19
5.1.3.1.4 FMT_CFG_EXT.1.2 Guidance 19
5.1.3.2 FMT_MEC_EXT.1 Supported Configuration Mechanism 19
5.1.3.2.1 FMT_MEC_EXT.1.1 TSS 19
5.1.3.2.2 FMT_MEC_EXT.1.1 Guidance 20
5.1.3.3 FMT_SMF.1 Specification of Management Functions 20
5.1.3.3.1 FMT_SMF.1.1 TSS 20
5.1.3.3.2 FMT_SMF.1.1 Guidance 20

5.1.4 Privacy (FPR) 21
5.1.4.1 FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information 21
5.1.4.1.1 FPR_ANO_EXT.1.1 TSS 21
5.1.4.1.2 FPR_ANO_EXT.1.1 Guidance 21
5.1.5 Protection of the TSF (FPT) 21
5.1.5.1 FPT_AEX_EXT.1 Anti-Exploitation Capabilities 21
5.1.5.1.1 FPT_AEX_EXT.1.1 TSS (TD0798) 21
5.1.5.1.2 FPT_AEX_EXT.1.1 Guidance 21
5.1.5.1.3 FPT_AEX_EXT.1.2 TSS 22
5.1.5.1.4 FPT_AEX_EXT.1.2 Guidance 22
5.1.5.1.5 FPT_AEX_EXT.1.3 TSS 22
5.1.5.1.6 FPT_AEX_EXT.1.3 Guidance 22
5.1.5.1.7 FPT_AEX_EXT.1.4 TSS 22
5.1.5.1.8 FPT_AEX_EXT.1.4 Guidance 22
5.1.5.1.9 FPT_AEX_EXT.1.5 TSS 22
5.1.5.1.10 FPT_AEX_EXT.1.5 Guidance 22
5.1.5.2 FPT_API_EXT.1 Use of Supported Services and APIs 22
5.1.5.2.1 FPT_API_EXT.1.1 TSS 22
5.1.5.2.2 FPT_API_EXT.1.1 Guidance 23
5.1.5.3 FPT_IDV_EXT.1 Software Identification and Versions 23
5.1.5.3.1 FPT_IDV_EXT.1.1 TSS 23
5.1.5.3.2 FPT_IDV_EXT.1.1 Guidance 24
5.1.5.4 FPT_LIB_EXT.1 Use of Third Party Libraries 24
5.1.5.4.1 FPT_LIB_EXT.1.1 TSS 24
5.1.5.4.2 FPT_LIB_EXT.1.1 Guidance 24
5.1.5.5 FPT_TUD_EXT.1 Integrity for Installation and Update 24
5.1.5.5.1 FPT_TUD_EXT.1.1 TSS 24
5.1.5.5.2 FPT_TUD_EXT.1.1 Guidance 24
5.1.5.5.3 FPT_TUD_EXT.1.2 TSS 24
5.1.5.5.4 FPT_TUD_EXT.1.2 Guidance 24
5.1.5.5.5 FPT_TUD_EXT.1.3 TSS 25
5.1.5.5.6 FPT_TUD_EXT.1.3 Guidance 25
5.1.5.5.7 FPT_TUD_EXT.1.4 TSS 25
5.1.5.5.8 FPT_TUD_EXT.1.4 Guidance 25
5.1.5.5.9 FPT_TUD_EXT.1.5 TSS 25
5.1.5.5.10 FPT_TUD_EXT.1.5 Guidance 26
5.1.6 Trusted Path/Channels (FTP) 26
5.1.6.1 FTP_DIT_EXT.1 Protection of Data in Transit 26
5.1.6.1.1 FTP_DIT_EXT.1.1 TSS 26
5.1.6.1.2 FTP_DIT_EXT.1.1 Guidance 26
5.2 Optional Requirements 26
5.3 Selection-Based Requirements 26
5.3.1 Cryptographic Support (FCS) 26
5.3.1.1 FCS_CKM.1/AK Cryptographic Asymmetric Key Generation (Applied TD0717) 26
5.3.1.1.1 FCS_CKM.1.1/AK TSS 27
5.3.1.1.2 FCS_CKM.1.1/AK Guidance 27
5.3.1.2 FCS_CKM.2 Cryptographic Key Establishment (TD0717) 27
5.3.1.2.1 FCS_CKM.2.1 TSS 27
5.3.1.2.2 FCS_CKM.2.1 Guidance 28

5.3.1.3 FCS_COP.1/Hash Cryptographic Operation – Hashing (Applied TD0717) 28
5.3.1.3.1 FCS_COP.1.1/Hash TSS 28
5.3.1.3.2 FCS_COP.1.1/Hash Guidance 29
5.3.1.4 FCS_COP.1/KeyedHash Cryptographic Operation - Keyed-Hash Message Authentication (Applied TD0717) 29
5.3.1.4.1 FCS_COP.1.1/KeyedHash TSS 29
5.3.1.4.2 FCS_COP.1.1/KeyedHash Guidance 29
5.3.1.5 FCS_COP.1/Sig Cryptographic Operation – Signing (Applied TD0717) 29
5.3.1.5.1 FCS_COP.1.1/Sig TSS 29
5.3.1.5.2 FCS_COP.1.1/Sig Guidance 29
5.3.1.6 FCS_COP.1/SKC Cryptographic Operation - Encryption/Decryption (Applied TD0717) 29
5.3.1.6.1 FCS_COP.1.1/SKC TSS 29
5.3.1.6.2 FCS_COP.1.1/SKC Guidance 29
5.3.1.7 FCS_HTTPS_EXT.1/Client HTTPS Protocol 30
5.3.1.7.1 FCS_HTTPS_EXT.1.1/Client TSS 30
5.3.1.7.2 FCS_HTTPS_EXT.1.1/Client Guidance 30
5.3.1.7.3 FCS_HTTPS_EXT.1.2/Client TSS 30
5.3.1.7.4 FCS_HTTPS_EXT.1.2/Client Guidance 30
5.3.1.7.5 FCS_HTTPS_EXT.1.3/Client TSS 30
5.3.1.7.6 FCS_HTTPS_EXT.1.3/Client Guidance 30
5.3.1.8 FCS_RBG_EXT.2 Random Bit Generation from Application 30
5.3.1.8.1 FCS_RBG_EXT.2.1 TSS 30
5.3.1.8.2 FCS_RBG_EXT.2.1 Guidance 30
5.3.1.8.3 FCS_RBG_EXT.2.2 TSS 31
5.3.1.8.4 FCS_RBG_EXT.2.2 Guidance 31
5.3.1.9 FCS_TLSC_EXT.1 TLS Client Protocol (Applied TD0442) 31
5.3.1.9.1 FCS_TLSC_EXT.1.1 TSS 31
5.3.1.9.2 FCS_TLSC_EXT.1.1 Guidance 31
5.3.1.9.3 FCS_TLSC_EXT.1.2 TSS 32
5.3.1.9.4 FCS_TLSC_EXT.1.2 Guidance 32
5.3.1.9.5 FCS_TLSC_EXT.1.3 TSS 32
5.3.1.9.6 FCS_TLSC_EXT.1.3 Guidance 33
5.3.1.10 FCS_TLSC_EXT.5 Client Support for Supported Groups Extension 33
5.3.1.10.1 FCS_TLSC_EXT.5.1 TSS 33
5.3.1.10.2 FCS_TLSC_EXT.5.1 Guidance 33
5.3.2 Identification and Authentication (FIA) 33
5.3.2.1 FIA_X509_EXT.1 X.509 Certificate Validation 33
5.3.2.1.1 FIA_X509_EXT.1.1 TSS 33
5.3.2.1.2 FIA_X509_EXT.1.1 Guidance 34
5.3.2.1.3 FIA_X509_EXT.1.2 TSS 34
5.3.2.1.4 FIA_X509_EXT.1.2 Guidance 34
5.3.2.2 FIA_X509_EXT.2 X.509 Certificate Authentication 34
5.3.2.2.1 FIA_X509_EXT.2 TSS 34
5.3.2.2.2 FIA_X509_EXT.2 Guidance 35
5.3.3 Protection of the TSF (FPT) 35
5.3.3.1 FPT_TUD_EXT.2 Integrity for Installation and Update (Applied TD0628) 35
5.3.3.1.1 FPT_TUD_EXT.2.1 TSS 35
5.3.3.1.2 FPT_TUD_EXT.2.1 Guidance 35
5.3.3.1.3 FPT_TUD_EXT.2.2 TSS 35
5.3.3.1.4 FPT_TUD_EXT.2.2 Guidance 35
5.3.3.1.5 FPT_TUD_EXT.2.3 TSS 35

5.3.3.1.6 FPT_TUD_EXT.2.3 Guidance 36

6 Security Assurance Requirements 37

6.1 Security Target (ASE) 37

6.2 Development (ADV) 37

6.2.1 ADV_FSP.1 Basic Functional Specification 37

6.2.1.1 ADV_FSP.1.1E 37

6.2.1.2 ADV_FSP.1.2E 37

6.3 Guidance Documentation (AGD) 37

6.3.1 AGD_OPE.1 Operational User Guidance 37

6.3.1.1 AGD_OPE.1.1E 37

6.3.2 AGD_PRE.1 Preparative Procedures 38

6.3.2.1 AGD_PRE.1.1E and AGD_PRE.1.2E 38

6.4 Life-cycle Support (ALC) 39

6.4.1 ALC_CMC.1 Labeling of the TOE 39

6.4.1.1 ALC_CMC.1.1E 39

6.4.2 ALC_CMS.1 TOE CM Coverage 40

6.4.2.1 ALC_CMS.1.1E 40

6.4.3 ALC_TSU_EXT.1 Timely Security Updates 40

6.4.3.1 ALC_TSU_EXT.1.1E 40

6.5 Tests (ATE) 41

6.5.1 ATE_IND.1 Independent Testing 41

6.5.1.1 ATE_IND.1.1E and ATE_IND.1.2E 41

6.6 Vulnerability Assessment (AVA) 43

6.6.1 AVA_VAN.1 Vulnerability Survey 43

6.6.1.1 AVA_VAN.1.1E, AVA_VAN.1.2E, and AVA_VAN.1.3E 43

7 Detailed Test Cases (Test Activities) 45

7.1 APP_V1.4 45

7.1.1 FCS_CKM.1/AK Test/CAVP 1 45

7.1.2 FCS_CKM.2 Test/CAVP 1 47

7.1.3 FCS_COP.1/Hash Test/CAVP 1 50

7.1.4 FCS_COP.1/KeyedHash Test/CAVP 1 51

7.1.5 FCS_COP.1/Sig Test/CAVP 1 52

7.1.6 FCS_COP.1/SKC Test/CAVP 1 53

7.1.7 FCS_RBG_EXT.2.1 Test/CAVP 1 58

7.1.8 FCS_RBG_EXT.2.2/Client Test #1 59

7.1.9 FCS_HTTPS_EXT.1.2/Client Test #1 60

7.1.10 FCS_HTTPS_EXT.1.3/Client Test #1 60

7.1.11 FCS_STO_EXT.1.1 Test #2 60

7.1.12 FDP_DAR_EXT.1.1 Test #1 61

7.1.13 FDP_DAR_EXT.1.1 Test #2 61

7.1.14 FDP_DEC_EXT.1.1 Test #1 61

7.1.15 FDP_DEC_EXT.1.2 Test #1 62

7.1.16 FDP_NET_EXT.1.1 Test #1 62

7.1.17 FDP_NET_EXT.1.1 Test #2 62

7.1.18 FIA_X509_EXT.1.1 Test #1 62

7.1.19 FIA_X509_EXT.1.1 Test #2 64
7.1.20 FIA_X509_EXT.1.1 Test #3 65
7.1.21 FIA_X509_EXT.1.1 Test #4 - CRL 65
7.1.22 FIA_X509_EXT.1.1 Test #5 66
7.1.23 FIA_X509_EXT.1.1 Test #6 66
7.1.24 FIA_X509_EXT.1.1 Test #7 66
7.1.25 FIA_X509_EXT.1.2 Test #1 66
7.1.26 FIA_X509_EXT.1.2 Test #2 67
7.1.27 FIA_X509_EXT.2 Test #1 67
7.1.28 FIA_X509_EXT.2 Test #2 68
7.1.29 FMT_CFG_EXT.1.1 Test #1 68
7.1.30 FMT_CFG_EXT.1.1 Test #2 69
7.1.31 FMT_CFG_EXT.1.1 Test #3 69
7.1.32 FMT_CFG_EXT.1.2 Test #1 69
7.1.33 FMT_MEC_EXT.1.1 Test #1 70
7.1.34 FMT_MEC_EXT.1.1 Test #2 70
7.1.35 FMT_SMF.1.1 Test #1 71
7.1.36 FPR_ANO.1.1 Test #1 71
7.1.37 FPT_AEX_EXT.1.1 Test #1 71
7.1.38 FPT_AEX_EXT.1.2 Test #1 72
7.1.39 FPT_AEX_EXT.1.4 Test #1 72
7.1.40 FPT_API_EXT.1.1 Test #1 72
7.1.41 FPT_IDV_EXT.1.1 Test #1 73
7.1.42 FPT_LIB_EXT.1.1 Test #1 73
7.1.43 FPT_TUD_EXT.1.1 Test #1 73
7.1.44 FPT_TUD_EXT.1.2 Test #1 73
7.1.45 FPT_TUD_EXT.1.3 Test #1 74
7.1.46 FPT_TUD_EXT.2.1 Test #1 74
7.1.47 FPT_TUD_EXT.2.2 Test #1 74
7.1.48 FTP_DIT_EXT.1.1 Test #1 75
7.1.49 FTP_DIT_EXT.1.1 Test #2 75
7.1.50 FTP_DIT_EXT.1.1 Test #3 75
7.2 PKG_TLSC 76
7.2.1 FCS_TLSC_EXT.1.1 Test #1 76
7.2.2 FCS_TLSC_EXT.1.1 Test #2 76
7.2.3 FCS_TLSC_EXT.1.1 Test #3 77
7.2.4 FCS_TLSC_EXT.1.1 Test #4 77
7.2.5 FCS_TLSC_EXT.1.1 Test #5.1 77
7.2.6 FCS_TLSC_EXT.1.1 Test #5.2 78
7.2.7 FCS_TLSC_EXT.1.1 Test #5.3 78
7.2.8 FCS_TLSC_EXT.1.1 Test #5.4 78
7.2.9 FCS_TLSC_EXT.1.1 Test #5.5 79
7.2.10 FCS_TLSC_EXT.1.1 Test #5.6 79
7.2.11 FCS_TLSC_EXT.1.1 Test #5.7 79

7.2.12 FCS_TLSC_EXT.1.2 Test #1 80
7.2.13 FCS_TLSC_EXT.1.2 Test #2 80
7.2.14 FCS_TLSC_EXT.1.2 Test #3 80
7.2.15 FCS_TLSC_EXT.1.2 Test #4 81
7.2.16 FCS_TLSC_EXT.1.2 Test #5.1 81
7.2.17 FCS_TLSC_EXT.1.2 Test #5.2(a) 82
7.2.18 FCS_TLSC_EXT.1.2 Test #5.2(b) 82
7.2.19 FCS_TLSC_EXT.1.2 Test #5.2(c) 83
7.2.20 FCS_TLSC_EXT.1.2 Test #5.3(a) 83
7.2.21 FCS_TLSC_EXT.1.2 Test #5.3(b) 84
7.2.22 FCS_TLSC_EXT.1.2 Test #5.4 84
7.2.23 FCS_TLSC_EXT.1.2 Test #6 85
7.2.24 FCS_TLSC_EXT.1.2 Test #7 85
7.2.25 FCS_TLSC_EXT.1.3 Test #1(a) 85
7.2.26 FCS_TLSC_EXT.1.3 Test #1(b) 86
7.2.27 FCS_TLSC_EXT.1.3 Test #1(c) 86
7.2.28 FCS_TLSC_EXT.1.3 Test #2 86
7.2.29 FCS_TLSC_EXT.1.3 Test #3 86
7.2.30 FCS_TLSC_EXT.1.3 Test #4 87
7.2.31 FCS_TLSC_EXT.5.1 Test #1 87
8 Conclusion 88

1 TOE Overview

The Target of Evaluation (TOE) is the Nubo Client Version 3.2. It is a thin client application installed and executed on an Android mobile device. The TOE establishes communications to a Virtual Mobile Infrastructure (VMI) platform (using a remote display protocol) and remotely displays the virtual apps that are running within the VMI platform. No output is displayed from other applications. The TOE only connects the mobile device to the virtual servers and is not responsible for the execution of the virtual apps.

With VMI, virtual applications execute on a user's behalf on VMI servers. No executable code associated with the virtual applications is downloaded to the user's device. Instead, the TOE displays the output from the virtual applications, and forwards input from the user to the virtual applications.

The TOE controls all communication between itself and the VMI environment. The TOE is only to be used with the Nubo Management Server and the Nubo Gateway. This ensures that all communication occurs over a secure connection within a secure remote application infrastructure. All network connections are initiated by the TOE. Connection requests by a VMI server are not accepted.

Direct connection is established between the TOE and the Nubo Management Server. The Nubo Management Server processes user activation and login and communicates with the TOE and the Nubo Gateway. The Nubo Gateway implements the connection for executing the virtual applications. The traffic for the virtual applications (that are transmitted from the VMI platform to the Nubo Gateway) is sent over a single trusted channel between the Management Server and the TOE.

The user installs the TOE from the Google Play Store. The app store contains a generic version of the Android app which does not contain any user credentials or details. Initially, TOE user credentials are sent to the Management Server, the Management Server registers the TOE user, the user activates the TOE, and connects to the Nubo Management Server. Once registered, the user is required to authenticate itself to the Management server on successive sessions with the VMI environment.

2 Assurance Activities Identification

The TOE assurance requirements are taken directly from the *Protection Profile for Application Software* Version 1.4 which are derived from Common Criteria Version 3.1, Revision 5 and the *Functional Package for Transport Layer Security (TLS)*, Version 1.1.

3 Test Equivalency Justification

The TOE tested on the following mobile device. There are no test equivalences.

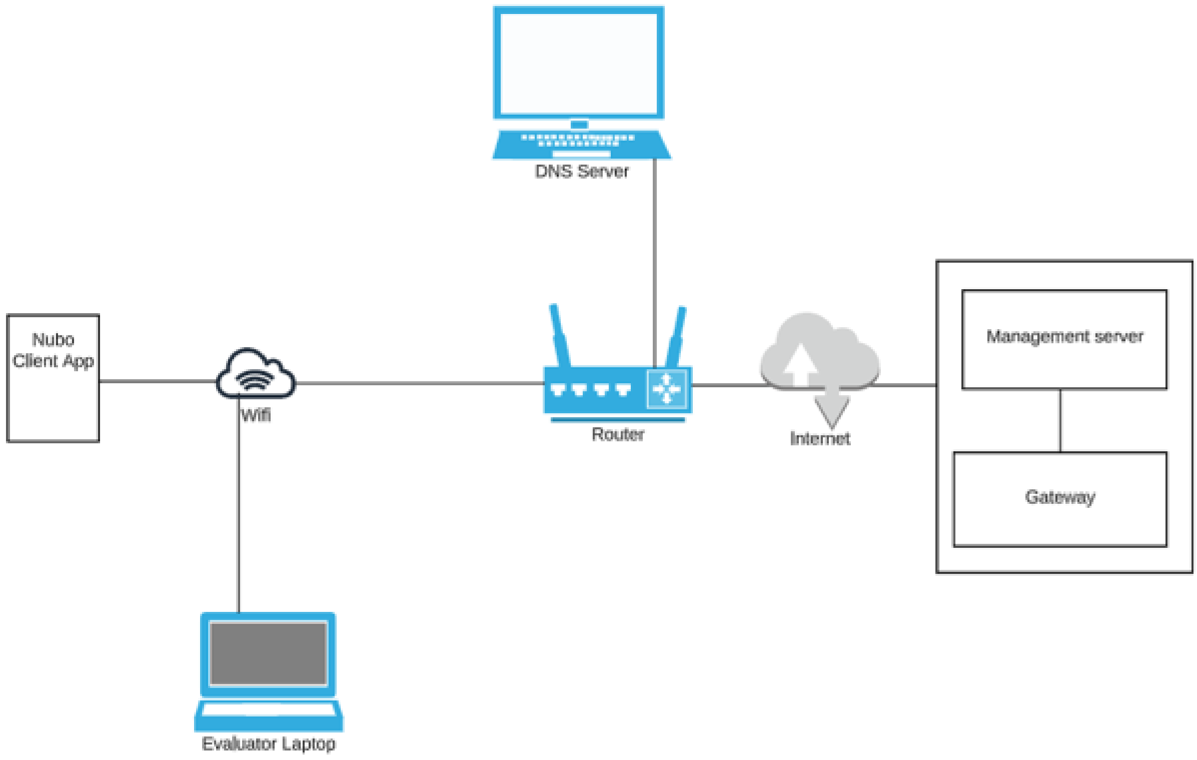
Device Name	Chipset Vendor	SoC	Base Model Number	32 bits/64 bits
Galaxy S10	Qualcomm	Snapdragon 855	SM-G973	64 bits

4 Test Bed Descriptions

4.1 Test Bed

The following figure depicts the Test Bed.

Figure 1: Test Bed



4.2 Configuration Information

Table 1: Test Configuration Information

Name	OS	Function	Protocols	Time	Tools (version)
Nubo Client Version 3.2	Android 12	TOE running on a Galaxy S10 phone with Qualcomm Snapdragon 855 processor with Android 12 OS installed.	TLS 1.2	Manually set and verified	
Evaluator Laptop	Windows 10	Test workstation	SSSHv2	Manually set and verified	MobaXterm 22.0 Wireshark 3.6.3

Name	OS	Function	Protocols	Time	Tools (version)
DNS Server	Ubuntu 20.04.4	DNS resolver	DNS	Manually set and verified	N/A
Test Virtual Machine	Ubuntu 20.04.4	Management Server Host running Management Server and Gateway Server	TLS 1.2/SSHv2	Manually set and verified	Wireshark 3.6.7 Nmap 7.93 OpenSSL 1.1.1 Acumen-tlsc-v2.1
Network Router (monsoon network)	Meraki MX 84 v18.107	Lab Router	N/A	Manually set and verified	N/A

4.3 Test Time and Location

All testing was carried out at the Acumen Security offices located in 2400 Research Blvd Suite #395, Rockville, MD 20850. Testing occurred from February 2023 to September 2023.

The TOE was in a physically protected, access controlled, designated test lab with no unattended entry/exit ways. At the start of each day, the test bed was verified to ensure that it was not compromised. All evaluation documentation was always kept in a secure repository.

5 Detailed Test Cases (TSS and AGD Activities)

5.1 Mandatory Requirements

5.1.1 Cryptographic Support (FCS)

5.1.1.1 **FCS_CKM_EXT.1 Cryptographic Key Generation Services (Applied TD0717)**

5.1.1.1.1 FCS_CKM_EXT.1.1 TSS

Objective:

- The evaluator shall inspect the application and its developer documentation to determine if the application needs asymmetric key generation services.
- If not, the evaluator shall verify the generate no asymmetric cryptographic keys selection is present in the ST.
- Otherwise, the evaluation activities shall be performed as stated in the selection-based requirements.

Evaluator Findings:

- The evaluator reviewed the "**TOE Summary Specification**" section in the Security Target and has determined that the TOE implements asymmetric key generation based on the selection made in the SFR.
- Keys that the TOE generates are 2048-bit RSA keys used specifically for digital signatures.
- The TOE implements ECDSA Key Generation, Signature Generation, and Signature Verification as part of TLS trusted channel establishment. NIST curves P-256 and P-384 are supported.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.1.1.2 FCS_CKM_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.1.2 **FCS_RBG_EXT.1 Random Bit Generation Services**

5.1.1.2.1 FCS_RBG_EXT.1.1 TSS

Objective:

- If "implement DRBG functionality" is selected, the evaluator shall review the TSS to ensure that additional FCS_RBG_EXT.2 elements are included in the ST.
- The evaluator shall examine the TSS to confirm that it identifies all functions (as described by the SFRs included in the ST) that obtain random numbers from the platform RBG.
- The evaluator shall determine that for each of these functions, the TSS states which platform interface (API) is used to obtain the random numbers.
- The evaluator shall confirm that each of these interfaces corresponds to the acceptable interfaces listed for each platform below.

Evaluator Findings:

- The evaluator reviewed the Security Target and determined that **'implement DRBG functionality'** has been selected for in the SFR. As such, FCS_RBG_EXT.2 has been included as part of the scope of the evaluation.
- The section labeled 'TOE Summary Specification' in the Security Target identifies that the DRBG functionality uses the BoringCrypto modules of the BoringSSL Library. The TOE seeds it's DRBG using the /dev/random API call.
- Based on these findings, this assurance activity is considered satisfied

Verdict: Pass.

5.1.1.2.2 FCS_RBG_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.1.3 **FCS_STO_EXT.1 Storage of Credentials**

5.1.1.3.1 FCS_STO_EXT.1 TSS

Objective:

- The evaluator shall check the TSS to ensure that it lists all persistent credentials (secret keys, PKI private keys, or passwords) needed to meet the requirements in the ST.
- For each of these items, the evaluator shall confirm that the TSS lists for what purpose it is used, and how it is stored.

Evaluator Findings:

- The evaluator reviewed the "**TOE Summary Specification**" section in the Security Target and ensured that it lists all persistent credentials (User full name, User email address, User password, and User Pass Coden Key) needed to meet the requirements in the ST.
- The evaluator reviewed the TSS "**TOE Summary Specification"** section in the Security Target and ensured that it lists for what purpose it is used, and how it is stored.
- Upon investigation, the evaluator found that the TSS states that: User credentials are User full name, User email address, User password, and User Pass Code. User credentials are encrypted by the platform with AES-CBC using a 256-bit key. The encryption key is stored in Android private keystore.
- The TOE also stores the Client Activation Key is also stored in Android private keystore.
- The TSS section describes the user of the Client Activation Key.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.1.3.2 FCS_STO_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.1.4 **FCS_TLS_EXT.1 TLS Protocol**

5.1.1.4.1 FCS_TLS_EXT.1.1 TSS

Objective: There are no TSS activities for this requirement.

5.1.1.4.2 FCS_TLS_EXT.1.1 Guidance

Objective:

- The evaluator shall ensure that the selections indicated in the ST are consistent with selections in the dependent components.

Evaluator Findings:

- The evaluator checked the AGD and ensured that the selections indicated in the ST are consistent with selections in the dependent components.
- The evaluator reviewed the AGD section **"Network Configuration"** and determined that the guidance identifies the TOE as a TLS Client.
- Upon investigation, the evaluator found that the AGD activity states that: "The TOE acts as a TLS Client communicating with the Nubo Management Server Host".

Verdict: Pass.

5.1.2 User Data Protection (FDP)

5.1.2.1 **FDP_DAR_EXT.1 Encryption Of Sensitive Application Data**

5.1.2.1.1 FDP_DAR_EXT.1.1 TSS

Objective:

- The evaluator shall examine the TSS to ensure that it describes the sensitive data processed by the application.
- The evaluator shall then ensure that the following activities cover all of the sensitive data identified in the TSS.
- If "not store any sensitive data" is selected, the evaluator shall inspect the TSS to ensure that it describes how sensitive data cannot be written to non-volatile memory.
- The evaluator shall also ensure that this is consistent with the filesystem test below.

Evaluator Findings:

- The evaluator examined the section titled "**TOE Summary Specification**" in the Security Target to verify that the TSS describes the sensitive data processed by the application. The TSS states that all sensitive data is stored by the TOE in /data/data/package/shared_prefs/ with the MODE_PRIVATE flag set and encrypted by the platform using AES-CBC using a 256-bit key.
- Upon investigation, the evaluator found that the TSS states provides a list of sensitive data that the TOE processes. These data types include: the User's full name, User email address, User password, and User passcode.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.2.1.2 FDP_DAR_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.2.2 **FDP_DEC_EXT.1 Access to Platform Resources**

5.1.2.2.1 FDP_DEC_EXT.1.1 TSS

Objective: There are no TSS activities for this requirement.

5.1.2.2.2 FDP_DEC_EXT.1.1 Guidance

Objective:

- The evaluator shall perform the platform-specific actions below and inspect user documentation to determine the application's access to hardware resources.
- The evaluator shall ensure that this is consistent with the selections indicated.
- The evaluator shall review documentation provided by the application developer and for each resource which it accesses, identify the justification as to why access is required.

Evaluator Findings:

- The evaluator checked the section **"TOE Access to Platform Resources"** in the AGD and ensured that the application has access to the stated hardware resources which is consistent with the SFR selections.
- Upon investigation, the evaluator found that the AGD states that network connectivity, camera, microphone, location services, and Bluetooth are the hardware platform resources accessed by the TOE. Additionally, the AGD states This access aims to support the user's access of applications in the Nubo VMI.
- The evaluator also examined the section titled **"TOE Access to Platform Resources"** in the AGD to verify that the stated hardware access is consistent with the results obtained from the test assurance activities. Upon investigation, the evaluator found that the hardware access information is consistent.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.2.2.3 FDP_DEC_EXT.1.2 TSS

Objective: There are no TSS activities for this requirement.

5.1.2.2.4 FDP_DEC_EXT.1.2 Guidance

Objective:

- The evaluator shall perform the platform-specific actions below and inspect user documentation to determine the application's access to sensitive information repositories.
- The evaluator shall ensure that this is consistent with the selections indicated.
- The evaluator shall review documentation provided by the application developer and for each sensitive information repository which it accesses, identify the justification as to why access is required.

Evaluator Findings:

- The ST was reviewed, and the selection stated, "no sensitive information repositories".
- The evaluator examined the section **"TOE Access to Platform Resources"** in the AGD to identify, AGD that "No sensitive information repositories are accessed".
- The test assurance activities is consistent with the AGD.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.2.3 **FDP_NET_EXT.1 Network Communications**

5.1.2.3.1 FDP_NET_EXT.1.1 TSS

Objective: There are no TSS activities for this requirement.

5.1.2.3.2 FDP_NET_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.3 Security Management (FMT)

5.1.3.1 **FMT_CFG_EXT.1 Secure by Default Configuration**

5.1.3.1.1 FMT_CFG_EXT.1.1 TSS

Objective:

The evaluator shall check the TSS to determine if the application requires any type of credentials and if the application installs with default credentials.

Evaluator Findings:

- The evaluator examined the section titled "**TOE Summary Specification**", section FMT_CFG_EXT.1 in the Security Target and has determined that there are no default credentials within the TOE.
- Upon investigation, the evaluator found that the TSS states that the TOE is not installed with any default credentials. Users of the TOE must activate with the Nubo Management Server before any other applications can be accesses.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.3.1.2 FMT_CFG_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.3.1.3 FMT_CFG_EXT.1.2 TSS

Objective: There are no TSS activities for this requirement.

5.1.3.1.4 FMT_CFG_EXT.1.2 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.3.2 **FMT_MEC_EXT.1 Supported Configuration Mechanism**

5.1.3.2.1 FMT_MEC_EXT.1.1 TSS

Objective:

- The evaluator shall review the TSS to identify the application's configuration options (e.g., settings) and determine whether these are stored and set using the mechanisms supported by the platform or implemented by the application in accordance with the PP-Module for File Encryption.
- At a minimum the TSS shall list settings related to any SFRs and any settings that are mandated in the AGD in response to an SFR.

- Conditional: If "implement functionality to encrypt and store configuration options as defined by FDP_PRT_EXT.1 in the PP-Module for File Encryption" is selected, the evaluator shall ensure that the TSS identifies those options, as well as indicates where the encrypted representation of these options is stored.

Evaluator Findings:

- The evaluator examined the Security Target and determined that the TOE invokes the recommended mechanisms by the platform vendor for storing and setting configuration options.
- The evaluator reviewed the AGD and section "Account Registration and Activation" states the phone must be registered and activated to configure the phone in the Common Criteria evaluated configuration. The section states "Follow the following steps to configure the user creation data (user full name, user email address, and job title)".
- Upon investigation, within the section labeled "**TOE Summary Specification**" , FMT_MEC_EXT.1, it is stated that the TSS states that the TOE stores configuration data in `/data/data/package/shared_prefs/` with the MODE_PRIVATE flag set. The section states that configuration data is "the URL of the Nubo Management Server and user creation data".
- The evaluator reviewed the AGD and section "Nubo Manager Server Host URL" states that "The TOE has the default URL of the Nubo Management Server Host hard coded in the build. Therefore, the user does not have to configure the URL".
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.3.2.2 FMT_MEC_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.3.3 **FMT_SMF.1 Specification of Management Functions**

5.1.3.3.1 FMT_SMF.1.1 TSS

Objective: There are no TSS activities for this requirement.

5.1.3.3.2 FMT_SMF.1.1 Guidance

Objective:

The evaluator shall verify that every management function mandated by the PP is described in the AGD and that the description contains the information required to perform the management duties associated with the management function.

Evaluator Findings:

- The evaluator examined the AGD to verify that every management function mandated by the PP is described in the operational guidance and that the description contains the information required to perform the management duties associated with the management function.
- Upon investigation, the evaluator found that the AGD states that the TOE allows upgrading of TOE in section "**Updating the TOE**".
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.4 Privacy (FPR)

5.1.4.1 **FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information**

5.1.4.1.1 FPR_ANO_EXT.1.1 TSS

Objective:

The evaluator shall inspect the TSS documentation to identify functionality in the application where PII can be transmitted.

Evaluator Findings:

- The evaluator examined the section titled "TOE Summary Specification (FPR_ANO_EXT.1)" in the Security Target to verify that the TSS identifies functionality in the application where PII can be transmitted.
- Upon investigation, the evaluator found that the TSS states that user related PII is transmitted to the Management server during activation of the TOE. When the user interacts with the GUI the TOE asks for consent from the user before sending information. The GUI also provides notice to the user that PII is being transmitted over a secure connection.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.4.1.2 FPR_ANO_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5 Protection of the TSF (FPT)

5.1.5.1 **FPT_AEX_EXT.1 Anti-Exploitation Capabilities**

5.1.5.1.1 *FPT_AEX_EXT.1.1 TSS* (TD0798)

Objective:

The evaluator shall ensure that the TSS describes the compiler flags used to enable ASLR when the application is compiled. If any explicitly-mapped exceptions are claimed, the evaluator shall check that the TSS identifies these exceptions, describes the static memory mapping that is used, and provides justification for why static memory mapping is appropriate in this case.

Evaluator Findings:

- The evaluator examined the section titled "**TOE Summary Specification**", FPT_AEX_EXT.1, in the Security Target and the SFR selection to verify that TOE does not map memory.
- Upon investigation, the evaluator found that the TSS supports that selection for this requirement by stating that because the TOE is a Java application memory mapping and permission on the memory regions are not accessible to the TOE.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.5.1.2 FPT_AEX_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.1.3 FPT_AEX_EXT.1.2 TSS

Objective: There are no TSS activities for this requirement.

5.1.5.1.4 FPT_AEX_EXT.1.2 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.1.5 FPT_AEX_EXT.1.3 TSS

Objective: There are no TSS activities for this requirement.

5.1.5.1.6 FPT_AEX_EXT.1.3 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.1.7 FPT_AEX_EXT.1.4 TSS

Objective: There are no TSS activities for this requirement.

5.1.5.1.8 FPT_AEX_EXT.1.4 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.1.9 FPT_AEX_EXT.1.5 TSS

Objective: There are no TSS activities for this requirement.

5.1.5.1.10 FPT_AEX_EXT.1.5 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.2 **FPT_API_EXT.1 Use of Supported Services and APIs**

5.1.5.2.1 FPT_API_EXT.1.1 TSS

Objective:

The evaluator shall verify that the TSS lists the platform APIs used in the application.

Evaluator Findings:

- The evaluator examined the section titled “TOE Summary Specification” (FPT_API_EXT.1) in the Security Target and confirms that the TSS lists the platform APIs used in the application.
- Upon investigation, the evaluator found that the TSS states that the TOE uses the following Android Java APIs:
    - android.security.keystore.KeyGenParameterSpec
    - android.security.keystore.KeyProperties
    - java.security.InvalidAlgorithmParameterException
    - java.security.InvalidKeyException
    - java.security.Key
    - java.security.KeyManagementException
    - java.security.KeyStore
    - java.security.KeyStoreException
    - java.security.MessageDigest
    - java.security.NoSuchAlgorithmException

  - java.security.NoSuchProviderException
  - java.security.PublicKey
  - java.security.SecureRandom
  - java.security.UnrecoverableKeyException
  - java.security.cert.Certificate
  - java.security.cert.CertificateException
  - java.security.cert.CertificateFactory
  - java.security.spec.InvalidParameterSpecException
  - javax.crypto.BadPaddingException
  - javax.crypto.Cipher
  - javax.crypto.IllegalBlockSizeException
  - javax.crypto.KeyGenerator
  - javax.crypto.Mac
  - javax.crypto.NoSuchPaddingException
  - javax.crypto.SecretKey
  - javax.crypto.SecretKeyFactory
  - javax.crypto.spec.IvParameterSpec
  - javax.crypto.spec.SecretKeySpec
  - javax.net.ssl.HttpsURLConnection
  - javax.net.ssl.SSLContext
  - javax.net.ssl.SSLPeerUnverifiedException
  - javax.net.ssl.SSLSession
  - javax.net.ssl.SSLSocket
  - javax.net.ssl.SSLSocketFactory
  - javax.security.auth.x500.X500Principal
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.5.2.2 FPT_API_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.3 **FPT_IDV_EXT.1 Software Identification and Versions**

5.1.5.3.1 FPT_IDV_EXT.1.1 TSS

Objective:

If "other version information" is selected the evaluator shall verify that the TSS contains an explanation of the versioning methodology.

Evaluator Findings:

- The evaluator examined the section titled “**TOE Summary Specification**”, FPT_IDV_EXT.1, in the Security Target which discusses the details about versioning for the TOE.
- Upon investigation, the evaluator found that within the TSS it is stated that the version number of the TOE can be displayed thought the app settings and also in the platform settings. If viewing information about the TOE on Google Play the version is also displayed. The model of format for the versioning is expressed as “3.X(.Y)” where the “3” is the major version, “X” is the minor version and “Y” is an optional build number.
- The evaluated TOE’s version is 3.2.

- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.5.3.2 FPT_IDV_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.4 **FPT_LIB_EXT.1 Use of Third Party Libraries**

5.1.5.4.1 FPT_LIB_EXT.1.1 TSS

Objective: There are no TSS activities for this requirement.

5.1.5.4.2 FPT_LIB_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.5 **FPT_TUD_EXT.1 Integrity for Installation and Update**

5.1.5.5.1 FPT_TUD_EXT.1.1 TSS

Objective: There are no TSS activities for this requirement.

5.1.5.5.2 FPT_TUD_EXT.1.1 Guidance

Objective:

The evaluator shall check to ensure the AGD includes a description of how updates are performed.

Evaluator Findings:

- The evaluator examined the section titled "**Updating the TOE**" in the AGD to verify that it includes a description of how updates are performed.
- Upon investigation, the evaluator found that the AGD provides the instructions for an update. This includes the process of checking for the current version and installation of the new version. Updates and patches to the TOE are distributed by the Google Play Store (APK format). The updates are signed with a Nubo Software certificate prior to installation. The user may query the current version of the TOE by following the section titled "**Verifying Product Versioning**" in the AGD.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.5.5.3 FPT_TUD_EXT.1.2 TSS

Objective: There are no TSS activities for this requirement.

5.1.5.5.4 FPT_TUD_EXT.1.2 Guidance

Objective:

The evaluator shall verify the AGD includes a description of how to query the current version of the application.

Evaluator Findings:

- The evaluator examined the section titled **"Verifying Product Versioning"** in the AGD to verify that it includes a description of how to query the current version of the application.

- Upon investigation, the evaluator found that the AGD states that the user may query the current version of the TOE by entering the Settings screen, which is available from the Nubo icon navigating to the Settings menu.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.5.5.5 FPT_TUD_EXT.1.3 TSS

Objective: There are no TSS activities for this requirement.

5.1.5.5.6 FPT_TUD_EXT.1.3 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.5.7 FPT_TUD_EXT.1.4 TSS

Objective:

- The evaluator shall verify that the TSS identifies how updates to the application are signed by an authorized source. The definition of an authorized source must be contained in the TSS.
- The evaluator shall also ensure that the TSS (or the AGD) describes how candidate updates are obtained.

Evaluator Findings:

- The evaluator examined the section titled **"TOE Summary Specification"**, FPT_TUD_EXT.1, in the Security Target and has verified that the TSS provides information on how the application installation package and updated are signed by an authorized source.
- Upon investigation, the evaluator found that the TSS states that the TOE's initial installation is downloaded from Google Play Store, installed by the user, and the app is signed by Nubo. As the installation package is signed by the TOE's developer it is considered an authorized source. In the event that the user downloads an APK which is not signed by Nubo, or fails signature validation, the platform will not allow for the installation of the package to take place.
- Updates and patches are distributed by the Google Play Store. Each release is in a packaged APK format and is signed with a Nubo Software certificate. The evaluator had determined that as the certificate is from the developer that it is considered an authorized source.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.5.5.8 FPT_TUD_EXT.1.4 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.5.5.9 FPT_TUD_EXT.1.5 TSS

Objective:

- The evaluator shall verify that the TSS identifies how the application is distributed.
- If "with the platform" is selected the evaluated shall perform a clean installation or factory reset to confirm that TOE software is included as part of the platform OS.

Evaluator Findings:

- The evaluator examined the selection for this requirement in the Security Target and determined that the TOE is an additional software package to the platform OS but is not "with the platform."
- Upon investigation, the evaluator found that the TSS states that the TOE is installed by the user which is interpreted as not being part of the platform OS. This corresponds with the selection made for FPT_TUD_EXT.1.5.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.5.5.10 FPT_TUD_EXT.1.5 Guidance

Objective: There are no Guidance activities for this requirement.

5.1.6 Trusted Path/Channels (FTP)

5.1.6.1 **FTP_DIT_EXT.1 Protection of Data in Transit**

5.1.6.1.1 FTP_DIT_EXT.1.1 TSS

Objective:

For platform-provided functionality, the evaluator shall verify the TSS contains the calls to the platform that TOE is leveraging to invoke the functionality.

Evaluator Findings:

- The evaluator examined "**TOE Summary Specification**", FTP_DIT_EXT.1, the Security Target and determined that the TOE relies on the platform for HTTP and calls the javax.net.ssl.HttpsURLConnection API. That function then calls the TOE for the TLS layer (Conscrypt BoringSSL).
- The evaluator examined the SFRs and confirmed javax.net.ssl.HttpsURLConnection is identified in FPT_LIB_EXT.1.1.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.1.6.1.2 FTP_DIT_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.2 Optional Requirements

There are no optional requirements claimed.

5.3 Selection-Based Requirements

5.3.1 Cryptographic Support (FCS)

5.3.1.1 **FCS_CKM.1/AK Cryptographic Asymmetric Key Generation (Applied TD0717)**

5.3.1.1.1 FCS_CKM.1.1/AK TSS

Objective:

- The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE.
- If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.
- If the application "invokes platform-provided functionality for asymmetric key generation," the evaluator shall examine the TSS to verify that it describes how the key generation functionality is invoked.

Evaluator Findings:

- The evaluator has reviewed section labeled 'TOE Summary Specification' in the Security Target as well as the selection made in the SFR.
- The TOE generates 2048-bit RSA key sizes for use with digital signatures.
- The TOE implements ECDSA Key Generation, Signature Generation, and Signature Verification as part of TLS trusted channel establishment. NIST curves P-256 and P-384 are supported.
- The TOE does not invoke platform provided functionality but claims the selection for 'implement functionality'.

Verdict: Pass.

5.3.1.1.2 FCS_CKM.1.1/AK Guidance

Objective:

The evaluator shall verify that the AGD instructs the administrator how to configure the TOE to use the selected key generation scheme(s) and key size(s) for all uses defined in this PP.

Evaluator Findings:

- The evaluator checked the AGD and ensured that it instructs the administrator how to configure the TOE to use the selected key generation scheme(s) and key size(s) for all uses defined in this PP.
- Upon investigation, the evaluator found that the AGD activity states in section "**Network Configuration**", "Aside from identifying the Nubo Management Server, there is no additional configuration needed for TLS communication".
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.2 **FCS_CKM.2 Cryptographic Key Establishment (TD0717)**

5.3.1.2.1 FCS_CKM.2.1 TSS

Objective:

- The evaluator shall ensure that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1/AK.
- If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Evaluator Findings:

- The evaluator reviewed section 6 TOE Summary Specification in the Security Target and ensured that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1/AK.
- Upon investigation, the evaluator found that the TSS states that the TOE uses Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) as the key agreement, and ECDSA is claimed in FCS_CKM.1/AK and EC is claimed in FCS_CKM.1/AK and FCS_CKM.2.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.2.2 FCS_CKM.2.1 Guidance

Objective:

- The evaluator shall verify that the AGD instructs the administrator how to configure the TOE to use the selected key establishment scheme(s).

Evaluator Findings:

- The evaluator checked the AGD and ensured that it instructs the administrator how to configure the TOE to use the selected key establishment scheme(s).
- Upon investigation, the evaluator found that the AGD activity states in section "**Network Configuration**", "Aside from identifying the target Nubo Management Server, there is no additional configuration needed for TLS communication".
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.3 **FCS_COP.1/Hash Cryptographic Operation – Hashing (Applied TD0717)**

5.3.1.3.1 FCS_COP.1.1/Hash TSS

Objective:

The evaluator shall check that the association of the hash function with other application cryptographic functions (for example, the digital signature verification function) is documented in the TSS.

Evaluator Findings:

- The evaluator examined the section titled 'TOE Summary Specification' in the Security Target and has reviewed the hash function with the other cryptographic functions.
- Upon investigation, the evaluator found that the TSS states that the following algorithms are implemented:
    - AES-GCM 256-bit encryption and decryption
    - SHA-256 and SHA-384 Hash function used in association with digital signature computation
    - HMAC-SHA-256 and HMACH-SHA-384 keyed-hash message authentication
    - Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) key establishment
    - 2048 bit RSA for signature computation
- The evaluator has determined that the relationship between the hash function and other cryptographic functions is appropriately documented.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.3.2 FCS_COP.1.1/Hash Guidance

Objective: There are no Guidance activities for this requirement.

5.3.1.4 **FCS_COP.1/KeyedHash Cryptographic Operation - Keyed-Hash Message Authentication (Applied TD0717)**

5.3.1.4.1 FCS_COP.1.1/KeyedHash TSS

Objective: There are no TSS activities for this requirement.

5.3.1.4.2 FCS_COP.1.1/KeyedHash Guidance

Objective: There are no Guidance activities for this requirement.

5.3.1.5 **FCS_COP.1/Sig Cryptographic Operation – Signing (Applied TD0717)**

5.3.1.5.1 FCS_COP.1.1/Sig TSS

Objective: There are no TSS activities for this requirement.

5.3.1.5.2 FCS_COP.1.1/Sig Guidance

Objective: There are no Guidance activities for this requirement.

5.3.1.6 **FCS_COP.1/SKC Cryptographic Operation - Encryption/Decryption (Applied TD0717)**

5.3.1.6.1 FCS_COP.1.1/SKC TSS

Objective: There are no TSS activities for this requirement.

5.3.1.6.2 FCS_COP.1.1/SKC Guidance

Objective:

- The evaluator checks the AGD documents to determine that any configuration that is required to be done to configure the functionality for the required modes and key sizes is present.

Evaluator Findings:

- The evaluator checked the AGD and ensured that it provides documentation to determine that any configuration that is required to be done to configure the functionality for the required modes and key sizes is present.
- Upon investigation, the evaluator found that the AGD activity states in section "**Network Configuration**", "Aside from identifying the target Nubo Management Server, there is no additional configuration needed for TLS communication".
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.7 FCS_HTTPS_EXT.1/Client HTTPS Protocol

5.3.1.7.1 FCS_HTTPS_EXT.1.1/Client TSS

Objective:

The evaluator shall examine the TSS and determine that enough detail is provided to explain how the implementation complies with RFC 2818.

Evaluator Findings:

- The evaluator examined the section titled 'TOE Summary Specification' in the Security Target to verify that the TSS provides enough detail to explain how the implementation complies with RFC 2818.
- Upon investigation, the evaluator found that the TSS states that the TOE uses platform-provided APIs (javax.net.ssl.HttpsURLConnection) to implement HTTPS, and sets up an SSL Socket factory which uses the implemented TLS functionality.
- The TLS implementation used for HTTPS is in full accordance with RFC2818.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.7.2 FCS_HTTPS_EXT.1.1/Client Guidance

Objective: There are no Guidance activities for this requirement.

5.3.1.7.3 FCS_HTTPS_EXT.1.2/Client TSS

Objective: There are no TSS activities for this requirement.

5.3.1.7.4 FCS_HTTPS_EXT.1.2/Client Guidance

Objective: There are no Guidance activities for this requirement.

5.3.1.7.5 FCS_HTTPS_EXT.1.3/Client TSS

Objective: There are no TSS activities for this requirement.

5.3.1.7.6 FCS_HTTPS_EXT.1.3/Client Guidance

Objective: There are no Guidance activities for this requirement.

5.3.1.8 FCS_RBG_EXT.2 Random Bit Generation from Application

5.3.1.8.1 FCS_RBG_EXT.2.1 TSS

Objective: There are no TSS activities for this requirement.

5.3.1.8.2 FCS_RBG_EXT.2.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.3.1.8.3 FCS_RBG_EXT.2.2 TSS

Objective:

Documentation shall be produced – and the evaluator shall perform the activities – in accordance with Appendix C – Entropy Documentation and Assessment and the Clarification to the Entropy Documentation and Assessment Annex.

Evaluator Findings:

- The evaluator reviewed the **"TOE Summary Specification"** under FCS_RBG_EXT.2 and had found that the TSS has provided information directing the reader to review the Entropy Assessment Report (EAR) which is documentation that is included as part of the evaluation.
- As the TSS references the EAR this objective is considered satisfied.

Verdict: Pass

5.3.1.8.4 FCS_RBG_EXT.2.2 Guidance

Objective: There are no Guidance activities for this requirement.

5.3.1.9 **FCS_TLSC_EXT.1 TLS Client Protocol (Applied TD0442)**

5.3.1.9.1 FCS_TLSC_EXT.1.1 TSS

Objective:

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the cipher suites supported are specified. The evaluator shall check the TSS to ensure that the cipher suites specified include those listed for this component.

Evaluator Findings:

- The evaluator examined the section titled **"TOE Summary Specification"** in the Security Target to and was able to verify that the TSS states that the TOE implements TLS v1.2 by using Conscrypt and BoringSSL libraries included with the TOE. No other TLS versions are supported. The only supported cipher suite is:
  - TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- The cipher suite listed in the TSS is the only one listed for this component.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.9.2 FCS_TLSC_EXT.1.1 Guidance

Objective:

The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the product so that TLS conforms to the description in the TSS.

Evaluator Findings:

- The relevant information is found in the following section: **"TOE Summary Specification"**, SFR FCS_TLSC_EXT.1.
- Upon investigation, the evaluator found that the AGD activity states in section "**Network Configuration**", "Aside from information identified in Pre-Installation, there is no additional configuration needed for TLS communication".
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.9.3 FCS_TLSC_EXT.1.2 TSS

Objective:

- The evaluator shall ensure that the TSS describes the client's method of establishing all reference identifiers from the application-configured reference identifier, including which types of reference identifiers are supported (e.g., Common Name, DNS Name, URI Name, Service Name, or other application-specific Subject Alternative Names) and whether IP addresses and wildcards are supported.
- The evaluator shall ensure that this description identifies whether and the manner in which certificate pinning is supported or used by the product.

Evaluator Findings:

- The evaluator reviewed the TSS in the section labeled "**TOE Summary Specification"** in the Security Target which states that the TOE uses the Conscrypt and BoringSSL libraries for all certificate validations. Conscrypt and BoringSSL supports Common Name (CN) and Subject Alternative Name (SAN) (DNS and IP address) as reference identifiers.
- The TOE supports the use of wildcards in X.509 reference identifiers (CN and SAN). The TOE utilizes certificate pinning. Connections to the Nubo Management Server are pinned to certificates built into the TOE application.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.9.4 FCS_TLSC_EXT.1.2 Guidance

Objective:

- The evaluator shall verify that the AGD includes instructions for setting the reference identifier to be used for the purposes of certificate validation in TLS.

Evaluator Findings:

- The evaluator checked the AGD and ensured that it includes instructions for setting the reference identifier to be used for the purposes of certificate validation in TLS.
- Upon investigation, the evaluator found that the AGD activity states in section "**Network Configuration**", "Aside from identifying the Nubo Management Server, there is no additional configuration needed for TLS communication".
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.9.5 FCS_TLSC_EXT.1.3 TSS

Objective:

- If the selection for authorizing override of invalid certificates is made, then the evaluator shall ensure that the TSS includes a description of how and when user or administrator authorization is obtained.
- The evaluator shall also ensure that the TSS describes any mechanism for storing such authorizations, such that future presentation of such otherwise-invalid certificates permits establishment of a trusted channel without user or administrator action.

Evaluator Findings:

- The evaluator has reviewed the Security Target and based on the selection made for FCS_TLSC_EXT.1.3 the TOE does not select for authorizing overrides of invalid certificates.
- As the selection is not made to require additional feedback in the TSS the evaluator has determined this assurance activity to be satisfied.

Verdict: Pass.

5.3.1.9.6 FCS_TLSC_EXT.1.3 Guidance

Objective: There are no Guidance activities for this requirement.

5.3.1.10 **FCS_TLSC_EXT.5 Client Support for Supported Groups Extension**

5.3.1.10.1 FCS_TLSC_EXT.5.1 TSS

Objective:

The evaluator shall verify that TSS describes the Supported Groups Extension.

Evaluator Findings:

- The evaluator reviewed the section labeled "**TOE Summary Specification**" in the Security Target and found that that TSS listed the only supported group extension which is:
  - secp384r1
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.1.10.2 FCS_TLSC_EXT.5.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.3.2 Identification and Authentication (FIA)

5.3.2.1 **FIA_X509_EXT.1 X.509 Certificate Validation**

5.3.2.1.1 FIA_X509_EXT.1.1 TSS

Objective:

- The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place.
- The evaluator shall ensure the TSS also provides a description of the certificate path validation algorithm.

Evaluator Findings:

- The evaluator examined the section titled **"TOE Summary Specification"** in the Security Target for verifying where the check of validity of the certificates takes place and the certificate path validation algorithm.
- Upon investigation, the evaluator found that the TSS states that All certificate validations are implemented using the Conscrypt and BoringSSL libraries. The libraries also validate X.509v3 certificates and check their revocation status. The TOE checks the validity of all imported CA certificates by checking for the presence of the basicConstraints extension and that the CA flag is set to TRUE as the TOE imports the certificate into the TOE's Trust Anchor Database.

- If the TOE detects an absence of the basicConstraints extension or the CA flag, the TOE imports the certificate as a user public key and adds it to the keystore (instead of the Trust Anchor Database). The TOE also checks for the presence of the basicConstraints extension and CA flag in each CA certificate presented in a peer server's certificate chain. Similarly, the TOE verifies the extendedKeyUsage Server Authentication purpose during certificate validation.
- Additionally, the certificate path validation algorithm examines each certificate in the path starting with the peer's certificate:
  - It first checks the validity of that certificate (e.g., the certificate has not expired, the certificate is valid, whether the certificate contains the appropriate X.509 extensions [e.g., the CA flag in the basic constraints extension for a CA certificate, or that a server certificate contains the Server Authentication purpose in the extendedKeyUsage field]).
  - The algorithm then verifies each certificate in the chain (applying the same rules but also ensuring that the Issuer of each certificate matches the Subject in the next rung "up" in the chain and that the chain ends in a self-signed certificate present in either the TOE's trusted anchor database or matches a specified Root CA).
  - Finally, the TOE checks the revocation status of all certificates in the chain.
  - OCSP as specified in RFC 2560 and CRL checking as specified in RFC 5280 Section 6.3 revocation checking will be attempted on certificates that have listed distribution points.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.2.1.2 FIA_X509_EXT.1.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.3.2.1.3 FIA_X509_EXT.1.2 TSS

Objective: There are no TSS activities for this requirement.

5.3.2.1.4 FIA_X509_EXT.1.2 Guidance

Objective: There are no Guidance activities for this requirement.

5.3.2.2 **FIA_X509_EXT.2 X.509 Certificate Authentication**

5.3.2.2.1 FIA_X509_EXT.2 TSS

Objective:

- The evaluator shall check the TSS to ensure that it describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.
- The evaluator shall examine the TSS to confirm that it describes the behavior of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel.
- The evaluator shall verify that any distinctions between trusted channels are described.
- If the requirement that the administrator is able to specify the default action, the evaluator shall ensure that the AGD contains instructions on how this configuration action is performed.

Evaluator Findings:

- The evaluator examined the section titled **"TOE Summary Specification"** in the Security Target to verify that the TSS describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.
- Upon investigation, the evaluator found that the TSS states that the TOE uses X.509v3 certificates as defined by RFC 5280 for server authentication using HTTPS/TLS connections. Certificates for the Nubo Management Server are built into the TOE application. As the certificates for the Management Server, the evaluator has determined that no additional guidance for configuring the OE is needed.
- When revocation status cannot be determined, certificates are not accepted.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.2.2.2 FIA_X509_EXT.2 Guidance

Objective: There are no Guidance activities for this requirement.

5.3.3 Protection of the TSF (FPT)

5.3.3.1 FPT_TUD_EXT.2 Integrity for Installation and Update (Applied TD0628)

5.3.3.1.1 FPT_TUD_EXT.2.1 TSS

Objective: There are no TSS activities for this requirement.

5.3.3.1.2 FPT_TUD_EXT.2.1 Guidance

Objective: There are no Guidance activities for this requirement.

5.3.3.1.3 FPT_TUD_EXT.2.2 TSS

Objective: There are no TSS activities for this requirement.

5.3.3.1.4 FPT_TUD_EXT.2.2 Guidance

Objective: There are no Guidance activities for this requirement.

5.3.3.1.5 FPT_TUD_EXT.2.3 TSS

Objective:

The evaluator shall verify that the TSS identifies how the application installation package is signed by an authorized source. The definition of an authorized source must be contained in the TSS.

Evaluator Findings:

- The evaluator examined the section titled **"TOE Summary Specification"** in the Security Target to verify that the TSS describes how the application installation package is signed by an authorized source.
- Upon investigation, the evaluator found that the TSS for FPT_TUD_EXT.2 states that The TOE is packaged in the Android application package (APK) format. Referencing the TSS for FPT_TUD_EXT.1 it is stated that each release packaged in APK format is signed with a Nubo Software certificate. As the signed certificate comes from the developer the evaluator has determined it to be an authorized source.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

5.3.3.1.6 FPT_TUD_EXT.2.3 Guidance

Objective: There are no Guidance activities for this requirement.

6 Security Assurance Requirements

6.1 Security Target (ASE)

There are no new Assurance Activities included in the PP for ASE.

6.2 Development (ADV)

6.2.1 ADV_FSP.1 Basic Functional Specification

6.2.1.1 **ADV_FSP.1.1E**

Objective:

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

Evaluator Findings:

- The evaluator confirmed that the Assurance Activities for AGD, ATE, and AVA satisfy this SAR.
- Based on these findings, this work unit is considered satisfied.

Verdict: Pass.

6.2.1.2 **ADV_FSP.1.2E**

Objective:

The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

Evaluator Findings:

- The evaluator confirmed that the Assurance Activities for AGD, ATE, and AVA satisfy this SAR.
- Based on these findings, this work unit is considered satisfied.

Verdict: Pass.

6.3 Guidance Documentation (AGD)

6.3.1 AGD_OPE.1 Operational User Guidance

6.3.1.1 **AGD_OPE.1.1E**

Objective:

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

Some of the contents of the AGD will be verified by the evaluation activities in Section 5.1 Security Functional Requirements and evaluation of the TOE according to the [CEM].

- If cryptographic functions are provided by the TOE, the AGD shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

- The documentation must describe the process for verifying updates to the TOE by verifying a digital signature – this may be done by the TOE or the underlying platform.
- The evaluator shall verify that this process includes the following steps:
  - Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).
  - Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the digital signature. The TOE will likely contain security functionality that does not fall in the scope of evaluation under this PP. The AGD shall make it clear to an administrator which security functionality is covered by the evaluation activities.

Evaluator Findings:

- Section "**Network Configuration**" of the AGD was used to determine that there is no configuration required for encryption other than the requirement to connect to the Nubo Management Server Host.
- Section "**Updating the TOE**" of the AGD describes how to update the TOE and includes that the user is required to register and activate the TOE before use.
- Section "**Update Verification**" of the AGD describes how to obtain updates and states that the platform will disallow the installation of the package if the update is not signed by Nubo.
- Section "**Security Target Introduction**" of the ST was used to determine if there is any functionality excluded from the TOE. Section "**Product Functionality not Included in the Scope of the Evaluation**" in the ST states that fingerprint authentication functionality is excluded in from the evaluation and only the default Nubo Management Server Host (cc.nubo.co) is supported. The evaluator then examined the AGD and determined that section "**Common Criteria Target of Evaluation (TOE) Description**" advises the user that fingerprint authentication functionality has been exclude from the Common Criteria evaluation and that only the default Nubo Management Server Host has been supported.
- Based on this, the assurance activity is considered satisfied.

Verdict: Pass.

6.3.2 AGD_PRE.1 Preparative Procedures

6.3.2.1 **AGD_PRE.1.1E and AGD_PRE.1.2E**

Objective:

AGD_PRE.1.1E - The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AGD_PRE.1.2E - The evaluator shall apply the preparative procedures to confirm that the TOE can be prepared securely for operation.

- The evaluator shall confirm that the documentation describes how to configure the TOE platform.
- The evaluator shall confirm that the documentation describes how to configure the TOE's Operational Environment.

Evaluator Findings:

- The evaluator examined section "**Supported Phone**" of AGD to identify the supported platform of the TOE.
- The evaluator examined section "**Physical Boundary**" of the ST and identified the operational environment.

- The evaluator examined section "**Pre-Installation**" of AGD and determined configuration requirements of the TOE platform required having Android 12 OS installed and having . Access to the Internet via WiFi or Mobile Network and knowing the URL of the Nubo Management Server are both required.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

6.4 Life-cycle Support (ALC)

6.4.1 ALC_CMC.1 Labeling of the TOE

6.4.1.1 **ALC_CMC.1.1E**

Objective:

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

- The evaluator shall check the ST to ensure that it contains an identifier (such as a product name/version number) that specifically identifies the version that meets the requirements of the ST.
- The evaluator shall check the AGD guidance to ensure that the version number is consistent with that in the ST.
- The evaluator shall check the TOE samples received for testing to ensure that the version number is consistent with that in the ST.
- If the vendor maintains a web site advertising the TOE, the evaluator shall examine the information on the web site to ensure that the information in the ST is sufficient to distinguish the product.

Evaluator Findings:

- The evaluator examined the ST section titled "**Security Target and TOE Identifier, TOE Reference, TOE Identification**" to verify that the ST contains an identifier "Nubo Client Version 3.2" that specifically identifies the version that meets the requirements of the ST.
- The evaluator examined the AGD, section titled "**Common Criteria Target of Evaluation (TOE) Description**" to verify that the AGD contains an identifier "Nubo Client Version 3.2" that specifically identifies the version number is consistent with that in the ST.
- The evaluator examined the AGD, section titled "**Installation, Verifying Product Versioning**" to verify that the TOE contains an identifier "Nubo 3.2x" that specifically identifies the version number is consistent with that in the ST.
- The evaluator examined the vendor's website "www.nubosoftware.com" and determined the vendor does not maintain a website advertising a Common Criteria evaluated version of their product.
- The evaluator examined the Nubo listing on Google Play to verify that the Nubo listing contains a version identifier of "3.2" and that it is consistent with that in the ST.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

6.4.2 ALC_CMS.1 TOE CM Coverage

6.4.2.1 **ALC_CMS.1.1E**

Objective:

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

- The evaluator shall ensure that the developer has identified (in AGD for application developers concerning the targeted platform) one or more development environments appropriate for use in developing applications for the developer's platform. For each of these development environments, the developer shall provide information on how to configure the environment to ensure that buffer overflow protection mechanisms in the environment(s) are invoked (e.g., compiler flags).
- The evaluator shall ensure that this documentation also includes an indication of whether such protections are on by default, or have to be specifically enabled.
- The evaluator shall ensure that the TSF is uniquely identified (with respect to other products from the TSF vendor), and that documentation provided by the developer in association with the requirements in the ST is associated with the TSF using this unique identification.

Evaluator Findings:

- The evaluator examined FPT_AEX_EXT.1 in the TSS section of the ST and determined that the Nubo application and libraries included in the TOE are compiled with flags that protect buffer overflow. The flags are automatically enabled by invoking a script, used to compile the Java application.
- The evaluator examined the documentation received by the vendor and determined that the TOE is Nubo's only product.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass.

6.4.3 ALC_TSU_EXT.1 Timely Security Updates

6.4.3.1 **ALC_TSU_EXT.1.1E**

Objective:

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

- The evaluator shall verify that the TSS contains a description of the timely security update process used by the developer to create and deploy security updates.
- The evaluator shall verify that this description addresses the entire application.
- The evaluator shall also verify that, in addition to the TOE developer's process, any third-party processes are also addressed in the description.
- The evaluator shall also verify that each mechanism for deployment of security updates is described.
- The evaluator shall verify that, for each deployment mechanism described for the update process, the TSS lists a time between public disclosure of a vulnerability and public availability of the security update to the TOE patching this vulnerability, to include any third-party or carrier delays in deployment.
- The evaluator shall verify that this time is expressed in a number or range of days.

- The evaluator shall verify that this description includes the publicly available mechanisms (including either an email address or website) for reporting security issues related to the TOE.
- The evaluator shall verify that the description of this mechanism includes a method for protecting the report either using a public key for encrypting email or a trusted channel for a website.

Evaluator Findings:

- The evaluator examined "**TOE Summary Specification**" for ALC_TSU_EXT.1 in the ST and found that the entries contain a description of how security updates are created and deployed and addresses the complete TOE.
- The evaluator examined the same section and noted that updates are available from a third-party, Google Play Store as the mechanism of deployment of security updates.
- The evaluator examined the same section and determined the procedure of submitting a potential security issue of the TOE.
- The evaluator examined the same section and determined the procedures and terms of public disclosure of a vulnerability and that the security update will be uploaded to Google Play within 14 days.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass

6.5 Tests (ATE)

6.5.1 ATE_IND.1 Independent Testing

6.5.1.1 ATE_IND.1.1E and ATE_IND.1.2E

Objective:

ATE_IND.1.1E - The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.1.2E - The evaluator shall test a subset of the TSF to confirm that the TSF operates as specified.

- The evaluator shall prepare a test plan and report documenting the testing aspects of the system, including any application crashes during testing.
- The evaluator shall determine the root cause of any application crashes and include that information in the report. The test plan covers all of the testing actions contained in the [CEM] and the body of this PP's evaluation activities.
- While it is not necessary to have one test case per test listed in an evaluation activity, the evaluator must document in the test plan that each applicable testing requirement in the ST is covered.
- The test plan identifies the platforms to be tested, and for those platforms not included in the test plan but included in the ST, the test plan provides a justification for not testing the platforms. This justification must address the differences between the tested platforms and the untested platforms, and make an argument that the differences do not affect the testing to be performed. It is not sufficient to merely assert that the differences have no effect; rationale must be provided. If all platforms claimed in the ST are tested, then no rationale is necessary.
- The test plan describes the composition of each platform to be tested, and any setup that is necessary beyond what is contained in the AGD documentation. It should be noted that the evaluator is expected to follow the AGD documentation for installation and setup of each platform either as part of a test or as a standard pre-test condition. This may include special test drivers or tools. For each driver or tool, an

argument (not just an assertion) should be provided that the driver or tool will not adversely affect the performance of the functionality by the TOE and its platform.

- This also includes the configuration of the cryptographic engine to be used. The cryptographic algorithms implemented by this engine are those specified by this PP and used by the cryptographic protocols being evaluated (e.g SSH). The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives. These procedures include expected results.
- The test report (which could just be an annotated version of the test plan) details the activities that took place when the test procedures were executed, and includes the actual results of the tests. This shall be a cumulative account, so if there was a test run that resulted in a failure; a fix installed; and then a successful re-run of the test, the report would show a “fail” and “pass” result (and the supporting details), and not just the “pass” result.

Evaluator Findings:

- An evaluator prepared a test plan that included for each SFR included in the ST, the tests required for that SFR as found in the PP and Functional Package. Specifically, the Test Plan included for each test:
    - The identity of the SFR and unique test number,
    - Test Assurance Activity – the test definition found in the PP or Functional Package,
    - Test Steps - the test steps required to complete the test,
    - Expected Test Results - a description of the expected test results,
    - Test Output – a place holder for screenshots and wireshark images from running the specific test.
    - Pass/Fail with Explanation – a place holder to report whether the test passed or failed and an explanation of the result.
- The evaluator then examined the “**Physical Boundary**” section in the ST and the following sections to determine what the TOE is and what is required in the TOE’s operational environment. The evaluator determined based on the SFRs and the “TOE Operational Environment” figure in the [ST], the connections that needed to be configured and tested and the tools that are required. The evaluator then created a detailed diagram of the “TOE Operational Environment” figure in the [ST] identifying the TOE, the operational environment, and the required test equipment. Each system in the diagram, the Test Bed, included an IP address.
- Additionally, the evaluator created a detailed table identifying the TOE, the OE, and the hardware/software required for testing.
- The evaluator then set up the Test Bed. The AGD was used to install and configure the TOE.
- Another evaluator then examined the completed Test Report that included for each SFR included in the ST, the tests required for that SFR as found in the PP and Functional Package. Specifically, the Test Plan included for each test:
    - A diagram and description of the Test Bed including all required test equipment and IP addresses.
    - The identity of the SFRs and unique test number,
    - Test Assurance Activity – the test definition found in the PP or Functional Package,
    - Test Steps - the test steps required to complete the test,
    - Expected Test Results - a description of the expected test results,
    - Test Output – the screenshots and wireshark images from running the specific test.

  - Pass/Fail with Explanation – a report on whether the test passed or failed and an explanation of the result.

- All tests were reported as passed and there were no reported system crashes.

Verdict: Pass

6.6 Vulnerability Assessment (AVA)

6.6.1 AVA_VAN.1 Vulnerability Survey

6.6.1.1 AVA_VAN.1.1E, AVA_VAN.1.2E, and AVA_VAN.1.3E

Objective:

AVA_VAN.1.1E - The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VAN.1.2E - The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VAN.1.3E - The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.

- The evaluator shall generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in ATE_IND, or a separate document. The evaluator performs a search of public information to find vulnerabilities that have been found in similar applications with a particular focus on network protocols the application uses and document formats it parses.
- The evaluator documents the sources consulted and the vulnerabilities found in the report.
- For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability, or the evaluator formulates a test (using the guidelines provided in ATE_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable, and an appropriate justification would be formulated.
- For Windows, Linux, macOS and Solaris: The evaluator shall also run a virus scanner with the most current virus definitions against the application files and verify that no files are flagged as malicious.

Evaluator Findings:

- The evaluator performed a *Vulnerability Assessment for Nubo Client Version 3.2* on December 7, 2023 and generated a vulnerability report to document their findings with respect to this requirement.
- Another evaluator examined the [VAN] and determined that the report included a list of public search sites and search strings. The search strings included the product; the product vendor; the application's name; the libraries packaged with the TOE, specifically the crypto library included in the TOE; the API's invoked; the operating system; and the hardware platform. The evaluator concluded that both lists were reasonable for an application running on an Android 12 device. The evaluator concluded that the two lists satisfied this work unit.
- The evaluator examined the [VAN] and concluded that for each vulnerability found, the report included a determination if the vulnerability applied to the TOE and if it did, the action that occurred to

remediate the vulnerability. If a vulnerability did not apply to the TOE, a rational of why the vulnerability did not apply to the TOE was included. The evaluator concluded that this work unit is satisfied.

- The TOE runs on Android operating system, therefore, running a virus scanner is not required.
- Based on these findings, this assurance activity is considered satisfied.

Verdict: Pass

7 Detailed Test Cases (Test Activities)

7.1 APP_V1.4

7.1.1 FCS_CKM.1/AK Test/CAVP 1

Item	Data
Test Assurance Activity	If the application "implements asymmetric key generation," then the following test activities shall be carried out. Evaluation Activity Note: The following tests may require the developer to provide access to a developer environment that provides the evaluator with tools that are typically available to endusers of the application. **Key Generation for FIPS PUB 186-4 RSA Schemes** The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key Generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent e, the private prime factors p and q, the public modulus n and the calculation of the private signature exponent d. Key Pair generation specifies 5 ways (or methods) to generate the primes p and q. These include: 1. Random Primes: o Provable primes o Probable primes 2. Primes with Conditions: o Primes p1, p2, q1,q2, p and q shall all be provable primes o Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes o Primes p1, p2, q1,q2, p and q shall all be probable primes To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), the public exponent of the RSA key, and the desired key length. For each key length supported, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. If possible, the Random Probable primes method should also be verified against a known good implementation as described above. Otherwise, the evaluator shall have the TSF generate 10 keys pairs for each supported key length nlen and verify: • $n = p \cdot q$, • p and q are probably prime according to Miller-Rabin tests, • GCD(p-1,e) = 1, • GCD(q-1,e) = 1, • $2^{16} \le e \le 2^{256}$ and e is an odd integer, • $\|p-q\| > 2^{nlen/2 - 100}$, • $p \ge 2^{nlen/2 -1/2}$, • $q \ge 2^{nlen/2 -1/2}$, • $2^{(nlen/2)} < d < LCM(p-1,q-1)$, • $e \cdot d = 1 \bmod LCM(p-1,q-1)$.

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation. FIPS 186-4 Public Key Verification (PKV) Test For each supported NIST curve, i.e., P-256, P384 and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

Key Generation for Finite-Field Cryptography (FFC)

The evaluator shall verify the implementation of the Parameters Generation and the Key Generation for FFC by the TOE using the Parameter Generation and Key Generation test. This test verifies the ability of the TSF to correctly produce values for the field prime p, the cryptographic prime q (dividing p-1), the cryptographic group generator g, and the calculation of the private key x and public key y. The Parameter generation specifies 2 ways (or methods) to generate the cryptographic prime q and the field prime p:
Cryptographic and Field Primes:

- Primes q and p shall both be provable primes
- Primes q and field prime p shall both be probable primes

and two ways to generate the cryptographic group generator g:
Cryptographic Group Generator:

- Generator g constructed through a verifiable process
- Generator g constructed through an unverifiable process.

The Key generation specifies 2 ways to generate the private key x:
Private Key:

- len(q) bit output of RBG where 1 ≤x ≤ q-1
- len(q) + 64 bit output of RBG, followed by a mod q-1 operation where 1≤ x≤q-1.

The security strength of the RBG must be at least that of the security offered by the FFC parameter set. To test the cryptographic and field prime generation method for the provable primes method and/or the group generator g for a verifiable process, the evaluator must seed the TSF parameter generation routine with sufficient data to deterministically generate the parameter set. For each key length supported, the evaluator shall have the TSF generate 25 parameter sets and key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification must also confirm

- g ≠ 0,1
- q divides p-1
- g^q mod p = 1
- g^x mod p = y

for each FFC parameter set and key pair.

Diffie-Hellman Group 14 and FFC Schemes using "safe-prime" groups

	Testing for FFC Schemes using Diffie-Hellman group 14 and/or safe-prime groups is done as part of testing in CKM.2.1.
Test Steps	**Key Generation for FIPS PUB 186-4 RSA Schemes** The evaluator examined the ST and found that in Section "Cryptographic Support" that the TOE was awarded the CAVP certificate #A1109 for FIPS186-4 RSA key generation using the key size 2048. This certificate provides assurance that the TSF performs these functions as required. **Key Generation for Elliptic Curve Cryptography (ECC)** The evaluator examined the ST and found that in Section "Cryptographic Support" that the TOE was awarded the CAVP certificate #A1109 for FIPS186-4 ECDSA key generation and key verification using the key sizeS P256 and P384. This certificate provides assurance that the TSF performs these functions as required. **Key Generation for Finite-Field Cryptography (FFC)** FFC tests are not applicable as the TOE does not use or claim FCC key generation. **Diffie-Hellman Group 14 and FFC Schemes using "safe-prime" groups** DH tests are not applicable as the TOE does not use or claim DH key generation.
Pass/Fail with Explanation	**Key Generation for FIPS PUB 186-4 RSA Schemes** Pass **Key Generation for Elliptic Curve Cryptography (ECC)** Pass **Key Generation for Finite-Field Cryptography (FFC)** N/A because FFC tests are not applicable as the TOE does not use or claim FCC key generation. **Diffie-Hellman Group 14 and FFC Schemes using "safe-prime" groups** N/A because DH tests are not applicable as the TOE does not use or claim DH key generation.

7.1.2 FCS_CKM.2 Test/CAVP 1

Item	Data
Test Assurance Activity	**Key Establishment Schemes** The evaluator shall verify the implementation of the key establishment schemes supported by the TOE using the applicable tests below. **SP800-56A Key Establishment Schemes** The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes using the following Function and Validity tests. These validation tests for each key agreement scheme verify that a TOE has implemented the components of the key agreement scheme according to the specifications in the Recommendation. These components include the calculation of the DLC primitives (the shared secret value Z) and the calculation of the derived keying material (DKM) via the Key Derivation Function (KDF). If key confirmation is supported, the evaluator shall also verify that the components of key confirmation have been implemented correctly, using the test procedures described below. This includes the parsing of the DKM, the generation of MACdata and the calculation of MACtag. **Function Test** The Function test verifies the ability of the TOE to implement the key agreement schemes correctly. To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each supported key agreement scheme-key agreement role combination, KDF type, and, if supported, key confirmation rolekey

confirmation type combination, the tester shall generate 10 sets of test vectors. The data set consists of one set of domain parameter values (FFC) or the NIST approved curve (ECC) per 10 sets of public keys. These keys are static, ephemeral or both depending on the scheme being tested.

The evaluator shall obtain the DKM, the corresponding TOE's public keys (static and/or ephemeral), the MAC tag(s), and any inputs used in the KDF, such as the Other Information (OtherInfo) and TOE id fields.

If the TOE does not use a KDF defined in SP 800-56A, the evaluator shall obtain only the public keys and the hashed value of the shared secret.

The evaluator shall verify the correctness of the TSF's implementation of a given scheme by using a known good implementation to calculate the shared secret value, derive the keying material DKM, and compare hashes or MAC tags generated from these values.

If key confirmation is supported, the TSF shall perform the above for each implemented approved MAC algorithm.

Validity Test

The Validity test verifies the ability of the TOE to recognize another party's valid and invalid key agreement results with or without key confirmation. To conduct this test, the evaluator shall obtain a list of the supporting cryptographic functions included in the SP800-56A key agreement implementation to determine which errors the TOE should be able to recognize.

The evaluator generates a set of 24 (FFC) or 30 (ECC) test vectors consisting of data sets including domain parameter values or NIST approved curves, the evaluator's public keys, the TOE's public/private key pairs, MACTag, and any inputs used in the KDF, such as the OtherInfo and TOE id fields.

The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the OtherInfo field, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

The TOE shall use these modified test vectors to emulate the key agreement scheme using the corresponding parameters. The evaluator shall compare the TOE's results with the results using a known good implementation verifying that the TOE detects these errors.

SP800-56B Key Establishment Schemes

The evaluator shall verify that the TSS describes whether the TOE acts as a sender, a recipient, or both for RSA-based key establishment schemes.

If the TOE acts as a sender, the following evaluation activity shall be performed to ensure the proper operation of every TOE supported combination of RSA-based key establishment scheme:

> To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each combination of supported key establishment scheme and its options (with or without key confirmation if supported, for each supported key confirmation MAC function if key confirmation is supported, and for each supported mask generation function if KTS-OAEP is supported), the tester shall generate 10 sets of test vectors. Each test vector shall include the RSA public key, the plaintext keying material, any additional input parameters if applicable, the MacKey and

	MacTag if key confirmation is incorporated, and the outputted ciphertext. For each test vector, the evaluator shall perform a key establishment encryption operation on the TOE with the same inputs (in cases where key confirmation is incorporated, the test shall use the MacKey from the test vector instead of the randomly generated MacKey used in normal operation) and ensure that the outputted ciphertext is equivalent to the ciphertext in the test vector. If the TOE acts as a receiver, the following evaluation activities shall be performed to ensure the proper operation of every TOE supported combination of RSA-based key establishment scheme: To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each combination of supported key establishment scheme and its options (with our without key confirmation if supported, for each supported key confirmation MAC function if key confirmation is supported, and for each supported mask generation function if KTS-OAEP is supported), the tester shall generate 10 sets of test vectors. Each test vector shall include the RSA private key, the plaintext keying material (KeyData), any additional input parameters if applicable, the MacTag in cases where key confirmation is incorporated, and the outputted ciphertext. For each test vector, the evaluator shall perform the key establishment decryption operation on the TOE and ensure that the outputted plaintext keying material (KeyData) is equivalent to the plaintext keying material in the test vector. In cases where key confirmation is incorporated, the evaluator shall perform the key confirmation steps and ensure that the outputted MacTag is equivalent to the MacTag in the test vector. The evaluator shall ensure that the TSS describes how the TOE handles decryption errors. In accordance with NIST Special Publication 800-56B, the TOE must not reveal the particular error that occurred, either through the contents of any outputted or logged error message or through timing variations. If KTS-OAEP is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.2.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each. If KTS-KEM-KWS is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.3.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each. **RSA-based key establishment** The evaluator shall verify the correctness of the TSF's implementation of RSAES-PKCS1-v1_5 by using a known good implementation for each protocol selected in FTP_DIT_EXT.1 that uses RSAES-PKCS1-v1_5. **Diffie-Hellman Group 14** The evaluator shall verify the correctness of the TSF's implementation of Diffie-Hellman group 14 by using a known good implementation for each protocol selected in FTP_DIT_EXT.1 that uses Diffie-Hellman group 14. **FFC Schemes using "safe-prime" groups** The evaluator shall verify the correctness of the TSF's implementation of safe-prime groups by using a known good implementation for each protocol selected in FTP_DIT_EXT.1 that uses safe-prime groups. This test must be performed for each safe-prime group that each protocol uses.
Test Steps	**SP800-56A Key Establishment Schemes**

	The evaluator examined the ST and found that in Section "Cryptographic Support" that the TOE was awarded the CAVP certificate #A1109 for Elliptic curve based key establishment (NIST SP 800-56A). This certificate provides assurance that the TSF performs these functions as required. **SP800-56B Key Establishment Schemes** SP800-56 tests are not applicable as the TOE does not use or claim SP800-56B key establishment schemes. **RSA-based key establishment** RSA-based tests are not applicable as the TOE does not use or claim RSAES-PKCS1-v1_5 key establishment schemes. **Diffie-Hellman Group 14** Diffie-Hellman tests are not applicable as the TOE does not use or claim Diffie-Helman Group 14 key establishment schemes. **FFC Schemes using "safe-prime" groups** FCC Schemes tests are not applicable as the TOE does not use or claim safe-prime groups key establishment schemes.
Pass/Fail with Explanation	**SP800-56A Key Establishment Schemes** The evaluator examined the ST and found that in Section "Cryptographic Support" that the TOE was awarded the CAVP certificate #A1109 for Elliptic curve based key establishment (NIST SP 800-56A) using P-256 and P384. This certificate provides assurance that the TSF performs these functions as required. **SP800-56B Key Establishment Schemes** N/A because the TOE does not use or claim SP800-56B key establishment schemes. **RSA-based key establishment** N/A because the TOE does not use or claim RSAES-PKCS1-v1_5 key establishment schemes. **Diffie-Hellman Group 14** N/A because the TOE does not use or claim Diffie-Helman Group 14 key establishment schemes. **FFC Schemes using "safe-prime" groups** N/A because the TOE does not use or claim safe-prime groups key establishment schemes.

7.1.3 FCS_COP.1/Hash Test/CAVP 1

Item	Data
Test Assurance Activity	The TSF hashing functions can be implemented in one of two modes. The first mode is the byteoriented mode. In this mode the TSF hashes only messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented testmacs. The evaluator shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP. The following tests require the developer to provide access to a test application that provides the evaluator with tools that are typically not found in the production application. • **Test 1**: Short Messages Test - Bit oriented Mode. The evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The

	length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. • **Test 2**: Short Messages Test - Byte oriented Mode. The evaluators devise an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. • **Test 3**: Selected Long Messages Test - Bit oriented Mode. The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 99*i, where 1 ≤ i ≤ m. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. • **Test 4**: Selected Long Messages Test - Byte oriented Mode. The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 8*99*i, where 1 ≤ i ≤ m/8. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. • **Test 5**: Pseudorandomly Generated Messages Test. This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.
Test Steps	The evaluator examined the ST and found that in Section "Cryptographic Support" that the TOE was awarded the CAVP certificate #A1109 for SHA2-256 (FIPS Pub 180-4) and SHA2-384 (FIPS Pub 180-4). This certificate provides assurance that the TSF performs these functions as required.
Pass/Fail with Explanation	Pass.

7.1.4 FCS_COP.1/KeyedHash Test/CAVP 1

Item	Data
Test Assurance Activity	For each of the supported parameter sets, the evaluator shall compose 15 sets of test data. Each set shall consist of a key and message data. The evaluator shall have the TSF generate HMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating HMAC tags with the same key and IV using a known-good implementation.
Test Steps	The evaluator examined the ST and found that in Section "Cryptographic Support" that the TOE was awarded the CAVP certificate #A1109 for HMAC-SHA2-256 (FIPS Pub 198-1) and HMAC-SHA2-384 (FIPS Pub 198-1). This certificate provides assurance that the TSF performs these functions as required.

Pass/Fail with Explanation	Pass.

7.1.5 FCS_COP.1/Sig Test/CAVP 1

Item	Data
Test Assurance Activity	The following tests require the developer to provide access to a test application that provides the evaluator with tools that are typically not found in the production application. **ECDSA Algorithm Tests** • **Test 1**: ECDSA FIPS 186-4 Signature Generation Test. For each supported NIST curve (i.e., P256, P-384 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation. • **Test 2**: ECDSA FIPS 186-4 Signature Verification Test. For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate a set of 10 1024- bit message, public key and signature tuples and modify one of the values (message, public key or signature) in five of the 10 tuples. The evaluator shall obtain in response a set of 10 PASS/FAIL values. **RSA Signature Algorithm Tests** • **Test 1**: Signature Generation Test. The evaluator shall verify the implementation of RSA Signature Generation by the TOE using the Signature Generation Test. To conduct this test the evaluator must generate or obtain 10 messages from a trusted reference implementation for each modulus size/SHA combination supported by the TSF. The evaluator shall have the TOE use their private key and modulus value to sign these messages. The evaluator shall verify the correctness of the TSF's signature using a known good implementation and the associated public keys to verify the signatures. • **Test 2**: Signature Verification Test. The evaluator shall perform the Signature Verification test to verify the ability of the TOE to recognize another party's valid and invalid signatures. The evaluator shall inject errors into the test vectors produced during the Signature Verification Test by introducing errors in some of the public keys, e, messages, IR format, and/or signatures. The TOE attempts to verify the signatures and returns success or failure.
Test Steps	**ECDSA Algorithm Tests** ECDSA tests are not applicable as the TOE does not use or claim ECDSA algorithm support. **RSA Signature Algorithm Tests** The evaluator examined the ST and found that in Section "Cryptographic Support" that the TOE was awarded the CAVP certificate #A1109 for RSA signature generation and signature verification (FIPS Pub 186-4) using 2048 bit RSA keys. This certificate provides assurance that the TSF performs these functions as required.
Pass/Fail with Explanation	**ECDSA Algorithm Tests** N/A because ECDSA is not claimed. **RSA Signature Algorithm Tests** Pass

7.1.6 FCS_COP.1/SKC Test/CAVP 1

Item	Data
Test Assurance Activity	The evaluator shall perform all of the following tests for each algorithm implemented by the TSF and used to satisfy the requirements of this PP: **AES-CBC Known Answer Tests** There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation. • **KAT-1**. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all-zeros key, and the other five shall be encrypted with a 256-bit all- zeros key. To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input and AES-CBC decryption. • **KAT-2**. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. Five of the keys shall be 128-bit keys, and the other five shall be 256-bit keys. To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption. • **KAT-3**. To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall have 256 256-bit keys. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit key/ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key. • **KAT-4**. To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from AES-CBC encryption of the given plaintext using a 128-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost 128-i bits be zeros, for i in [1,128]. To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input and AES-CBC decryption.

AES-CBC Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 < i <= 10. The evaluator shall choose a key, an IV and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key and IV. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation. The evaluator shall also test the decrypt functionality for each mode by decrypting an i-block message where 1 < i <=10. The evaluator shall choose a key, an IV and a ciphertext message of length i blocks and decrypt the message, using the mode to be tested, with the chosen key and IV. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key and IV using a known good implementation. AES-CBC Monte Carlo Tests The evaluator shall test the encrypt functionality using a set of 200 plaintext, IV, and key 3- tuples. 100 of these shall use 128 bit keys, and 100 shall use 256 bit keys. The plaintext and IV values shall be 128-bit blocks. For each 3-tuple, 1000 iterations shall be run as follows:

```
# Input: PT, IV, Key
for i = 1 to 1000:
    if i == 1:
        CT[1] = AES-CBC-Encrypt(Key, IV, PT)
        PT = IV
    else:
        CT[i] = AES-CBC-Encrypt(Key, PT)
        PT = CT[i-1]
```

The ciphertext computed in the 1000th iteration (i.e., CT[1000]) is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation.

The evaluator shall test the decrypt functionality using the same test as for encrypt, exchanging CT and PT and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

AES-GCM Monte Carlo Tests

The evaluator shall test the authenticated encrypt functionality of AES-GCM for each combination of the following input parameter lengths:

- 128 bit and 256 bit keys
- Two plaintext lengths. One of the plaintext lengths shall be a non-zero integer multiple of 128 bits, if supported. The other plaintext length shall not be an integer multiple of 128 bits, if supported.
- Three AAD lengths. One AAD length shall be 0, if supported. One AAD length shall be a non-zero integer multiple of 128 bits, if supported. One AAD length shall not be an integer multiple of 128 bits, if supported.
- Two IV lengths. If 96 bit IV is supported, 96 bits shall be one of the two IV lengths tested.

The evaluator shall test the encrypt functionality using a set of 10 key, plaintext, AAD, and IV tuples for each combination of parameter lengths above and obtain the ciphertext value and tag that results from AES-GCM authenticated encrypt. Each supported tag length shall be tested at least once per set of 10. The IV value may be supplied by the evaluator or the implementation being tested, as long as it is known.

The evaluator shall test the decrypt functionality using a set of 10 key, ciphertext, tag, AAD, and IV 5-tuples for each combination of parameter lengths above and obtain a Pass/Fail result on authentication and the decrypted plaintext if Pass. The set shall include five tuples that Pass and five that Fail.

The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

AES-XTS Tests

The evaluator shall test the encrypt functionality of XTS-AES for each combination of the following input parameter lengths:

256 bit (for AES-128) and 512 bit (for AES-256) keys

Three data unit (i.e., plaintext) lengths. One of the data unit lengths shall be a non-zero integer multiple of 128 bits, if supported. One of the data unit lengths shall be an integer multiple of 128 bits, if supported. The third data unit length shall be either the longest supported data unit length or 216 bits, whichever is smaller.

Using a set of 100 (key, plaintext and 128-bit random tweak value) 3-tuples and obtain the ciphertext that results from XTS-AES encrypt.

The evaluator may supply a data unit sequence number instead of the tweak value if the implementation supports it. The data unit sequence number is a base-10 number ranging between 0 and 255 that implementations convert to a tweak value internally.

The evaluator shall test the decrypt functionality of XTS-AES using the same test as for encrypt, replacing plaintext values with ciphertext values and XTS-AES encrypt with XTS-AES decrypt.

AES-CCM Tests

It is not recommended that evaluators use values obtained from static sources such as http://csrc.nist.gov/groups/STM/cavp/documents/mac/ccmtestvectors.zip or use values not generated expressly to exercise the AES-CCM implementation.
The evaluator shall test the generation-encryption and decryption-verification functionality of AES-CCM for the following input parameter and tag lengths:

- Keys: All supported and selected key sizes (e.g., 128, 256 bits).
- Associated Data: Two or three values for associated data length: The minimum (≥ 0 bytes) and maximum (≤ 32 bytes) supported associated data lengths, and 2^16 (65536) bytes, if supported.
- Payload: Two values for payload length: The minimum (≥ 0 bytes) and maximum (≤ 32 bytes) supported payload lengths.
- Nonces: All supported nonce lengths (7, 8, 9, 10, 11, 12, 13) in bytes.
- Tag: All supported tag lengths (4, 6, 8, 10, 12, 14, 16) in bytes.

The testing for CCM consists of five tests. To determine correctness in each of the below tests, the evaluator shall compare the ciphertext with the result of encryption of the same inputs with a known good implementation.

Variable Associated Data Test

For each supported key size and associated data length, and any supported payload length, nonce length, and tag length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Variable Payload Test

For each supported key size and payload length, and any supported associated data length, nonce length, and tag length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Variable Nonce Test

For each supported key size and nonce length, and any supported associated data length, payload length, and tag length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Variable Tag Test

For each supported key size and tag length, and any supported associated data length, payload length, and nonce length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Decryption-Verification Process Test

To test the decryption-verification functionality of AESCCM, for each combination of supported associated data length, payload length, nonce length, and tag length, the evaluator shall supply a key value and 15 sets of input plus ciphertext, and obtain the decrypted payload. Ten of the 15 input sets supplied should fail verification and five should pass.

AES-CTR Tests

Test 1: Known Answer Tests (KATs)

There are four Known Answer Tests (KATs) described below. For all KATs, the plaintext, IV, and ciphertext values shall be 128-bit blocks. The results from each test may either be obtained by the validator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

To test the encrypt functionality, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all zeros key, and the other five shall be encrypted with a 256-bit all zeros key. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input.

To test the encrypt functionality, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from encryption of an all zeros plaintext using the given key value and an IV of all zeros. Five of the key values shall be 128-bit keys, and the other five shall be 256-bit keys. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using an all zero ciphertext value as input.

To test the encrypt functionality, the evaluator shall supply the two sets of key values described below and obtain the ciphertext values that result from AES encryption of an all zeros plaintext using the given key values an an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second shall have 256 256-bit keys. Key_i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1, N]. To test the decrypt functionality, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from decryption of the given ciphertext using the given key values and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit pairs. Key_i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros for i in [1, N]. The ciphertext value in each pair shall be the value that results in an all zeros plaintext when decrypted with its corresponding key.

	To test the encrypt functionality, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from encryption of the given plaintext using a 128-bit key value of all zeros and using a 256 bit key value of all zeros, respectively, and an IV of all zeros. Plaintext value i in each set shall have the leftmost bits be ones and the rightmost 128-i bits be zeros, for i in [1, 128]. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input. **Test 2: Multi-Block Message Test** The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 lessthan i less-than-or-equal to 10. For each i the evaluator shall choose a key, IV, and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation. The evaluator shall also test the decrypt functionality by decrypting an i-block message where 1 less-than i less-than-or-equal to 10. For each i the evaluator shall choose a key and a ciphertext message of length i blocks and decrypt the message, using the mode to be tested, with the chosen key. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key using a known good implementation. **Test 3: Monte-Carlo Test** For AES-CTR mode perform the Monte Carlo Test for ECB Mode on the encryption engine of the counter mode implementation. There is no need to test the decryption engine. The evaluator shall test the encrypt functionality using 200 plaintext/key pairs. 100 of these shall use 128 bit keys, and 100 of these shall use 256 bit keys. The plaintext values shall be 128-bit blocks. For each pair, 1000 iterations shall be run as follows: For AES-ECB mode # Input: PT, Key for i = 1 to 1000: CT[i] = AES-ECB-Encrypt(Key, PT) PT = CT[i] The ciphertext computed in the 1000th iteration is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation.
Test Steps	**AES-CBC Known Answer Tests** This is not applicable as the TOE does not claim or use AES in CBC mode. **AES-CBC Multi-Block Message Test** This is not applicable as the TOE does not claim or use AES in CBC mode. **AES-GCM Monte Carlo Tests** The evaluator examined the ST and found that in Section “Cryptographic Support” that the TOE was awarded the CAVP certificate #A1109 for AES-GCM (NIST SP800-38D) using key size 256 for encryption and decryption. This certificate provides assurance that the TSF performs these functions as required. **AES-XTS Tests** This is not applicable as the TOE does not claim or use AES in XTS mode. **AES-CCM Tests** This is not applicable as the TOE does not claim or use AES in CCM mode. **AES-CTR Tests** This is not applicable as the TOE does not claim to use AES in CTR mode.

Pass/Fail with Explanation	**AES-CBC Known Answer Tests** N/A because the TOE does not claim AES in CBC mode. **AES-CBC Multi-Block Message Test** N/A because the TOE does not claim AES in CBC mode. **AES-GCM Monte Carlo Tests** Pass **AES-XTS Tests** N/A because the TOE does not claim AES in XTS mode. **AES-CCM Tests** N/A because the TOE does not claim AES in CCM mode. **AES-CTR Tests** N/A because the TOE does not claim AES in CTR mode.

7.1.7 FCS_RBG_EXT.2.1 Test/CAVP 1

Item	Data
Test Assurance Activity	The evaluator shall perform the following tests, depending on the standard to which the RBG conforms. **Implementations Conforming to FIPS 140-2 Annex C** The reference for the tests contained in this section is The Random Number Generator Validation System (RNGVS). The evaluators shall conduct the following two tests. Note that the "expected values" are produced by a reference implementation of the algorithm that is known to be correct. Proof of correctness is left to each Scheme. • **Test 1**: The evaluators shall perform a Variable Seed Test. The evaluators shall provide a set of 128 (Seed, DT) pairs to the TSF RBG function, each 128 bits. The evaluators shall also provide a key (of the length appropriate to the AES algorithm) that is constant for all 128 (Seed, DT) pairs. The DT value is incremented by 1 for each set. The seed values shall have no repeats within the set. The evaluators ensure that the values returned by the TSF match the expected values. • **Test 2**: The evaluators shall perform a Monte Carlo Test. For this test, they supply an initial Seed and DT value to the TSF RBG function; each of these is 128 bits. The evaluators shall also provide a key (of the length appropriate to the AES algorithm) that is constant throughout the test. The evaluators then invoke the TSF RBG 10,000 times, with the DT value being incremented by 1 on each iteration, and the new seed for the subsequent iteration produced as specified in NIST-Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4 Using the 3-Key Triple DES and AES Algorithms, Section E.3. The evaluators ensure that the 10,000th value produced matches the expected value. **Implementations Conforming to NIST Special Publication 800-90A** • **Test 1**: The evaluator shall perform 15 trials for the RNG implementation. If the RNG is configurable, the evaluator shall perform 15 trials for each configuration. The evaluator shall also confirm that the operational guidance contains appropriate instructions for configuring the RNG functionality. If the RNG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) generate a second block of random bits

	(4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. “generate one block of random bits” means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP 800-90A). If the RNG does not have prediction resistance, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) reseed, (4) generate a second block of random bits (5) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The fifth value is additional input to the first call to generate. The sixth and seventh are additional input and entropy input to the call to reseed. The final value is additional input to the second generate call. The following paragraphs contain more information on some of the input values to be generated/selected by the evaluator. **Entropy input**: the length of the entropy input value must equal the seed length. **Nonce**: If a nonce is supported (CTR_DRBG with no Derivation Function does not use a nonce), the nonce bit length is one-half the seed length. **Personalization string**: The length of the personalization string must be less then or equal to seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is support, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied. **Additional input**: the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.
Test Steps	**Implementations Conforming to FIPS 140-2 Annex C** This test is not applicable because the TOE does not claim conformance to FIPS 140-2 Annex C. **Implementations Conforming to NIST Special Publication 800-90A** The evaluator examined the ST and found that in Section “Cryptographic Support” that the TOE was awarded the CAVP certificate #A1109 for Counter DRBG (NISP SP 800-90A) (AES-256). This certificate provides assurance that the TSF performs these functions as required.
Pass/Fail with Explanation	**Implementations Conforming to FIPS 140-2 Annex C** N/A because the TOE does not claim conformance to FIPS 140-2 Annex C. **Implementations Conforming to NIST Special Publication 800-90A** Pass

7.1.8 FCS_RBG_EXT.2.2/Client Test #1

Item	Data

Test Assurance Activity	In the future, specific statistical testing (in line with NIST SP 800-90B) will be required to verify the entropy estimates.
Test Steps	N/A
Pass/Fail with Explanation	N/A

7.1.9 FCS_HTTPS_EXT.1.2/Client Test #1

Item	**Data**
Test Assurance Activity	Other tests are performed in conjunction with the TLS package.
Test Steps	This test was performed in accordance with testing performed for FIA_X509_EXT.1
Pass/Fail with Explanation	Pass. This test was performed in accordance with testing performed for FIA_X509_EXT.1

7.1.10 FCS_HTTPS_EXT.1.3/Client Test #1

Item	**Data**
Test Assurance Activity	Certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1, and the evaluator shall perform the following test: The evaluator shall demonstrate that using a certificate without a valid certification path results in the selected action in the SFR. If "**notify the user**" is selected in the SFR, then the evaluator shall also determine that the user is notified of the certificate validation failure. Using the administrative guidance, the evaluator shall then load a certificate or certificates to the Trust Anchor Database needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that again, using a certificate without a valid certification path results in the selected action in the SFR, and if "notify the user" was selected in the SFR, the user is notified of the validation failure.
Test Steps	This test was performed in accordance with testing performed for FIA_X509_EXT.1
Expected Test Results	This test was performed in accordance with testing performed for FIA_X509_EXT.1
Test Output	This test was performed in accordance with testing performed for FIA_X509_EXT.1
Pass/Fail with Explanation	Pass, performed in accordance with testing performed for FIA_X509_EXT.1

7.1.11 FCS_STO_EXT.1.1 Test #2

Item	**Data**

Test Assurance Activity	The evaluator shall verify that the application uses the Android KeyStore or the Android KeyChain to store certificates.
Test Steps	• The evaluator shall go through the Nubo's source code and verify that the application uses Android KeyStore or the Android KeyChain. • The evaluator shall verify that the application uses Android KeyStore or the Android KeyChain for securely storing user credentials.
Pass/Fail with Explanation	Pass. the evaluator confirmed that the TOE uses AndroidKeyStore.

7.1.12 FDP_DAR_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	Evaluation activities (after the identification of the sensitive data) are to be performed on all sensitive data listed that are not covered by FCS_STO_EXT.1. If "implement functionality to encrypt sensitive data as defined in the PP-Module for File Encryption" or "protect sensitive data in accordance with FCS_STO_EXT.1" is selected, the evaluator shall inventory the filesystem locations where the application may write data. The evaluator shall run the application and attempt to store sensitive data. The evaluator shall then inspect those areas of the filesystem to note where data was stored (if any), and determine whether it has been encrypted. TD0756 has been applied.
Test Steps	• The evaluator shall run the TOE and attempts to create sensitive information. • The evaluator then traverses to the location where the information is stored. • The evaluator verifies that the data stored is encrypted.
Pass/Fail with Explanation	Pass. The evaluator verified that the credentials that contains sensitive information is being stored in encrypted format.

7.1.13 FDP_DAR_EXT.1.1 Test #2

Test Assurance Activity	The evaluator shall inspect the TSS and verify that it describes how files containing sensitive data are stored with the MODE_PRIVATE flag set.
Test Steps	• Review the TSS section.
Pass/Fail with Explanation	Pass. TSS shows that sensitive data are stored with MODE_PRIVATE flag set.

7.1.14 FDP_DEC_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	**For Android**: The evaluator shall verify that each uses-permission entry in the AndroidManifest.xml file for access to a hardware resource is reflected in the selection.
Test Steps	• The evaluator shall traverse to the file location of AndroidManifest.xml.

	- The evaluator shall verify that user-permission entry is mentioned in the file mentioned in the above step to require access to the hardware.
Pass/Fail with Explanation	Pass. The AndroidManifest.xml file of the TOE has uses-permission entry to access hardware of the device.

7.1.15 FDP_DEC_EXT.1.2 Test #1

Item	Data
Test Assurance Activity	**For Android:** The evaluator shall verify that each <uses-permission>entry in the AndroidManifest.xml file for access to a sensitive information repository is reflected in the selection.
Test Steps	- The evaluator shall traverse to the file location of AndroidManifest.xml. - The evaluator shall verify that AndroidManifest file shows no access to sensitive information repository.
Pass/Fail with Explanation	Pass. The AndroidManifest.xml file of the TOE shows no permission to access sensitive information repository.

7.1.16 FDP_NET_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall run the application. While the application is running, the evaluator shall sniff network traffic ignoring all non-application associated traffic and verify that any network communications witnessed are documented in the TSS or are user-initiated.
Test Steps	- The evaluator shall run the application and enter user credentials. - The evaluator shall verify with packet capture successful tls handshake negotiation. This shows user authentication with the management server as documented in the TSS.
Pass/Fail with Explanation	Pass. There is successful user authentication with management server which is documented in the TSS. This meets testing requirement.

7.1.17 FDP_NET_EXT.1.1 Test #2

Test Assurance Activity	The evaluator shall run the application. After the application initializes, the evaluator shall run network port scans to verify that any ports opened by the application have been captured in the ST for the third selection and its assignment. This includes connection-based protocols (e.g. TCP, DCCP) as well as connectionless protocols (e.g. UDP).
Test Steps	N/A. The ST does not claim the third selection of the SFR.
Pass/Fail with Explanation	N/A. The ST does not claim the third selection of the SFR.

7.1.18 FIA_X509_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function failing, for each of the following reasons, in turn:

	• by establishing a certificate path in which one of the issuing certificates is not a CA certificate, • by omitting the basicConstraints field in one of the issuing certificates, • by setting the basicConstraints field in an issuing certificate to have CA=False, • by omitting the CA signing bit of the key usage field in an issuing certificate, and • by setting the path length field of a valid CA field to a value strictly less than the certificate path. The evaluator shall then establish a valid certificate path consisting of valid CA certificates, and demonstrate that the function succeeds. The evaluator shall then remove trust in one of the CA certificates, and show that the function fails.
Test Steps	Establish a certificate path in which one of the issuing certificates is not a CA certificate: • By omitting the basicConstraints field in one of the issuing certificates. ▪ Configure the ICA certificate lacking the basicConstraints extension. ▪ Load the certificate lacking the basicConstraints extension on the TLS server. ▪ Sign the certificate using ICA lacking basic constraints. ▪ Attempt the connection from the TOE to the TLS server and verify the connection is unsuccessful. ▪ Verify with packet capture the unsuccessful connection between TOE and the server. • By setting the basicConstraints field in an issuing certificate to have CA=False. ▪ Configure the ICA certificate with the flag in the basicConstraints extension set to False. ▪ Load the certificate showing the basicConstraints flag set to False on the TLS server. ▪ Sign the certificate using ICA with the flag in the basicConstraints extension set to FALSE. ▪ Attempt the connection from the TOE to the TLS server and verify the connection is unsuccessful. ▪ Verify with packet capture the unsuccessful connection between TOE and the server. • By omitting the CA signing bit of the key usage field in an issuing certificate. ▪ Configure the ICA certificate lacking the CA signing bit in the Key usage field. ▪ Load the certificate lacking the ICA signing bit on the TLS server. ▪ Sign the certificate using ICA with no certificate sign key usage. ▪ Attempt the connection from the TOE to the TLS server and verify the connection is unsuccessful. ▪ Verify with packet capture the unsuccessful connection between TOE and the server. • By setting the path length field of a valid CA field to a value strictly less than the certificate path. ▪ Configure the root CA certificate with the Path length of 1. ▪ Configure the Intermediate ICA1 certificate with the Path length of 0. ▪ Configure the Intermediate ICA2 certificate with the Path length of 0.

	▪ Load the certificate chain having ICA certificate set pathlength of 0 to TLS server. ▪ Sign the node certificate with ICA2. ▪ Attempt the connection from the TOE to the TLS Server and verify the connection is unsuccessful. ▪ Verify with packet capture the unsuccessful connection between TOE and the server. • Valid certificate chain ▪ Create a full chain of certificates to connect to the TOE. ▪ Load a complete certificate validation chain to the TLS server. ▪ Attempt the connection from the TOE to the TLS server and verify the connection is successful. ▪ Verify with packet capture the successful connection between TOE and the server. • Invalid certificate chain ▪ Load the chain of certificate on server without IDC2. ▪ Attempt the connection from the TOE to the TLS server and verify the connection is unsuccessful. ▪ Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The evaluator verified that TOE makes the connection with the server for a valid certificate chain.

7.1.19 FIA_X509_EXT.1.1 Test #2

Test Assurance Activity	The evaluator shall demonstrate that validating an expired certificate results in the function failing.
Test Steps	• The evaluator used the XCA tool to create an expired certificate that expired on April 19, 2023 10:54:00 AM EST. • The evaluator uploaded both the ICA1 and ICA2 to the management server. • The evaluator ensured that the server certificate expired on April 19, 2023 10:54:00 AM EST and uploaded the expired certificate to the management server. • The evaluator checked the current date and time on the management server and ensured that the certificate expired as per the current time. • The evaluator checked the current date and time on the TOE Platform and ensured that the certificate expired as per the current time. • The evaluator configured the management server to leverage the uploaded expired server certificate and the server key for the TLS handshake with the client. • The evaluator attempted a connection from the TOE (TLS client) to the management server and confirmed that the client could not connect to the server as the server returned a certificate_unknown error to the client. • The evaluator observed the packet capture on the server and confirmed that the TLS handshake was not successful.
Pass/Fail with Explanation	Pass. The TOE does not validate an expired certificate and the TLS connection failed.

7.1.20 FIA_X509_EXT.1.1 Test #3

Test Assurance Activity	The evaluator shall test that the TOE can properly handle revoked certificates – conditional on whether CRL, OCSP, or OCSP Stapling or OCSP Multi-stapling is selected; if multiple methods are selected, then the following tests shall be performed for each method: • The evaluator shall test revocation of the node certificate. • The evaluator shall also test revocation of an intermediate CA certificate (i.e. the intermediate CA certificate should be revoked by the root CA), if intermediate CA certificates are supported. If OCSP stapling per RFC 6066 is the only supported revocation method, this test is omitted. • The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails.
Test Steps	CRL Valid certificate: • Create certificates with CRL Extended Key Usage. • Attempt connection with valid certificate and verify the connection is successful. • Verify with packet capture the successful connection between TOE and the server. Invalid server certificate. • Revoke the server certificate. • Attempt a connection using the revoked certificate and verify the connection is unsuccessful. • Verify with packet capture the unsuccessful connection between TOE and the server. Invalid ICA certificate. • Reset the certificate chain and revoke only the intermediate ICA2 certificate • Attempt a connection using the same certificate and verify the connection is unsuccessful. • Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE is able to establish a connection for a valid CRL certificate and able to reject revoked CRL certificates.

7.1.21 FIA_X509_EXT.1.1 Test #4 - CRL

Test Assurance Activity	If CRL is selected, the evaluator shall likewise configure the CA to sign a CRL with a certificate that does not have the CRLsign key usage bit set. The evaluator shall verify that validation of the CRL fails and that the TOE treats the certificate being checked as invalid and rejects the connection. **TD0780 has been applied.**
Test Steps	• Configure ICA certificate lacking the CRL signing Key usage. • Attempt the connection from TOE to the TLS server and verify the connection is unsuccessful.

	- Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE is able to reject a connection for a certificate lacking in both CRL signing bit.

7.1.22 FIA_X509_EXT.1.1 Test #5

Test Assurance Activity	The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)
Test Steps	- Run the acumen-tlsc tool that will modify the first 8 bytes of the certificate and verify that the connection fails. - Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE rejects the connection when the first 8 bytes of the certificate are modified.

7.1.23 FIA_X509_EXT.1.1 Test #6

Test Assurance Activity	The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)
Test Steps	- Run the acumen-tlsc tool that that will modify the last byte of the certificate and verify that the connection fails. - Verify with packet capture the unsuccessful connection between TOE and the server. ○ Server sent modified certificate to TOE. ○ TOE rejects the connection with the server.
Pass/Fail with Explanation	Pass. The TOE rejects the connection when the last byte of the certificate is modified.

7.1.24 FIA_X509_EXT.1.1 Test #7

Test Assurance Activity	The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)
Test Steps	- Run the acumen-tlsc tool that will modify the public key of the certificate and verify that the connection fails. - Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE rejects the connection when the public key of the certificate is modified.

7.1.25 FIA_X509_EXT.1.2 Test #1

Test Assurance Activity	The evaluator shall ensure that the certificate of at least one of the CAs in the chain does not contain the basicConstraints extension. The evaluator shall confirm that validation of the certificate path fails:

	(i) as part of the validation of the peer certificate belonging to this chain; and/or when attempting to add the CA certificate without the basicConstraints extension to the TOE's trust store. (ii) when attempting to add the CA certificate without the basicConstraints extension to the TOE's trust store.
Test Steps	• Configure the ICA certificate lacking the basicConstraints extension. • Load the certificate lacking the basicConstraints extension on the TLS server. • Sign the certificate using ICA lacking basic constraints. • Attempt the connection from the TOE to the TLS server and verify the connection is unsuccessful. • Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE is able to reject a connection for a certificate that lacks a basic constraint.

7.1.26 FIA_X509_EXT.1.2 Test #2

Test Assurance Activity	The evaluator shall ensure that the certificate of at least one of the CAs in the chain has the CA flag in the basicConstraints extension not set (or set to FALSE). The evaluator shall confirm that validation of the certificate path fails (i) as part of the validation of the peer certificate belonging to this chain; and/or when attempting to add the CA certificate with the CA flag not set (or set to FALSE) in the basicConstraints extension to the TOE's trust store.
	The evaluator shall ensure that the certificate of at least one of the CAs in the chain has the CA flag in the basicConstraints extension not set (or set to FALSE). The evaluator shall confirm that validation of the certificate path fails (i) as part of the validation of the peer certificate belonging to this chain; and/or (ii) when attempting to add the CA certificate with the CA flag not set (or set to FALSE) in the basicConstraints extension to the TOE's trust store
Test Steps	• Configure the ICA certificate with the flag in the basicConstraints extension set to FALSE. • Load the certificate showing the basicConstraints flag set to FALSE on the TLS server. • Sign the certificate using ICA with the flag in the basicConstraints extension set to FALSE. • Attempt the connection from the TOE to the TLS server and verify the connection is unsuccessful. • Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE is able to reject connection to a certificate with the basic field extension field set to false.

7.1.27 FIA_X509_EXT.2 Test #1

Item	**Data**

Test Assurance Activity	The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the operational guidance to determine that all supported administrator-configurable options behave in their documented manner.
Test Steps	• The evaluator shall create chain of certificates with CRL Extended Key Usage. • The evaluator shall attempt connection with valid certificate and verify that connection succeeds. • The evaluator shall verify the successful connection via packet capture. • The evaluator shall delete the NodeCert.pem crl and replace it with a different crl(ICA2.pem) on the server. • The evaluator shall attempt to make a connection and verify the connection failed **.** • The evaluator shall verify the connection failure via packet capture.
Pass/Fail with Explanation	Pass. The TOE makes a successful connection with the client when certificate validity is confirmed and denies connection when the revocation status of the client certificate cannot be verified. This meets testing requirements.

7.1.28 FIA_X509_EXT.2 Test #2

Test Assurance Activity	The evaluator shall demonstrate that an invalid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity cannot be accepted.
Test Steps	Invalid certificate testing is performed in FIA_X509_EXT.1.1 Test1, FIA_X509_EXT.1.1 Test2 , FIA_X509_EXT.1.1 Test3, Where Certificate with Invalid path, Expired certificate, Revoked certificates resulted in connection failure.
Pass/Fail with Explanation	Pass. Invalid certificate testing is performed in FIA_X509_EXT.1.1 Test1, FIA_X509_EXT.1.1 Test2 , FIA_X509_EXT.1.1 Test3. This meets testing requirements.

7.1.29 FMT_CFG_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	If the application uses any default credentials the evaluator shall run the following tests. **Test 1:** The evaluator shall install and run the application without generating or loading new credentials and verify that only the minimal application functionality required to set new credentials is available.
Test Steps	N/A because the TOE does not use any default credentials.

Pass/Fail with Explanation	N/A because the TOE does not use any default credentials.

7.1.30 FMT_CFG_EXT.1.1 Test #2

Item	Data
Test Assurance Activity	If the application uses any default credentials the evaluator shall run the following tests. **Test 2:** The evaluator shall attempt to clear all credentials and verify that only the minimal application functionality required to set new credentials is available.
Test Steps	N/A because the TOE does not use any default credentials.
Pass/Fail with Explanation	N/A because the TOE does not use any default credentials.

7.1.31 FMT_CFG_EXT.1.1 Test #3

Item	Data
Test Assurance Activity	If the application uses any default credentials the evaluator shall run the following tests. **Test 3:** The evaluator shall run the application, establish new credentials and verify that the original default credentials no longer provide access to the application.
Test Steps	N/A because the TOE does not use any default credentials.
Pass/Fail with Explanation	N/A because the TOE does not use any default credentials.

7.1.32 FMT_CFG_EXT.1.2 Test #1

Test Assurance Activity	The evaluator shall install and run the application. The evaluator shall inspect the filesystem of the platform (to the extent possible) for any files created by the application and ensure that their permissions are adequate to protect them. **Platform: Android..** The evaluator shall run the command find -L . -perm /002 inside the application's data directories to ensure that all files are not world-writable. The command should not print any files.
Test Steps	• The evaluator shall install and run the TOE. • Then traverse to the data directories location and issue the command find -L . -perm /002. • Finally, the evaluator verifies that the command does not print any files.
Pass/Fail with Explanation	Pass. The TOE does not provide any output when command “find -L -perm /002” is executed.

7.1.33 FMT_MEC_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall inspect the TSS and verify that it describes what Android API is used (and provides a link to the documentation of the API) when storing configuration data. The evaluator shall run the application and run the application and verify that the behavior of the TOE is consistent with where and how the API documentation says the configuration data will be stored. **TD0747 has been addressed.**
Test Steps	• The evaluator shall check documentation on what API is used for storing configuration data. • The evaluator shall access the TOE. • Then proceed to make changes to the configuration data on the TOE (Management url/IP). • Finally, the evaluator shall verify that the change is consistent with the data stored at location /data/data/package/shared_prefs/. Note: the evaluated configuration supports only the default Nubo Management URL which is hard coded in the TOE and therefore, does not support modifying the URL. However, this test proves the URL, which is configuration data, is stored in /data/data/package/shared_prefs. • The evaluator shall access the TOE. • The evaluator shall deactivate the phone. • The evaluator shall attempt to activate the phone and doing so makes changes to the configuration data on the TOE (User creation). • Finally, the evaluator shall verify that the change is consistent with the data stored at location /data/data/package/shared_prefs/.
Pass/Fail with Explanation	Pass. The evaluator verified that the changes that was made is reflected in the shared preference location which is consistent with the API documentation.

7.1.34 FMT_MEC_EXT.1.1 Test #2

Item	Data
Test Assurance Activity	If "**implement functionality to encrypt and store configuration options as defined by FDP_PRT_EXT.1 in the PP-Module for File Encryption**" is selected, for all configuration options listed in the TSS as being stored and protected using encryption, the evaluator shall examine the contents of the configuration option storage (identified in the TSS) to determine that the options have been encrypted.
Test Steps	N/A because FDP_PRT_EXT.1 is not claimed.
Expected Test Results	N/A because FDP_PRT_EXT.1 is not claimed.
Test Output	N/A because FDP_PRT_EXT.1 is not claimed.
Pass/Fail with Explanation	N/A because FDP_PRT_EXT.1 is not claimed.

7.1.35 FMT_SMF.1.1 Test #1

Test Assurance Activity	The evaluator shall test the application's ability to provide the management functions by configuring the application and testing each option selected from above. The evaluator is expected to test these functions in all the ways in which the ST and guidance documentation state the configuration can be managed.
Test Steps	The TOE implements one management function: upgrading the TOE. Upgrade TOE: • The evaluator shall check the current version of the TOE. • The evaluator shall download a new version of the TOE. • The evaluator shall upgrade the TOE to a new version. The evaluator shall verify that upgrade is successful.
Pass/Fail with Explanation	Pass. The evaluator was able to verify that TOE was able to perform the test for the management function specified in ST.

7.1.36 FPR_ANO.1.1 Test #1

Test Assurance Activity	If require user approval before executing is selected, the evaluator shall run the application and exercise the functionality responsibly for transmitting PII and verify that user approval is required before transmission of the PII.
Test Steps	• The evaluator shall access the TOE and try sending PII data to the management server. • The evaluator shall verify that user approval is required prior to the transfer of the PII.
Pass/Fail with Explanation	Pass. The TOE provides a message with transferring PII details.

7.1.37 FPT_AEX_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall perform either a static or dynamic analysis to determine that no memory mappings are placed at an explicit and consistent address except for any exceptions claimed in the SFR. For these exceptions, the evaluator shall verify that this analysis shows explicit mappings that are consistent with what is claimed in the TSS. The method of doing so varies per platform. For those platforms requiring the same application running on two different systems, the evaluator may alternatively use the same device. After collecting the first instance of mappings, the evaluator must uninstall the application, reboot the device, and reinstall the application to collect the second instance of mappings. **Platforms:Android...** The evaluator shall run the same application on two different Android systems. Both devices do not need to be evaluated, as the second device is acting only as a tool. Connect via ADB and inspect /proc/PID/maps. Ensure the two different instances share no memory mappings made by the application at the same location. Applied TD0798
Test Steps	• The evaluator runs the application on two different android machines. • The evaluator then verifies the directory /proc/PID/maps and confirms that the entry is not same in both the instances.

Pass/Fail with Explanation	Pass. The TOE memory maps differ when run on different devices.

7.1.38 FPT_AEX_EXT.1.2 Test #1

Test Assurance Activity	The evaluator shall verify that no memory mapping requests are made with write and execute permissions. **Platforms:Android…** The evaluator shall perform static analysis on the application to verify that mmap is never invoked with both the PROT_WRITE and PROT_EXEC permissions, and mprotect is never invoked.
Test Steps	• The evaluator shall decompile the TOE apk file using apktool. • The evaluator shall run grep command on the apk file and verify if PROT_WRITE and PROT_EXEC permissions are invoked with mmap. • The evalutor shall run grep command on the apk file and verify if mprotect is invoked.
Pass/Fail with Explanation	Pass. The TOE has never invoked 'mprotect' and not invoked 'mmap' with both the PROT_WRITE and PROT_EXEC permissions.

7.1.39 FPT_AEX_EXT.1.4 Test #1

Test Assurance Activity	The evaluator shall run the application and determine where it writes its files. For files where the user does not choose the destination, the evaluator shall check whether the destination directory contains executable files.
Test Steps	• The evaluator shall run the TOE and run operation on it. • The evaluator shall check where the user data is stored. • The evaluator shall check the /data/data/package/ directory to see no executable files are stored.
Pass/Fail with Explanation	Pass. No executable files are stored in the application package location.

7.1.40 FPT_API_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall then compare the list with the supported APIs (available through e.g. developer accounts, platform developer groups) and ensure that all APIs listed in the TSS are supported.
Test Steps	• The evaluator reviewed the TSS listed the Platform APIs in the application. • The evaluator then compared the list with the supported APIs available through Developer groups and ensured that all the APIs listed in the TSS are supported.
Pass/Fail with Explanation	Pass. The TOE API listed in ST must match the API's in the android developer site.

7.1.41 FPT_IDV_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall install the application, then check for the existence of version information. If SWID tags is selected the evaluator shall check for a .swidtag file. The evaluator shall open the file and verify that is contains at least a SoftwareIdentity element and an Entity element.
Test Steps	• The evaluator installs the application file (apk). • The evaluator verifies the version of the application in the about section of the application.
Pass/Fail with Explanation	Pass. The evaluator verified the version mentioned in the application.

7.1.42 FPT_LIB_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall install the application and survey its installation directory for dynamic libraries. The evaluator shall verify that libraries found to be packaged with or employed by the application are limited to those in the assignment.
Test Steps	• The evaluator installed the TOE application. • The evaluator verified that the library file is only packaged with the ones mentioned in ST.
Pass/Fail with Explanation	Pass. The TOE only uses the library files that are mentioned in the ST.

7.1.43 FPT_TUD_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall check for an update using procedures described in either the application documentation or the platform documentation and verify that the application does not issue an error. If it is updated or if it reports that no update is available this requirement is considered to be met.
Test Steps	• The evaluator shall check Google Play Store for an update of the TOE. • The evaluator shall verify if TOE has the current update.
Pass/Fail with Explanation	Pass. The TOE is updated to the current version.

7.1.44 FPT_TUD_EXT.1.2 Test #1

Test Assurance Activity	The evaluator shall query the application for the current version of the software according to the operational user guidance. The evaluator shall then verify that the current version matches that of the documented and installed version.
Test Steps	• The evaluator shall run the TOE application. • The evaluator shall query to determine the correct version by issuing command and comparing to the ones mentioned in the ST.
Pass/Fail with Explanation	Pass. The current version of TOE matches installed and documented version

7.1.45 FPT_TUD_EXT.1.3 Test #1

Test Assurance Activity	The evaluator shall verify that the application's executable files are not changed by the application. **Platforms:Apple iOS...** The evaluator shall consider the requirement met because the platform forces applications to write all data within the application working directory (sandbox). **For all other platforms**, the evaluator shall perform the following test: The evaluator shall install the application and then locate all of its executable files. The evaluator shall then, for each file, save off either a hash of the file or a copy of the file itself. The evaluator shall then run the application and exercise all features of the application as described in the ST. The evaluator shall then compare each executable file with the either the saved hash or the saved copy of the files. The evaluator shall verify that these are identical.
Test Steps	• The evaluator shall access the TOE application and find the directory where the executable files are located. • The evaluator shall save the hash of the exec files in a hash format. • The evaluator exercises all features of the application. • The evaluator shall save the new hash of the exec files in a hash format. • The saved files that are generated in the above step is compared.
Pass/Fail with Explanation	Pass. The evaluator verified by comparing the hashes and the files are identical.

7.1.46 FPT_TUD_EXT.2.1 Test #1

Test Assurance Activity	**Platforms:Android...** The evaluator shall ensure that the application is packaged in the Android application package (APK) format. **TD0628 has been applied.**
Test Steps	• The evaluator verifies that the TOE application is in the APK format.
Pass/Fail with Explanation	Pass. The TOE is in the APK format.

7.1.47 FPT_TUD_EXT.2.2 Test #1

Test Assurance Activity	**Platforms: Android...** The evaluator shall consider the requirement met because the platform forces applications to write all data within the application working directory (sandbox). **Applied TD0664**
Pass/Fail with Explanation	Pass. The requirement is met because the platform forces applications to write all data within the application working directory (sandbox).

7.1.48 FTP_DIT_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall exercise the application (attempting to transmit data; for example by connecting to remote systems or websites) while capturing packets from the application. The evaluator shall verify from the packet capture that the traffic is encrypted with HTTPS, TLS, DTLS, SSH, or IPsec in accordance with the selection in the ST.
Test Steps	• The evaluator shall attempt a connection from the TOE to the management server. • The evaluator shall confirm that the communication between the TOE and management server is encrypted with TLS using packet captures.
Pass/Fail with Explanation	Pass. The evaluator confirmed that the communication is encrypted using TLS.

7.1.49 FTP_DIT_EXT.1.1 Test #2

Test Assurance Activity	The evaluator shall exercise the application (attempting to transmit data; for example by connecting to remote systems or websites) while capturing packets from the application. The evaluator shall review the packet capture and verify that no sensitive data is transmitted in the clear.
Test Steps	• The evaluator shall attempt a connection from an external IT entity to the TOE. • Then confirm that the sensitive information transmitted between them encrypted and not in plain text using packet captures.
Pass/Fail with Explanation	Pass. The test has been covered as part of test FTP_DIT_EXT.1.1 Test #1

7.1.50 FTP_DIT_EXT.1.1 Test #3

Test Assurance Activity	The evaluator shall inspect the TSS to determine if user credentials are transmitted. If credentials are transmitted the evaluator shall set the credential to a known value. The evaluator shall capture packets from the application while causing credentials to be transmitted as described in the TSS. The evaluator shall perform a string search of the captured network packets and verify that the plaintext credential previously set by the evaluator is not found.
Test Steps	• The evaluator shall attempt a connection from the TOE to the management server. • The evaluator shall capture the packets that is being transmitted between the TOE and its management server. • The evaluator verifies that the credentials are not transmitted in plain text.
Pass/Fail with Explanation	Pass. The evaluator verified that user credentials are not being transmitted in plain text in the packet capture. This meets testing requirement.

7.2 PKG_TLSC

7.2.1 FCS_TLSC_EXT.1.1 Test #1

Test Assurance Activity	The evaluator shall establish a TLS connection using each of the cipher suites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a cipher suite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the cipher suite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).
Test Steps	• The evaluator shall check the public IP of the TOE. • The evaluator shall login to the TOE and establish a TLS session between the TOE and the management server. • The evaluator shall verify the cipher suits used to negotiate the TLS connection.
Pass/Fail with Explanation	Pass. The evaluator established a TLS connection using the cipher suite with the management server and observed the successful negotiation.

7.2.2 FCS_TLSC_EXT.1.1 Test #2

Test Assurance Activity	The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage extension and verify that a connection is established. The evaluator shall repeat this test using a different, but otherwise valid and trusted, certificate that lacks the Server Authentication purpose in the extendedKeyUsage extension and ensure that a connection is not established. Ideally, the two certificates should be similar in structure, the types of identifiers used, and the chain of trust.
Test Steps	• The evaluator shall load the server certificate (serverCert.crt) and the server key (serverCert_key.pem) to the management server and configure the server to leverage the loaded certificate and key to establish a TLS connection with the TOE. • The evaluator shall attempt a connection from the TOE to the management server and verify the connection to be successful. • The evaluator shall observe the packet capture and ensure that the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage extension established successfully. • The evaluator created an identical certificate (server_nsa_cert.crt) that is similar in structure, the types of identifiers used, and the chain of trust but lacks the Server Authentication purpose in the extendedKeyUsage extension. • The evaluator shall load the server certificate (server_nsa_cert.crt) and the server key (server_nsa_key.pem) to the management server and configure the server to leverage the loaded certificate and key to establish a TLS connection with the TOE. • The evaluator shall attempt a connection from the TOE to the management server and verify the connection to be unsuccessful. • The evaluator verifies with packet capture that connection is not successful.

Pass/Fail with Explanation	Pass. The TOE established the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage extension and did not establish a connection with a server certificate that lacks the Server Authentication purpose in the extendedKeyUsage extension.

7.2.3 FCS_TLSC_EXT.1.1 Test #3

Test Assurance Activity	The evaluator shall send a server certificate in the TLS connection that does not match the server-selected cipher suite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA cipher suite or send a RSA certificate while using one of the ECDSA cipher suites.) The evaluator shall verify that the product disconnects after receiving the server's Certificate handshake message.
Test Steps	• Create a certificate with ECDSA signature scheme for the server. • Run the acumen-tlsc tool to negotiate the server cipher suites not matching with server. • Verify unsuccessful connection between TOE and the server. • Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE denied connection to server using a certificate not matching its cipher suite.

7.2.4 FCS_TLSC_EXT.1.1 Test #4

Test Assurance Activity	The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL cipher suite and verify that the client denies the connection.
Test Steps	• The evaluator configures the server to TLS_NULL_WITH_NULL_NULL and attempts to establish a TLS session between the TOE and the remote server. • The evaluator verifies connection is not established. • The evaluator verifies with packet capture that client denies connection.
Pass/Fail with Explanation	Pass. Client denies connection to the server when Null cipher is configured.

7.2.5 FCS_TLSC_EXT.1.1 Test #5.1

Test Assurance Activity	Change the TLS version selected by the server in the Server Hello to an undefined TLS version (for example 1.5 represented by the two bytes 03 06) and verify that the client rejects the connection.
Test Steps	• The evaluator shall configure the server to use TLS 1.5. • The evaluator shall try to establish a TLS connection with the TOE and the management server. • The evaluator shall verify that the connections fails. • The evaluator verifies with packet capture that no connection is established.

Pass/Fail with Explanation	Pass. No connection is established between the TOE and server.

7.2.6 FCS_TLSC_EXT.1.1 Test #5.2

Test Assurance Activity	Change the TLS version selected by the server in the Server Hello to the most recent unsupported TLS version (for example 1.1 represented by the two bytes 03 02) and verify that the client rejects the connection.
Test Steps	• The evaluator shall configure the server to use TLS 1.1. • The evaluator shall try to establish a TLS connection with the TOE and the management server. • The evaluator shall verify that the connections fails. • The evaluator verifies with packet capture that the connection fails.
Pass/Fail with Explanation	Pass. TLS connection fails when server's TLS version is set to an unsupported version.

7.2.7 FCS_TLSC_EXT.1.1 Test #5.3

Test Assurance Activity	[conditional] If **DHE or ECDHE cipher suites are supported**, modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the client does not complete the handshake and no application data flows
Test Steps	• Run the acumen-tlsc tool to modify the server's nonce field in the Server Hello handshake message and verify the connection fails. • Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE rejects connection with the server due to a modified server nonce in the Server Hello handshake message.

7.2.8 FCS_TLSC_EXT.1.1 Test #5.4

Test Assurance Activity	Modify the server's selected cipher suite in the Server Hello handshake message to be a cipher suite not presented in the Client Hello handshake message. The evaluator shall verify that the client does not complete the handshake and no application data flows.
Test Steps	• The evaluator shall try to establish a TLS connection between the TOE and the management server(different cipher suite). • The evaluator shall verify that connection fails. • The evaluator verifies with packet capture that connection fails.
Pass/Fail with Explanation	Pass. TLS connection fails when server's cipher suite does not match with the client.

7.2.9 FCS_TLSC_EXT.1.1 Test #5.5

Test Assurance Activity	[conditional] If **DHE or ECDHE cipher suites are supported**, modify the signature block in the server's Key Exchange handshake message, and verify that the client does not complete the handshake and no application data flows. This test does not apply to cipher suites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.
Test Steps	• Run the acumen-tlsc tool to modify a byte in the signature block of the Server key exchange message on the server and verify that the connection fails. • Verify with packet capture the unsuccessful connection between TOE and the server. ○ Server has modified a byte in signature field of server key exchange message. ○ TOE rejects connection with the server.
Pass/Fail with Explanation	Pass. The TOE rejects connection to the server due to a modified signature block in the server's Key Exchange handshake message.

7.2.10 FCS_TLSC_EXT.1.1 Test #5.6

Test Assurance Activity	Modify a byte in the Server Finished handshake message, and verify that the client does not complete the handshake and no application data flows.
Test Steps	• Run the acumen-tlsc tool to modify a byte in the Server Finished handshake message. • Verify that the connection between TOE and server fails. • Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE rejects connection with the server due to the modified Server Finished handshake message.

7.2.11 FCS_TLSC_EXT.1.1 Test #5.7

Test Assurance Activity	Send a message consisting of random bytes from the server after the server has issued the Change Cipher Spec message and verify that the client does not complete the handshake and no application data flows. The message must still have a valid 5-byte record header in order to ensure the message will be parsed as TLS.
Test Steps	• Run the acumen-tlsc tool on the server to send a garbled message after the Change Cipher Spec message is issued and verify the connection fails. • Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE rejects connection to a server containing garbled messages received after the Change Cipher Spec message.

7.2.12 FCS_TLSC_EXT.1.2 Test #1

Test Assurance Activity	The evaluator shall present a server certificate that contains a CN that does not match the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection fails. TD0499 has been applied.
Test Steps	• Configure the TOE for reference identifier name as FQDN. • Configure the server certificate showing incorrect CN and no SAN. • Load the certificate with incorrect CN and no SAN on management server. • Initiate the connection from the TOE to the management server and verify that the connection fails. • Verify the unsuccessful connection due to incorrect CN no SAN in packet capture.
Pass/Fail with Explanation	Pass. The TOE rejects connection for server certificate with an incorrect CN and no SAN. The test should fail because the TOE mandates the presence of CN and SAN.

7.2.13 FCS_TLSC_EXT.1.2 Test #2

Test Assurance Activity	The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type. TD0499 has been applied.
Test Steps	• Configure the TOE for reference identifier name as FQDN. • Configure the server certificate showing correct CN and incorrect SAN. • Load the certificate with correct CN and incorrect SAN on management server. • Initiate the connection from the TOE to the management server and verify the connection fails. • Verify the unsuccessful connection due to correct CN and incorrect SAN in packet capture.
Pass/Fail with Explanation	Pass. The TOE rejects connection with the server due to the modified Server Finished handshake message.

7.2.14 FCS_TLSC_EXT.1.2 Test #3

Test Assurance Activity	[conditional] If the TOE does not mandate the presence of the SAN extension, the evaluator shall present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds. If the TOE does mandate the presence of the SAN extension, this Test shall be omitted. TD0499 has been applied.

Pass/Fail with Explanation	This test would be not applicable because the TOE mandates the presence of both CN and SAN extension.

7.2.15 FCS_TLSC_EXT.1.2 Test #4

Test Assurance Activity	The evaluator shall present a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator shall verify that the connection succeeds. TD0499 has been applied.
Test Steps	• Configure the TOE for reference identifier name as FQDN. • Configure the Server certificate showing incorrect CN and correct SAN. • Load the certificate with incorrect CN and correct SAN on management server. • Initiate the connection from the TOE to the server and verify the connection is successful. • Verify with packet capture the successful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE accepts the connection for the server certificate with an incorrect CN and correct SAN.

7.2.16 FCS_TLSC_EXT.1.2 Test #5.1

Test Assurance Activity	[conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g. foo.*.example.com) and verify that the connection fails. TD0499 has been applied.
Test Steps	• The evaluator created a server certificate containing a wildcard that is not in the left-most label of the presented identifier (cc.*.nubo.co). • The evaluator uploaded the server key and the certificate on the server and ensured it contains a wildcard that is not in the left-most label of the presented identifier (cc.*.nubo.co) in the CN and SAN • The evaluator configured the management server to leverage the uploaded server certificate and the key for TLS handshake. • The evaluator attempted a connection to the server and ensured that the TOE could not connect to the server. • The evaluator observed the packet capture and ensured that connection fails due to “certificate unknown” message.
Pass/Fail with Explanation	Pass. The evaluator verified that the TLS connection failed when the server presented a certificate containing a wildcard that is not in the left-most label of the presented identifier (cc.*.nubo.co).

7.2.17 FCS_TLSC_EXT.1.2 Test #5.2(a)

Test Assurance Activity	[conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label but not preceding the public suffix (e.g. *.example.com). - The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds. TD0499 has been applied.
Test Steps	• The evaluator created a server certificate that contains a wildcard in the left-most label but not preceding the public suffix (*.nubo.co). • The evaluator uploaded the server key and the certificate on the server and ensured it contains a wildcard in the left-most label but not preceding the public suffix (*..nubo.co) in the SAN and CN field. • The evaluator configured the management server to leverage the uploaded server certificate and the key for TLS handshake. • The evaluator attempted a connection to the server with identifier “cc.nubo.co” resolved to 142.93.243.137 and ensured that the connection was successful. • The evaluator observed the packet capture and ensured that the TLS handshake was successful.
Pass/Fail with Explanation	Pass. The evaluator verified that the TLS connection succeeds when the server sends a certificate with wildcard in the left-most label of the presented identifier.

7.2.18 FCS_TLSC_EXT.1.2 Test #5.2(b)

Test Assurance Activity	[conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label but not preceding the public suffix (e.g. *.example.com). - The evaluator shall configure the reference identifier without a left-most label as in the certificate (e.g. example.com) and verify that the connection fails. TD0499 has been applied.
Test Steps	• The evaluator created a server certificate that contains a wildcard in the left-most label but not preceding the public suffix (*.nubo.co). • The evaluator uploaded the server key and the certificate on the server and ensured it to contain a server certificate that contains a wildcard in the left-most label but not preceding the public suffix (*.nubo.co) in the CN and SAN field. • The evaluator configured the management server to leverage the uploaded server certificate and the key for TLS handshake. • The evaluator attempted a connection to the server with identifier “nubo.co” resolved to 142.93.243.137 and ensured that the TOE could not connect to the server. • The evaluator observed the packet capture and ensured that the client does not connect to the server due to “certificate unknown” error.

Pass/Fail with Explanation	Pass. The evaluator verified that the TLS connection failed when the server presented a certificate containing a wildcard in the left-most label but not preceding the public suffix while the reference identifier did not contain a left most label.

7.2.19 FCS_TLSC_EXT.1.2 Test #5.2(c)

Test Assurance Activity	[conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label but not preceding the public suffix (e.g. *.example.com). - The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.example.com) and verify that the connection fails. TD0499 has been applied.
Test Steps	• The evaluator created a server certificate that contains a wildcard in the left-most label but not preceding the public suffix (*.nubo.co). • The evaluator uploaded the server key and the certificate on the server and ensured it to contain a server certificate that contains a wildcard in the left-most label but not preceding the public suffix (*.nubo.co) in the CN and SAN field. • The evaluator configured the management server to leverage the uploaded server certificate and the key for TLS handshake. • The evaluator attempted a connection to the server with identifier “random.cc.nubo.co” resolved to 142.93.243.137 and ensured that the TOE could not connect to the server. • The evaluator observed the packet capture and ensured that the client does not connect to the management server due to “certificate unknown” error.
Pass/Fail with Explanation	Pass. The evaluator verified that the TLS connection failed when the reference identifier on the client was configured with two left-most labels while the server presented a certificate containing a wildcard in the left-most label but not preceding the public suffix.

7.2.20 FCS_TLSC_EXT.1.2 Test #5.3(a)

Test Assurance Activity	[conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label immediately preceding the public suffix (e.g. *.com). - The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.com) and verify that the connection fails. TD0499 has been applied.
Test Steps	• The evaluator created a server certificate that contains a wildcard in the left-most label immediately preceding the public suffix (*.co). • The evaluator uploaded the server key and the certificate on the server and ensured it contains a wildcard in the left-most label immediately preceding the public suffix (*.co) in the CN and SAN field. • The evaluator configured the management server to leverage the uploaded server certificate and the key for TLS handshake.

	• The evaluator attempted a connection to the server with identifier “nubo.com” resolved to 142.93.243.137 and ensured that the TOE could not connect to the server. • The evaluator observed the packet capture and ensured that the client does not connect to the server due to “certificate unknown” message.
Pass/Fail with Explanation	Pass. The evaluator verified that the TLS connection fails when the server presented a certificate with wildcard in the left-most label of the presented identifier while the reference identifier configured on the TOE does not contain a left most label.

7.2.21 FCS_TLSC_EXT.1.2 Test #5.3(b)

Test Assurance Activity	[conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label immediately preceding the public suffix (e.g. *.com). - The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.com) and verify that the connection fails. TD0499 has been applied.
Test Steps	• The evaluator created a server certificate that contains a wildcard in the left-most label immediately preceding the public suffix (*.cc). Same certificate(server_5.3a) is used for Test 5.3a and 5.3b. • The evaluator uploaded the server key and the certificate on the management server. • The evaluator configured the management server to leverage the uploaded server certificate and the key for TLS handshake. • The evaluator attempted a connection to the server with identifier “cc.nubo.co” resolved to 142.93.243.137 and ensured that the TOE could not connect to the server. • The evaluator observed the packet capture and ensured that the client does not complete the TLS handshake due to “certificate unknown” error.
Pass/Fail with Explanation	Pass. The evaluator verified that the TLS connection fails when the server presented a certificate with wildcard in the left-most label of the presented identifier immediately preceding the public suffix while the reference identifier configured on the TOE contain two left most labels.

7.2.22 FCS_TLSC_EXT.1.2 Test #5.4

Test Assurance Activity	[conditional]: If wildcards are not supported, the evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection fails. TD0499 has been applied.
Pass/Fail with Explanation	N/A. As per ST, the TOE supports wildcards, thus this test is omitted.

7.2.23 FCS_TLSC_EXT.1.2 Test #6

Test Assurance Activity	If URI or Service name reference identifiers are supported, the evaluator shall configure the DNS name and the service identifier. The evaluator shall present a server certificate containing the correct DNS name and service identifier in the URIName or SRVName fields of the SAN and verify that the connection succeeds. The evaluator shall repeat this test with the wrong service identifier (but correct DNS name) and verify that the connection fails.
Pass/Fail with Explanation	N/A because URI or Service name reference identifiers are not supported.

7.2.24 FCS_TLSC_EXT.1.2 Test #7

Test Assurance Activity	[conditional] If pinned certificates are supported the evaluator shall present a certificate that does not match the pinned certificate and verify that the connection fails.
Test Steps	Pinned Certificate • Check the current server certificate Public Key. • Initiate the connection from the TOE to the management server and verify the connection is successful. • Verify the successful connection in packet capture. Unpinned Certificate • Configure server certificate using new public key. • Load the server certificate using new public key on the management server. • Initiate the connection from the TOE to the management server and verify the connection fails. • Verify with packet capture the unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. TOE accepts connection to a pinned certificate while rejecting connection to a certificate whose public key does match with the pinned certificate.

7.2.25 FCS_TLSC_EXT.1.3 Test #1(a)

Test Assurance Activity	The evaluator shall demonstrate that a server using a certificate with a valid certification path successfully connects.
Test Steps	• Create a full chain of certificates. • Upload a complete certificate chain on the TLS server. • Initiate the connection from the TOE to the TLS Server and verify the connection is successful. • Verify with packet capture successful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE establishes the connection to the server using a certificate with a valid certification path.

7.2.26 FCS_TLSC_EXT.1.3 Test #1(b)

Test Assurance Activity	The evaluator shall modify the certificate chain used by the server in test 1a to be invalid and demonstrate that a server using a certificate without a valid certification path to a trust store element of the TOE results in an authentication failure. TD0513 has been applied.
Test Steps	• Show a complete certificate chain on the TOE's trust store. • Remove the rootcert from the TOE's trust store. • Initiate the TLS connection from the TOE to the TLS server without the CA in the trust store. • Verify with packet capture unsuccessful connection between TOE and the server.
Pass/Fail with Explanation	Pass. TOE rejects connection for server certificate with invalid certification path to its trust store element

7.2.27 FCS_TLSC_EXT.1.3 Test #1(c)

Item	**Data**
Test Assurance Activity	[conditional]: **If the TOE trust store can be managed**, the evaluator shall modify the trust store element used in Test 1a to be untrusted and demonstrate that a connection attempt from the same server used in Test 1a results in an authentication failure. **TD0513 has been applied.**
Test Steps	N/A because the TOE does not manage a trust store.
Expected Test Results	N/A because the TOE does not manage a trust store.
Test Output	N/A because the TOE does not manage a trust store.
Pass/Fail with Explanation	N/A because the TOE does not manage a trust store.

7.2.28 FCS_TLSC_EXT.1.3 Test #2

Test Assurance Activity	The evaluator shall demonstrate that a server using a certificate which has been revoked results in an authentication failure.
Pass/Fail with Explanation	N/A. This test is covered as a part of FIA_X509_EXT.1.1 Test #3.

7.2.29 FCS_TLSC_EXT.1.3 Test #3

Test Assurance Activity	The evaluator shall demonstrate that a server using a certificate which has passed its expiration date results in an authentication failure.

Pass/Fail with Explanation	This test is covered as a part of FIA_X509_EXT.1.1 Test #2.

7.2.30 FCS_TLSC_EXT.1.3 Test #4

Test Assurance Activity	The evaluator shall demonstrate that a server using a certificate which does not have a valid identifier results in an authentication failure.
Pass/Fail with Explanation	This test is covered as a part of FCS_TLSC_EXT.1.2 Test 1-5.3

7.2.31 FCS_TLSC_EXT.5.1 Test #1

Test Assurance Activity	The evaluator shall configure a server to perform key exchange using each of the TOE's supported curves and/or groups. The evaluator shall verify that the TOE successfully connects to the server.
Test Steps	• Configure the valid identifier on TOE. • Create a certificate which does have a valid identifier. • Attempt the connection from the TOE to the Server with Curve secp384r1 and verify the connection is successful. • Verify with packet capture the successful connection between TOE and the server.
Pass/Fail with Explanation	Pass. The TOE is able to establish successful connection with server for the claimed EC curves

8 Conclusion

The testing shows that all test cases required for conformance have passed testing.

End of Document