NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

ORION: ON-DEMAND REGISTRATION AND REVOCATION IN ON-THE-MOVE NETWORKS

by

Jack J. Chang

September 2020

Thesis Advisor:	Geoffrey G. Xie
Co-Advisor:	Gurminder Singh

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2020	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE ORION: ON-DEMAND REGISTRATION AND REVOCATION IN ON-THE-MOVE NETWORKS	**5. FUNDING NUMBERS**
6. AUTHOR(S) Jack J. Chang	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

The management complexity, hardware limitations, and lack of scalability in the Marine Corps' traditional networking infrastructure creates an opportunity gap that can be filled by software-defined networking (SDN). At the same time, mobile ad-hoc networks (MANETs) have proved to be indispensable in austere environments, allowing tactical units to communicate without the need for permanent infrastructure. Anticipating the proliferation of mobile hand-held technology, a case is made for On-Demand Registration/Revocation in On-the-Move Networks (ORION), a flexible public key infrastructure (PKI) authentication framework for ad-hoc mobile devices. Resembling a localized extension of DISA's Purebred solution, ORION was designed specifically for tactical edge networks. ORION combines the centralized management and programmable capabilities of SDN with the decentralized, self-healing properties of MANET into one scalable, autonomous, interoperable system. The proposed model is designed, developed, and evaluated to demonstrate that forward-deployed, SDN-hosted Certificate Authorities are capable of providing PKI services to edge devices under adversarial network conditions characterized by low bandwidth, high latency, and high loss probabilities.

14. SUBJECT TERMS software-defined networks, SDN, Mobile Ad-Hoc Networks, MANET, Internet of Things, IoT, Public Key Infrastructure, PKI, authentication, authorization, Certificate Authority, CA, key management	**15. NUMBER OF PAGES** 159
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

ORION: ON-DEMAND REGISTRATION AND REVOCATION IN ON-THE-MOVE NETWORKS

Jack J. Chang
Captain, United States Marine Corps
BS, U.S. Naval Academy, 2014

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2020

Approved by: Geoffrey G. Xie
Advisor

Gurminder Singh
Co-Advisor

Gurminder Singh
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The management complexity, hardware limitations, and lack of scalability in the Marine Corps' traditional networking infrastructure creates an opportunity gap that can be filled by software-defined networking (SDN). At the same time, mobile ad-hoc networks (MANETs) have proved to be indispensable in austere environments, allowing tactical units to communicate without the need for permanent infrastructure. Anticipating the proliferation of mobile hand-held technology, a case is made for On-Demand Registration/Revocation in On-the-Move Networks (ORION), a flexible public key infrastructure (PKI) authentication framework for ad-hoc mobile devices. Resembling a localized extension of DISA's Purebred solution, ORION was designed specifically for tactical edge networks. ORION combines the centralized management and programmable capabilities of SDN with the decentralized, self-healing properties of MANET into one scalable, autonomous, interoperable system. The proposed model is designed, developed, and evaluated to demonstrate that forward-deployed, SDN-hosted Certificate Authorities are capable of providing PKI services to edge devices under adversarial network conditions characterized by low bandwidth, high latency, and high loss probabilities.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

2FA	two-factor authentication
3DES	Triple Data Encryption Standard
AAA	Authentication, Authorization, and Accounting
AAL	Authenticator Assurance Level
AES	Advanced Encryption Standard
AM	amplitude modulation
ANW2	Adaptive Networking Wideband Waveform
AOR	Area of Responsibility
API	application programming interface
ARPANET	Advanced Research Projects Agency Network
AS	authentication server
ASCII	American Standard Code for Information Interchange
BACN	Battlefield Airborne Communications Node
BER	bit error rate
BGAN	Broadband Global Area Network
BLOS	beyond line-of-sight
BN	battlefield networking
BYOD	Bring Your Own Device
C2	command and control

C4ISR command, control, communications, computers and intelligence, surveillance, and reconnaissance

CAC Common Access Card

CA certificate authority

CCI Controlled Cryptographic Item

CHAP Challenge Handshake Authentication Protocol

CIK crypto ignition key

CIO Chief Information Officer

CIO Chief Information Officer

CLI command line interface

CMD commercial mobile device

CMS Cryptographic Message Syntax

CN Common Name

COR Central Office of Record

COTS commercial-off-the-shelf

CPA Communications Programming Application

CPG Commandant's Planning Guidance

CPS Certification Practice Statement

CRL certificate revocation list

CSfC Commercial Solutions for Classified

CSR Certificate Signing Request

D2E denied and degraded environment

DDoS Distributed Denial of Service

DECC Defense Enterprise Computing Center

DEERS Defense Enrollment Eligibility Reporting System

DEPSECDEF Deputy Secretary of Defense

DES Data Encryption Standard

DIACAP DoD Information Assurance Certification and Accreditation Process

DISA Defense Information Systems Agency

DMDC Defense Manpower Data Center

DMO Distributed Maritime Operations

DON Department of the Navy

DTD Data Transfer Device

DTLS Datagram Transport Layer Security

DoDIN Department of Defense Information Network

DoD Department of Defense

EABO Expeditionary Advance Base Operations

EAP Extensible Authentication Protocol

ECC elliptic-curve cryptography

EDIPI electronic data interchange personal identifier

EST Enrollment over Secure Transport

FIPS Federal Information Processing Standard

FMR False Match Rate

FM frequency modulation

GFE Government Furnished Equipment

GOTS government-off-the-shelf

GPS global positioning system

GUI graphical user interface

HMAC hash-based message authentication code

HSM hardware security module

HSPD-12 Homeland Security Presidential Directive-12

I/O input/output

IA Information Assurance

IANA Internet Assigned Numbers Authority

ICC integrated circuit chip

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

INFOSEC Information Security

IP Internet Protocol

ISP Internet service provider

IoT Internet of Things

JTRS Joint Tactical Radio System

JVM Java Virtual Machine

KDC Key Distribution Center

KDF key derivation function

keymat keying material

KME Key Management Extension

LAN local area network

LOCE Littoral Operations in a Contested Environment

LOS line-of-sight

LOS line-of-sight

MAC message authentication code

MANET mobile ad-hoc network

MAS Mobile Application Store

MCEN Marine Corps Enterprise Network

MD5 Message Digest 5

MDM Mobile Device Management

MFA multi-factor authentication

MITM man-in-the-middle

MIT Massachusetts Institute of Technology

MMU memory mapping unit

NAS Network Access Server

NAT Network Address Translation

NATO North Atlantic Treaty Organization

NDS National Defense Strategy

NFC Near Field Communication

NFV Network Function Virtualization

NIAP National Information Assurance Partnership

NIST	National Institute of Standards and Technology
NOAA	National Oceanic and Atmospheric Administration
NOS	network operating system
NSA	National Security Agency
OCSP	Online Certificate Status Protocol
ONOS	Open Network Operating System
ORION	On-Demand Registration/Revocation in On-the-Move Networks
OS	operating system
OSGi	Open Services Gateway initiative
OTA	Over-the-Air
OTP	one-time password
P2P	point-to-point
PAN	personal area network
PAP	Password Authentication Protocol
PBA	Purebred Agent
PCMCIA	Personal Computer Memory Card International Association
PIN	Personal Identification Number
PIV	Personal Identity Verification
PKCS	Public Key Cryptography Standards
PKE	public key-enabled
PKITE	Public Key Infrastructure in a Tactical Environment

PKI Public Key Infrastructure

PMO Project Management Office

PNAC port-based Network Access Control

POM Project Object Model

PPP Point-to-Point Protocol

RADIUS Remote Authentication Dial-In User Service

RAPIDS Real-time Automated Personnel Identification System

RA registration authority

RFC Request for Comments

RF radio frequency

RISC reduced instruction set computer

RMF Risk Management Framework

RSA Rivest-Shamir-Adleman

SATCOM satellite communication

SCA Software Communications Architecture

SCEP Simple Certificate Enrollment Protocol

SCTP Streaming Control Transmission Protocol

SD-MANET Software-defined Mobile Ad-hoc Networking

SDN software-defined networking

SDR software-defined radio

SE Secure Element

SHA-3 Secure Hash Algorithm 3

SINCGARS Single Channel Ground and Airborne Radio System

S/MIME Secure/Multipurpose Internet Mail Extensions

SMS short message service

SPAN smartphone ad-hoc network

SRW Soldier Radio Waveform

SSL Secure Sockets Layer

TACAS+ Terminal Access Controller Access-Control System Plus

TA Trusted Agent

TCP Transmission Control Protocol

TCP Transmission Control Protocol

TDMA Time Division Multiple Access

TEE Trusted Execution Environment

TGS Ticket Granting Server

TGT Ticket Granting Ticket

TLS Transport Layer Security Protocol

TLS Transport Layer Security

TOTP time-based one-time password

TSM-X Tactical Scalable MANET X

TTP Tactics (Tools), Techniques, and Procedures

UAV Unmanned Aerial Vehicle

UDP User Datagram Protocol

UHF ultra high frequency

URL	Uniform Resource Locator
USB	Universal Serial Bus
UTF-8	Universal Transformation Format 8-bit
VANET	vehicular ad-hoc network
VHF	very high frequency
VM	virtual machine
VoIP	Voice over IP
VSAT	Very Small Aperture Terminal
WEZ	weapons engagement zone
WNW	Wideband Networking Waveform
XML	Extensible Markup Language

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

Foremost, I would like to express my deepest appreciation to my thesis advisors, Dr. Geoffrey Xie and Dr. Gurminder Singh, for their professional guidance, technical expertise, and patience throughout the planning and development of this research project. Your insightful suggestions, generosity with your time, and belief in my abilities were instrumental in the completion of this thesis.

Coming from a non-CS background, the completion of my master's degree would not be possible without the invaluable support of the 368-191 Marine cohort and my fellow Naval brethren classmates. Thank you for the comradery, the stimulating discussions, the sleepless nights of coding, the early morning study sessions, and the memorable laughs we shared over the last two years. Semper Fidelis.

Finally, I want to extend my heartfelt gratitude to my loving family whom I dedicate this thesis to: my parents, Morica and Gary Chang, and my younger brother, Senior Airman Kevin Chang, who is currently serving on active duty in the Air Force. Thank you for being my biggest lifelong supporters.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

We must get our critical infrastructure and vulnerabilities "off the X" by establishing mobile, low-signature forward presence. We must develop distributed, low-signature, lethal, networked, persistent, and risk-worthy joint expeditionary capabilities that can persist and operate within the adversary's weapons engagement zone. We must introduce uncertainty into the adversary risk calculus with more expeditionary bases, distributed signatures, and operationally relevant capabilities and posture. We must maintain persistent, forward forces with high lethality and operational reach to ensure we keep "a foot in the door" and don't have to risk "kicking in the door." We must provide resilience to our forward stand-in forces with relative economy.

—General David H. Berger, 38th Commandant of the Marine Corps [1]

After more than eighteen years of counter-terrorism operations abroad, the "war on terror" no longer remains the primary concern for the U.S. Department of Defense (DoD) [2]. With the accelerated growth of technology, the erosion of international order, and the re-emergence of near-peer threats, the U.S. National Defense Strategy (NDS) has reoriented its primary focus to inter-state strategic competition, otherwise known as the "great power competition" between both China and Russia. As noted in the NDS, this ever-complex operating environment has prompted a rapid modernization of the U.S. military as America seeks to maintain dominant superiority in every operating domain– air, land, sea, space, cyberspace. Key prioritizations of this modernization are the recognition of cyberspace as a warfighting domain and increased innovation in command, control, communications, computers and intelligence, surveillance, and reconnaissance (C4ISR) capabilities [2]. As stated in the Department of the Navy (DON) 2020 Information Superiority Vision, *Information is Combat Power* [3].

Research in wargames pertaining to the NDS have demonstrated the collaborative advantage of allied partners in great power conflicts. Essential to this partnership is the need for communications and information network interoperability across forward deployed

forces [4]. In keeping with the NDS, the 38th Commandant's Planning Guidance (CPG) further emphasizes the dire importance of improving the command and control (C2) processes to better support joint/coalition interoperability as well as the Marine Corps' warfighting philosophy of maneuver warfare. The requirement to innovate C2 processes is especially paramount in a presumably denied and degraded information network environment. As the CPG contends, modern operations have become more distributed than ever before, requiring networks that provide resiliency, robustness, and secure access for mission success.

The realization of U.S. cyberspace dominance necessitates an aggressively reduced reliance on expensive, hardware-defined solutions. Hardware technology runs counter to the military's modernization and automation endeavor as it requires manual configuration, is difficult to upgrade, and exhausts already limited resources/storage capacity. To meet this cyber strategy, C2 equipment must not only be reliable and secure, it must also be adaptable, scalable, lightweight, and inexpensive to replace. In concert with the Navy's Distributed Maritime Operations (DMO) strategy, the Marine Corps has shifted towards a Stand-in Forces concept in anticipation of a greater role in the maritime littorals– Littoral Operations in a Contested Environment (LOCE) and Expeditionary Advance Base Operations (EABO) [4]. With a renewed interest in the Indo-Pacific Area of Responsibility (AOR), the Marine Corps' ability to generate "technically disruptive, tactical stand-in engagements" within an adversary's long-range weapons engagement zone (WEZ) will require "low signature, affordable, and risk-worthy platforms and payloads" [4].

To exact speed and generate operational tempo over the competition, the USMC must take advantage of advanced systems that can virtualize hardware functionalities on-demand– software-defined systems. With software-defined technology, the rigid, physical limitations of hardware are removed, exposing a flexible system capable of performing multiple hardware roles. For the warfighter, machine automation provides the ability to eliminate tasks that are "repetitive, time-consuming, and routine" [4]. Per the Commandant's guidance regarding warfighting investments in artificial intelligence, data science, and emerging technology:

> We must set conditions so that the Marines can focus on warfighting tasks rather than data entry and redundant administrative processes. This will make the Marine Corps more lethal. [4]

The current USMC network mobility model for forward-deployed ground forces relies on a framework of software-defined radios (SDRs) with limited reachback capability. These SDRs are able to form mobile ad-hoc networks (MANETs), decentralized, self-healing networks capable of supporting tactical operations in environments which lack existing communications infrastructure– a necessary requirement for modern military communications systems. The USMC currently lacks adaptable authentication schemes in edge networks that can leverage software automation while taking user identity into account. The current approach to MANET authentication is based on symmetric keys and involves cumbersome cryptographic fill devices as well as complex key management infrastructures. As such, the present methodology suffers from a lack of scalability, interoperability, and autonomy.

The DoD Public Key Infrastructure (PKI) serves as the identity and authentication mechanism for the Department of Defense Information Network (DoDIN) and uses the issued Common Access Card (CAC) as the primary token for credentialing. PKI is important for secure, encrypted communications because public key certificates ensure that "someone is who they say they are" and that the information they are sending has not been modified in transit. However, PKI was not originally designed for implementation in tactical networks. Thus, PKI is mainly employed in the garrison environment on wired networks. PKI leverages asymmetric keys to overcome the issue of scalability but requires a robust and reliable backbone connection to the DoDIN in order to support and access back-end services. These challenges, among others, have curtailed PKI employment in tactical edge networks. Furthermore, the CACs presents modernization challenges due to its dependence on smart card readers making its functionality inadequate for the environment described in the NDS– an area that can be improved with software-based methods. Even then, certificate revocation and key recovery still present a challenge in a denied and degraded environment (D2E) as backbone connections are not always available, especially in tactical edge networks. For example, users with expired or revoked certificates should be immediately exempt from network access. Conversely, users who require a certificate renewal should not experience a lapse or interruption in network services.

With the pervasive development in mobile, hand-held technology, smartphones have become ubiquitous, enjoying increased adoption across the world, including in developing nations. More than just a basic communications device, smartphones have become

integrated into every facet of life– from reading the news to streaming Netflix, vehicle navigation, social media, banking, shopping on Amazon, ride-sharing, etc. As smartphone popularity continues to grow, so too will society's reliance on these smart devices to accomplish everyday tasks. Figures 1.1 and 1.2 show the growth of smartphones globally over recent years.

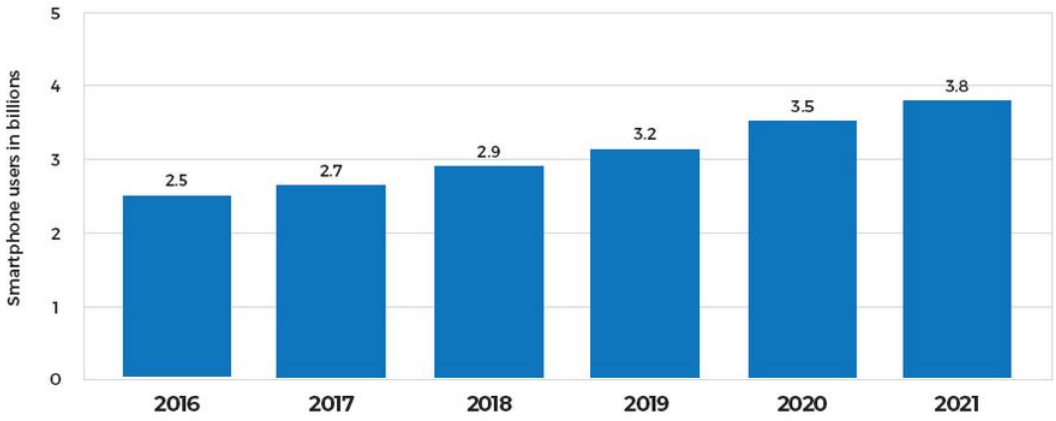


Figure 1.1. Number of Smartphone Users Worldwide. Source: [5].

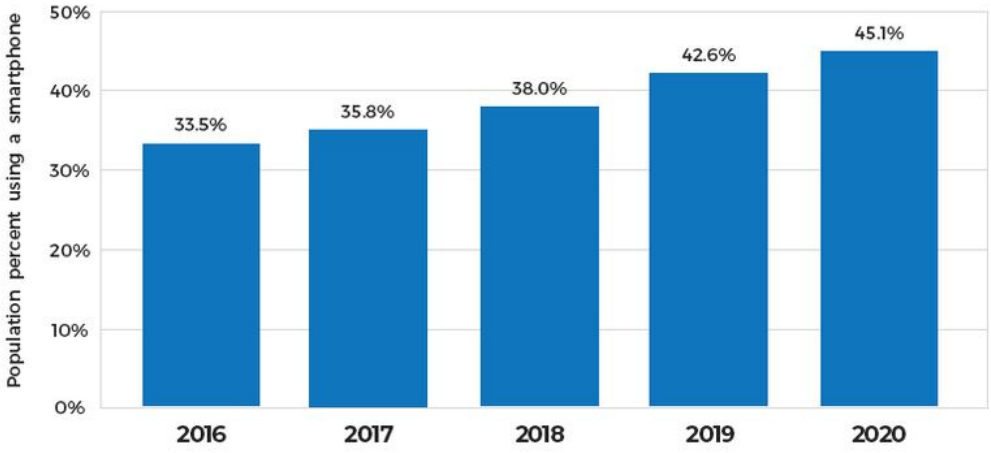


Figure 1.2. Percentage of Global Population Using Smartphones. Source: [5].

Given the smartphone's versatility, it is no surprise that military leadership has demonstrated great interest in its potential advantages, both on the battlefield and in garrison. Compared to the heavy, expensive radios currently in use, smartphones boast advanced computational capabilities in an inexpensive, lightweight, portable package. The technol-

ogy is familiar and user-friendly to younger troops, enabling superior learnability, productivity, and user satisfaction from inception. The value of combat-ready smartphones has yet to be fully realized. For example, the built-in global positioning system (GPS) can be used for terrain mapping, the cameras can be used to relay time-critical pictures and videos to commanders, and applications can be developed to support medical services, logistical needs, and track troop movements [6].

The advent of software-defined networking (SDN) brings centralized control and programmability to network management through separation of the data and control plane. SDN affords limitless potential for establishing new network services through an abstraction of the data plane. The implications are significant– networked hardware devices no longer need to implement complicated Internet Protocol (IP) routing protocols and instead become simple forwarding devices. Moreover, management of the control plane falls to logical controllers which decide how packets will flow throughout the network. Can we leverage SDN technology to facilitate secure authentication in tactical MANETs? In other words, in what scenarios could a SDN-based distributed certificate authority (CA) be deployed to improve network performance and support PKI in edge networks?

1.1 Motivating Scenario

The number of public key-enabled (PKE)-devices per warfighter is projected to increase as the battlefield becomes more digitally connected [7]. The growth of PKE-devices will easily outnumber the number of people, prompting a demand for increased software automation; SDN is well-positioned to address these future challenges. Given the objectives and requirements of the operating environment described in the NDS and CPG, we envision a future USMC mobility model that employs smartphones for ad-hoc networking– also known as a smartphone ad-hoc network (SPAN). A depiction of this vision is outlined as follows:

- A company-sized formation of foot-mounted troops dispersed 100m apart with limited mechanized support.
- Each individual is carrying a smartphone.
- Each smartphone is a participating node of a MANET for purposes of facilitating C2.
- Each node is able to transmit unicast, multicast, and broadcast messages.

- The SPAN leverages mobile PKI credentialing to support an authentication scheme that provides confidentiality, integrity, and non-repudiation.
- Reachback connectivity to the battalion combat operations center is degraded/unavailable.

The scenario described in Figure 1.3 is not impractical to conceptualize given the rapid pace of advancement in mobility and networking. Government-procured smartphones already exist within the DoD, but adoption has been slow and fielding has generally been limited to senior leadership. Within the last four years, the DoD has developed and released a mobile PKI system called Purebred which implements derived PKI credentials for clients on unclassified mobile platforms to access remote content, sign encrypted e-mails, and securely browse the Internet without the use of a CAC [8]. The derived credentials are escrowed private keys and X.509 public key certificates that maintain the same expiration datetime as their respective CAC certificates, either generated from a DoD CA or retrieved from a certificate repository [8].

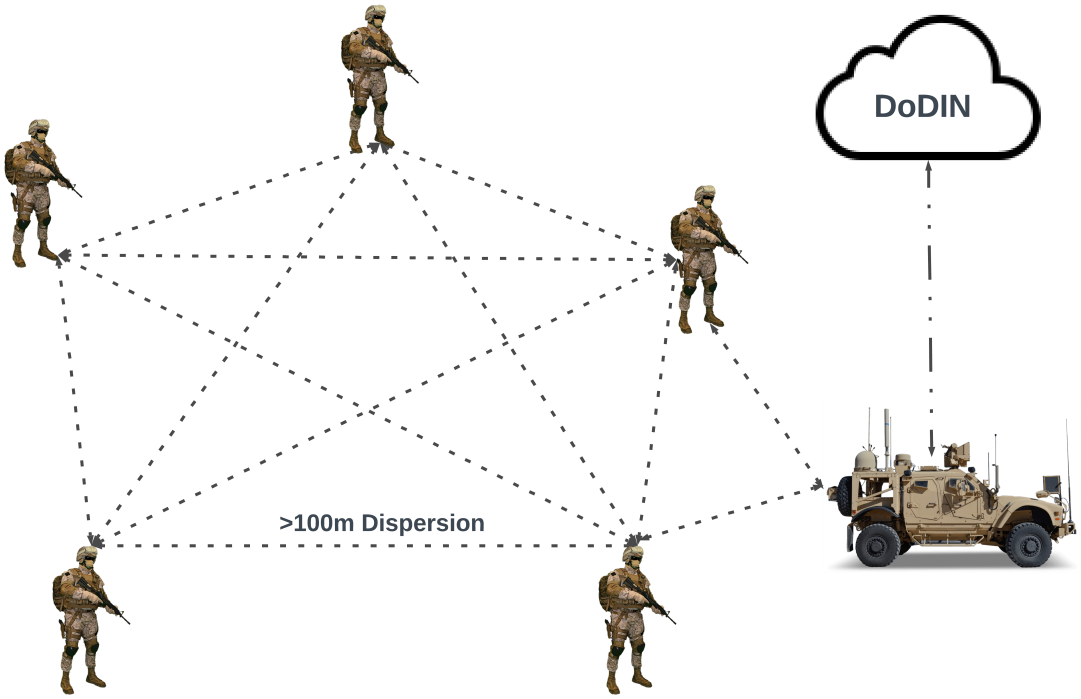


Figure 1.3. Mobile Ad-Hoc Network with Vehicle-Mounted Reachback

In the future operating environment, we anticipate the proliferation of the Purebred concept beyond garrison. To be specific, we envision an authentication scheme that resembles a localized extension of Purebred for networks at the tactical edge. In a D2E, this extension must be capable of supporting device registration (to include credentialing) and revocation in the absence of a reachback to PKI back-end services. We have decided to call this authentication scheme *On-Demand Registration/Revocation in On-the-Move Networks (ORION)*.

1.2 Research Questions

Given that a complete replication of Purebred registration and revocation services in the MANET setting would likely be resource-prohibited and would expose sensitive systems and data that are not necessary in the MANET, we seek to address the critical challenges facing ORION by answering the following research questions:

1. *In the event that a mobile device is lost, destroyed, or compromised, how can SDN facilitate timely revocation of the device's derived credentials?*
2. *How can SDN be exploited to securely automate the registration and credentialing process of new mobile devices?*

As the Marine Corps continues to modernize and invest in smartphone technologies, the requirement to properly manage and authenticate these devices is paramount. Given its programmability and centralized management, the SDN paradigm is well-poised to address the fundamental obstacles curtailing the propagation of PKI authentication to the tactical edge. The primary scope of this thesis is to design and develop ORION– a Software-defined Mobile Ad-hoc Networking (SD-MANET) authentication architecture for a potential USMC network mobility model which leverages Purebred functionality and uses commercial-off-the-shelf (COTS) and government-off-the-shelf (GOTS) products. We seek to demonstrate that SDN controllers can be forward deployed to host CAs within a MANET in order to extend PKI functionalities to tactical edge networks. In support of the NDS and CPG, the desired end state is that ORION will maximize the lethality, operational tempo, and security of Marine combat units through the exploitation of software-defined automation. To achieve this end state, ORION will aggressively leverage SDN and mobile technology to facilitate C2, mobility, scalability, interoperability with joint/coalition forces, and authenti-

cation in constrained (i.e., low bandwidth, high packet loss rate) and contested networking environments.

1.3 Thesis Organization

This thesis is divided into six chapters and is organized as follows. Chapter 1 frames the problem statement and motivation for the research. Chapter 2 describes background technology and related work. Chapter 3 explores the design of the proposed authentication scheme, ORION. Chapter 4 discusses the implementation process. Chapter 5 provides the results of the evaluation and experimentation process as well as lessons-learned. Chapter 6 concludes with a summary of the thesis and examines future work to be pursued.

CHAPTER 2:
Background

Everything changes so fast and the rules are against us. The most important thing we can do is be open to new ideas. Not to create chaos, not to create friction. But I am confident enough in the intellect and advice, that we will come up with the right solution, even if it is only 80 percent. If we do nothing, we lose. I am willing to take risk.

—General Robert B. Neller, 37th Commandant of the Marine Corps [9]

2.1 Authentication Concepts

The purpose of this section is to provide a general overview of widely-used authentication schemes and their application in the DoD with a focus on enterprise and commercial mobile devices (CMDs).

Data breaches are a growing problem across the world. In 2019 alone, at least 5.3 billion records from businesses to universities and government agencies (Facebook, T-Mobile, Capital One, the Federal Emergency Management Agency, and Georgia Tech, to name a few) had their databases compromised [10]. The pervasive growth of Internet of Things (IoT) connected devices and the concerns for data privacy management have indubitably brought about its own set of challenges and highlighted the necessities for information protection. In order to effectively protect an organization's data in the modern information age, the establishment of a robust access control policy is required [11]. Access control is defined as a method of ensuring that users are who they say they are (authentication) and that they have the appropriate access to the data they want to access to (authorization) [11]. The desired end state of an access control policy is to limit the roles and permissions of end users and/or computer systems on the network [12].

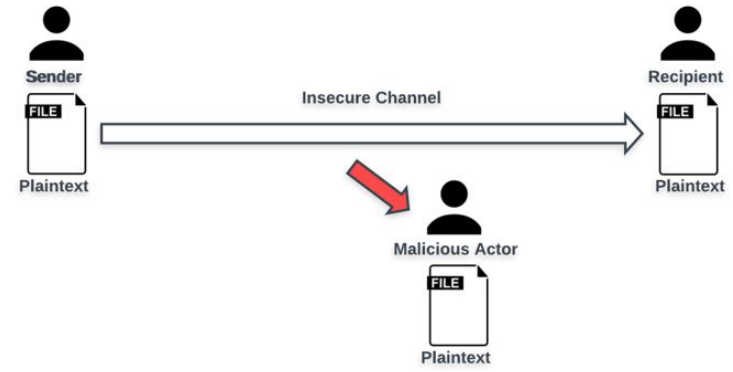
Since November 28, 2007, the DoD Information Assurance Certification and Accreditation Process (DIACAP) has served as the model for risk management on DoD information systems [13]. Specifically, the DIACAP defined the DoD Information Assurance (IA) re-

quirements and capabilities between all DoD systems and their enclaves- to include access control [13]. In March 12, 2014, the DoD Chief Information Officer (CIO) issued DoDI 8510.01 which announced the retirement of DIACAP and the adoption of the National Institute of Standards and Technology (NIST) Risk Management Framework (RMF) [14]. This marked, for the first time, a departure from legacy DoD-specific standards to a holistic alignment with federal standards in an effort to eliminate situations requiring compliance of two differing standards while reducing vendor costs [15]. See Table B.1 in Appendix A for current DoDIN access control management references..

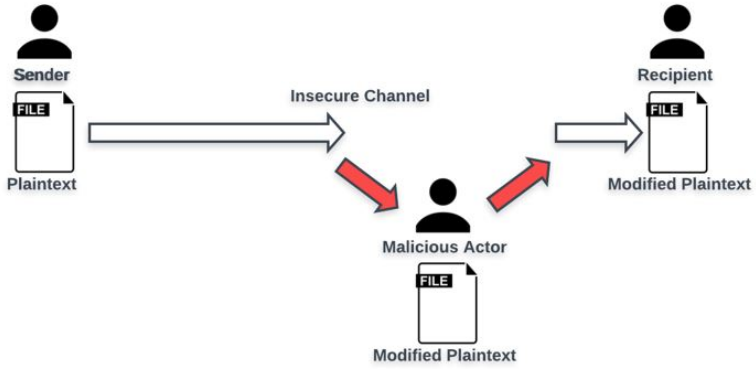
2.1.1 Password-based Authentication

Traditional non-cryptographic password-based authentication methods fail to provide adequate protection due to its inability to provide confidentiality, integrity, or non-repudiation [16]. Most user created passwords are, arguably, weak and reused across multiple different accounts [17]. Passwords can be cracked by brute force (e.g., dictionary attacks); however, increasing the complexity of a password increases the difficulty for user memorization. Even then, password strength is limited by the amount of effort placed into creating them and the best passwords can still be exploited by social engineering or man-in-the-middle (MITM) attacks (e.g., phishing, eavesdropping) [18]. If the authenticator or database where the passwords are stored is breached, the passwords could be compromised.

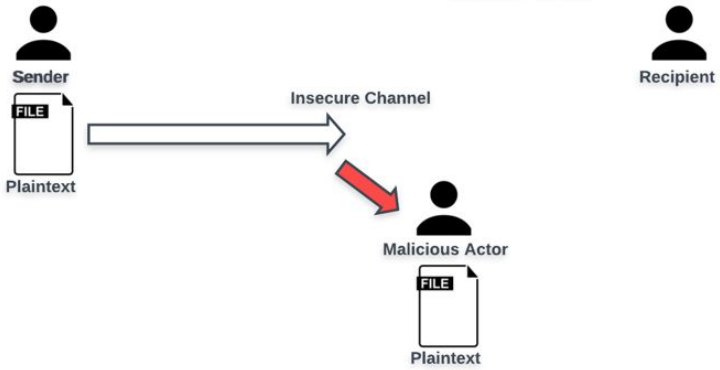
An improvement to static password authentication is a dynamic approach known as one-time passwords (OTPs) [19]. OTPs were derived from a need to safeguard against replay attacks due to the potential for passive network eavesdropping [20]. OTPs are typically utilized as part of a two-factor authentication (2FA) or multi-factor authentication (MFA) scheme since OTPs are considered "something you have" whereas passwords are considered as "something you know" [21]. OTPs are generated by combining the user password with an unique seed and function (e.g., secure hash) and then delivered to the client through various methods (e.g., security token, printed paper, software application, short message service (SMS) text) [21]. OTPs protect against password replay and reuse because a new OTP is generated for each session or transaction; however, its use as a shared key still makes OTPs vulnerable to MITM attacks (see Figure 2.1) [18], [21].



(a) Eavesdropping



(b) Manipulation



(c) Interception

These figures shows the most common MITM attacks on an insecure channel

Figure 2.1. MITM Attacks

(a) SKL. Source: [22].

Processor	32-bit Intel® XScale™ CPU (400 MHz)
Operating System	Windows® CE.net™ (4.2)
Memory	98MB Flash ROM
Working SDRAM	64MB
Storage SDRAM	64 MB(Battery Backed-up)
Graphics	2-D Accelerator for high-speed image manipulation
Display	3.5" QVGA, 65K Color Sunlight-readable Transflective TFT LCD Display, LED Backlit, Manual Brightness Control, Color Mapping for NVG use (8-bit mode) (Optional VGA CRT Output)
USB	1 USB mini-A host port, 1 USB mini-B device port
Size / Weight	7.45" x 4.25" x 2.25", 22.6 oz. with Standard Li-Ion Battery. Standard Li-Ion Battery: 9 oz. Heavy Duty Li-Ion Battery Pack: 14.3 oz.
Battery / Run-time	Standard Li-Ion Battery Pack, 6-24 hours, Heavy Duty Li-Ion Battery Pack, 12-48 hours

(b) SKL Specifications. Adapted from [22]

The SKL is a ruggedized hand-held USMC key management device that replaced the AN/CYZ-10 Data Transfer Device (DTD) [23]. Embedded within the SKL is a KOV-21 Personal Computer Memory Card International Association (PCMCIA) Information Security (INFOSEC) card that was developed and authorized by the National Security Agency (NSA) to encrypt and decrypt cryptographic functions using a removable crypto ignition key (CIK) [23], [24]. Due to the sensitive nature of the KOV-21 card, the SKL is a Controlled Cryptographic Item (CCI) and accounted by the Central Office of Record (COR) by serial number (ALC 1) [23]. The SKL is primarily used within the Marine Corps to securely distribute keying material (keymat) onto communications equipment [23].

Figure 2.2. AN/PYQ-10 Simple Key Loader (SKL)

2.1.2 Cryptographic Mechanisms

Cryptographic authentication protocols that leverage symmetric-key algorithms (e.g., Advanced Encryption Standard (AES), Data Encryption Standard (DES), Triple Data Encryption Standard (3DES)) rely on a shared key for both encryption and decryption (see Figure 2.3). Symmetric-key cryptography is primarily used for encryption (confidentiality) since the algorithms are considered fast and strong [16]. In this fashion, the sender converts the plaintext to ciphertext using the symmetric algorithm and the shared key [16]. The

recipient uses the same key to decrypt the ciphertext back to plaintext [16]. Key management, the establishment and sharing of the key between the sender and receiver(s), is the most difficult aspect of symmetric-key cryptography and the attack surface and complexity only grows as the number of users increase [16]. Common methods of key management are through "out-of-band" mechanisms such as a cryptographic fill device or a Key Distribution Center (KDC) outside of the shared network [22], [16].

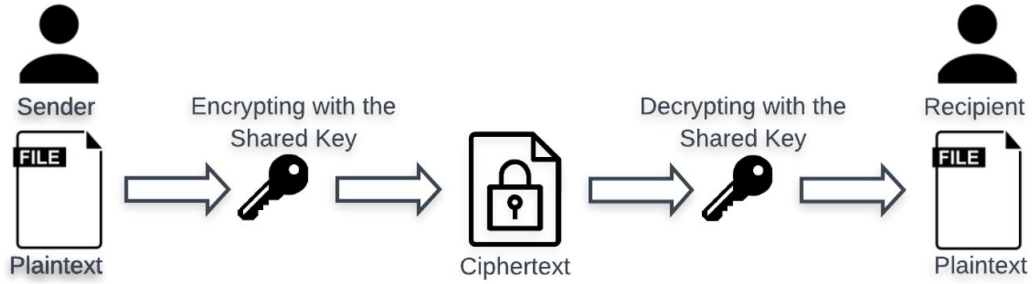


Figure 2.3. Symmetric-key Cryptography

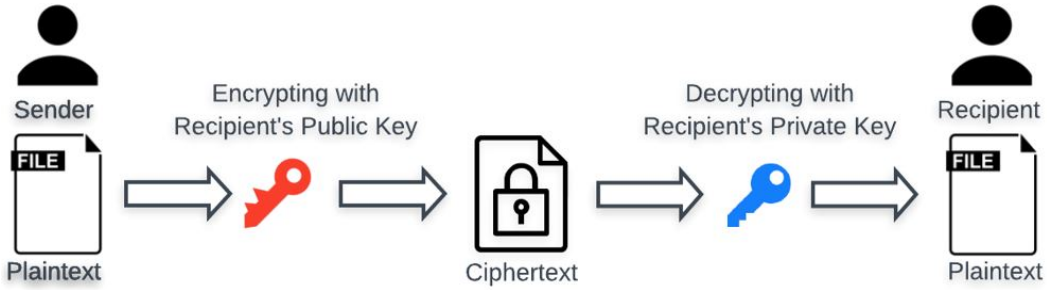


Figure 2.4. Public-key Cryptography

A secure hash function (e.g., Message Digest 5 (MD5), Secure Hash Algorithm 3 (SHA-3)) is a one-way mathematical function that creates a fixed size "fingerprint" from an arbitrary size data input [16]. The output of a hash function is called a message digest. Message digests are used to provide integrity (data was not modified in transit), but by themselves, do not offer confidentiality or non-repudiation as a malicious actor can intercept a sender's message and replace it with a modified message and its respective digest [16]. To provide authenticity, a message authentication code (MAC) tag can be generated by computing an algorithm with the message and a shared secret key as input [25]. The sender sends the original message with the MAC tag and the recipients compute their own MAC tag from the original message and the shared secret key in order to verify that

the tags match [25]. A variation of a MAC called a hash-based message authentication code (HMAC) uses secure hash functions as the computed algorithm (see Figure 2.5).

Asymmetric (public-key) algorithms (e.g., Diffie-Hellman, Rivest-Shamir-Adleman (RSA), elliptic-curve cryptography (ECC)) address the shortcomings of key management by introducing two keys: a public key and a private key [16]. Only the public key is published and the private key is kept secret [16]. Public-key algorithms are not as useful for encrypting large messages (compared to symmetric keys) due to their computational cost [16]. Rather, they are specifically used to provide authentication, non-repudiation, integrity through digital signatures and confidentiality through key management [16]. In asymmetric cryptography, any message may be encrypted with a public key but only the authorized recipient can decrypt the message using their respective private key (confidentiality) [16].

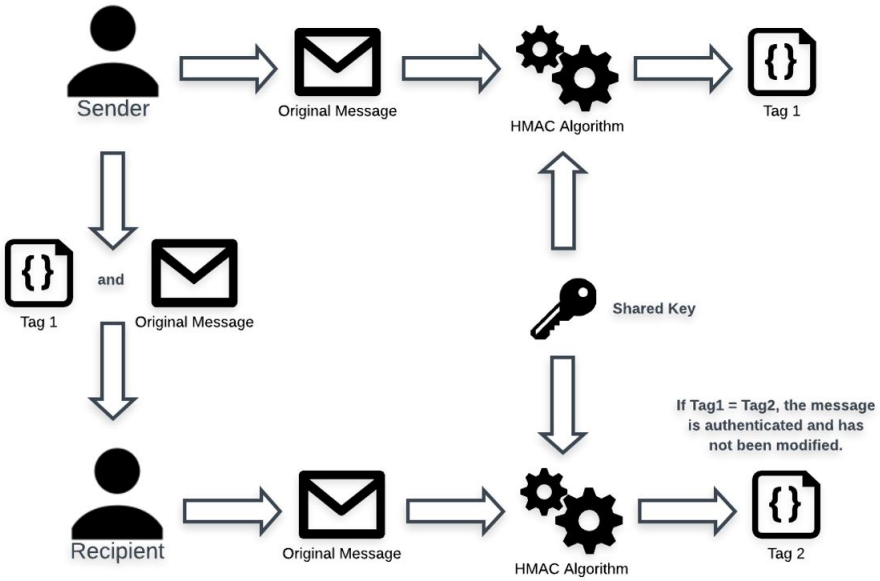


Figure 2.5. Authentication and Integrity with HMAC. Adapted from [25].

2.1.3 Point-to-Point Authentication

The Point-to-Point Protocol (PPP) defines the data link layer authentication standard that allows for remote client authentication over point-to-point (P2P) links (e.g., dial-up, circuit-switched) using protocols such as Password Authentication Protocol (PAP) and Challenge Handshake Authentication Protocol (CHAP) [26].

PAP is the most basic password-based authentication protocol [27]. PAP requires the subscriber to submit a username and password pair to the authenticator until either authentication occurs or the network connection is terminated [27]. However, this authentication scheme suffers from a wide array of vulnerabilities (e.g., MITM) due to the fact that PAP sends passwords in plaintext [27].

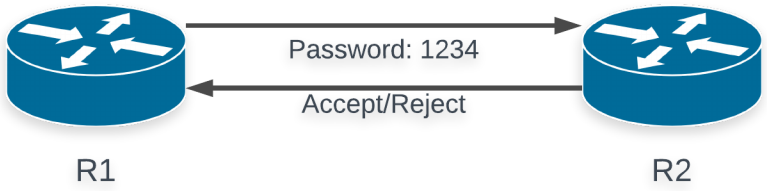


Figure 2.6. PAP (Two-Way Handshake)

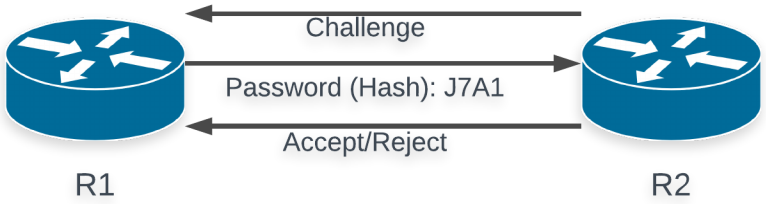


Figure 2.7. CHAP (Three-Way Handshake)

CHAP is known as a challenge-and-response protocol and uses a three-way handshake [28]. In this scheme, the authenticator transmits a "challenge" message to the user who responds with the hash (also called message digest) of the password [28]. The authenticator verifies the hash value using its own database of expected hash values [28]. If the hash values match, the subscriber is authenticated. Upon initial success, CHAP continues to conduct random periodic verifications [28]. While the plaintext password is never sent over the network, both client and server share the same password in plaintext [28]. Since the passwords are stored on both the client and server side, CHAP is not efficient for large organizations where storage resources are limited [28]. CHAP was designed to protect against replay attacks by incrementally changing the identifier and varying the challenge value which is used along with the password to compute the hash value [28].

Extensible Authentication Protocol (EAP) arose from the need for an open standard authentication framework as network connectivity outgrew traditional P2P links with the advent of the Institute of Electrical and Electronics Engineers (IEEE) 802 standard [29]. EAP is unique from PAP and CHAP in that it is not an authentication protocol [29]. Rather, EAP is a general protocol that allows for the selection and transport of open standard or proprietary third-party authentication mechanisms (e.g., EAP-MD5, LEAP, EAPoL, EAP-TLS, EAP-IKEv2) between the client (also called supplicant) and authenticator through the use of a back-end authentication server (AS) [29]. The back-end AS is typically a server that runs on Authentication, Authorization, and Accounting (AAA) protocols which support EAP such as Remote Authentication Dial-In User Service (RADIUS) or Diameter [29]. AAA protocols will be described in the following subsection. The capability to support different authentication schemes allows EAP-compatible Network Access Server (NAS) devices (e.g., switches or access points) the flexibility to provide remote client authentication over dedicated, switched circuited, wired, and wireless links while serving in two simultaneous capacities: an authenticator for local clients (assuming that the NAS supports the specified authentication method) or a pass-through agent for non-local clients (NAS requires no knowledge of the authentication method) [29]. Figure 2.8 shows a high-level diagram of EAP authenticating to a RADIUS AS over a local area network (LAN) authenticator.

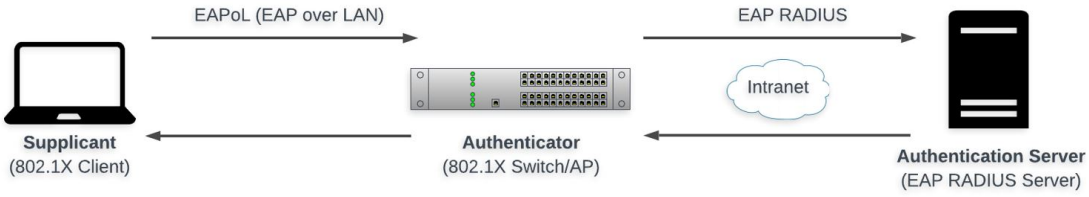


Figure 2.8. EAPoL via Authenticator to an EAP RADIUS AS. Adapted from [29].

2.1.4 Client-Server Authentication

PPP does not support device scalability, is vulnerable to physical tampering, and makes it difficult for network administrators to centralize logging and auditing [30]. As the Internet grew, the limitations of PPP led to the development of AAA protocols as researchers realized it was no longer effective nor efficient to manage access control over individual NAS devices.

Leveraging the IEEE 802.1X standard for port-based Network Access Control (PNAC), AAA provides a client-server model that allows end users to authenticate via an authenticator to a centralized server which stores all the credentials and information required for access management. Commonly used AAA protocols include RADIUS, Diameter, and Terminal Access Controller Access-Control System Plus (TACAS+). Although it is not an AAA protocol, Kerberos is another popular network authentication protocol that will be described in this subsection.

RADIUS was initially conceived for authenticating dial-up network users but has since become a general-purpose authentication protocol for wired and wireless users attempting to gain network access [31]. When RADIUS was first being developed, servers were single threaded meaning that individual requests were processed sequentially [31]. This was not conducive to scalability and would have resulted in long server queues for servers that saw hundreds of users per minute [31]. Thus, developers needed a lightweight multi-threaded solution and chose User Datagram Protocol (UDP) as the transport protocol due to its connectionless, low overhead features [31]. This greatly simplified server implementation and improved authentication response times over the connection-oriented Transmission Control Protocol (TCP) [31]. To improve reliability, RADIUS artificially incorporates its own retransmission timer [31]. Lastly, RADIUS only encrypts the segment of the data packet containing the user authentication credentials vice the entire packet, making it vulnerable to replay attacks [32].

Diameter was developed as the technological demands, complexity, and scale of AAA networks grew with the introduction of broadband, high-speed wireless, and new applications such as Voice over IP (VoIP) [32]. Additionally, developers of Diameter sought to overcome the inherent flaws of RADIUS due to its use of UDP. Diameter supports both TCP and Streaming Control Transmission Protocol (SCTP) which enables whole packet encryption via Transport Layer Security (TLS) or Datagram Transport Layer Security (DTLS), respectively [32]. The reliability provided by TCP and SCTP especially benefited accounting services where packet loss had the potential for lost revenue. Other features that were previously not supported by RADIUS included support for agents (e.g.,proxies, redirects, relays), support for server-initiated messages, error notifications, capability negotiations between clients and servers, and dynamic discovery of peers [32]. Diameter was also designed to be backwards compatible with legacy networks running RADIUS [32].

TACAS+ is proprietary Cisco protocol for access control and the most up-to-date version of Extended TACAS (XTACAS) and the Internet Engineering Task Force (IETF) standard of TACAS [30]. The original TACAS is one of the oldest authentication protocols as it was first designed to authenticate dial-up links for the Advanced Research Projects Agency Network (ARPANET), the precursor to the Internet [33]. TACAS+ performs the same functions of RADIUS and Diameter with an added feature when using Cisco devices. TACAS+ allows enterprises the ability to manage network administrators through command authorization [30]. Command authorization is useful for large organizations that have multiple administrators accessing different segments of the network as it can be used to restrict administrator commands input [30]. Just like Diameter, TACAS+ also supports TCP and whole packet encryption between the AS and NAS device [30].

Kerberos, developed by researchers at Massachusetts Institute of Technology (MIT), is another popular client-server network authentication protocol that provides mutual authentication through symmetric-key cryptography (see Figure 2.9) [34]. In Kerberos, the client is verified through a trusted-third party known as the KDC which contains two servers: an AS and a Ticket Granting Server (TGS) [35]. The client encrypts the access request with their own password and sends it to the AS [35]. The AS matches the user identification to the client's password stored in a database and uses the client's password to decrypt the client's request [35]. Once the client is verified, the AS sends a Ticket Granting Ticket (TGT) to the client which is encrypted with a second secret key that it shares with the TGS [35]. The client sends the TGT back to the TGS along with its initial request [35]. Once the TGS receives the TGT, it decrypts the TGT using the second secret key and then issues a short-lived ticket (also called session key) encrypted with a third secret key that it shares with the application server [35]. The client can now send the ticket to the application server who decrypts and verifies the ticket with the secret key which it shares with the TGT [35]. Unlike TACAS+, Kerberos is open-source and based entirely on open Internet standards; this allows for public scrutiny and thus, continual updates to the protocol.

2.1.5 Certificate-based Authentication

Similar to a handwritten signature, digital signatures leverage public-key cryptography to prove the sender's identity, making it difficult for the sender to deny that a message originated from them [16]. The sender's private key encrypts the message, which generates

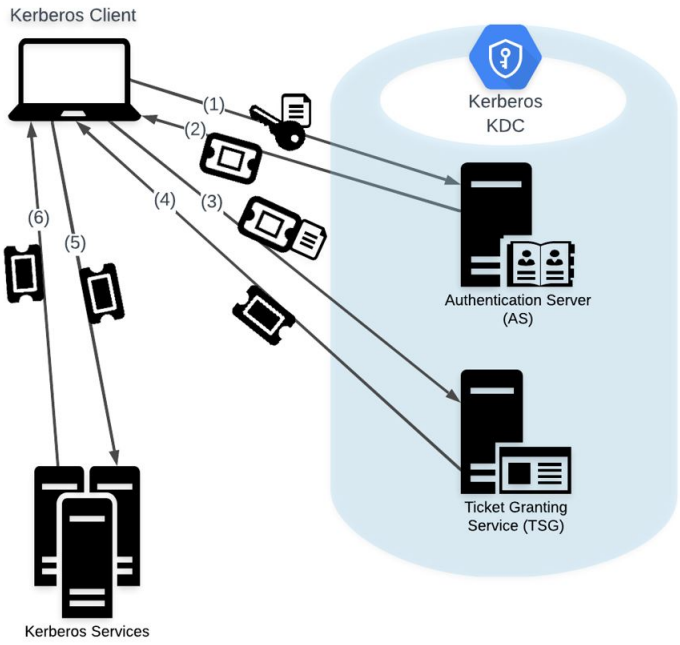


Figure 2.9. Kerberos Authentication Protocol Diagram. Adapted from [36].

a digital signature. To authenticate the sender, the recipient generates his/her own digest from the original message, decrypts the sender's signature with the sender's public key to produce another digest, and then compares the two digests to verify that they match (see Figure 2.10) [16]. Thus, the sender cannot deny that he/she generated the signature (non-repudiation) and as long as the digests match, the data was not modified (integrity) [16].

Just as a passport or driver's license is used to prove an individual's identity, a digital certificate (also called public-key certificate) is an electronic document that guarantees, against impersonation, the identity of an individual, server, organization, etc [37]. Request for Comments (RFC) 5280 details the X.509 framework which defines the current public-key certificate standard for use across the Internet [38]. Figure 2.11 displays the X.509 syntax for each certificate version. Like OTPs, digital certificates are another example of possession-based authentication with an additional authentication factor: the private key is "something you have" and the password that protects the private key is "something you know" [37]. With regards to passport authenticity, your picture along with other personal information is used to verify your identity and the passport is issued and endorsed by the Department of State upon application approval. This begs the question: how do we ensure the authenticity of digital certificates? The proof of validity for a certificate requires the

digital signature of a trusted-third party [37].

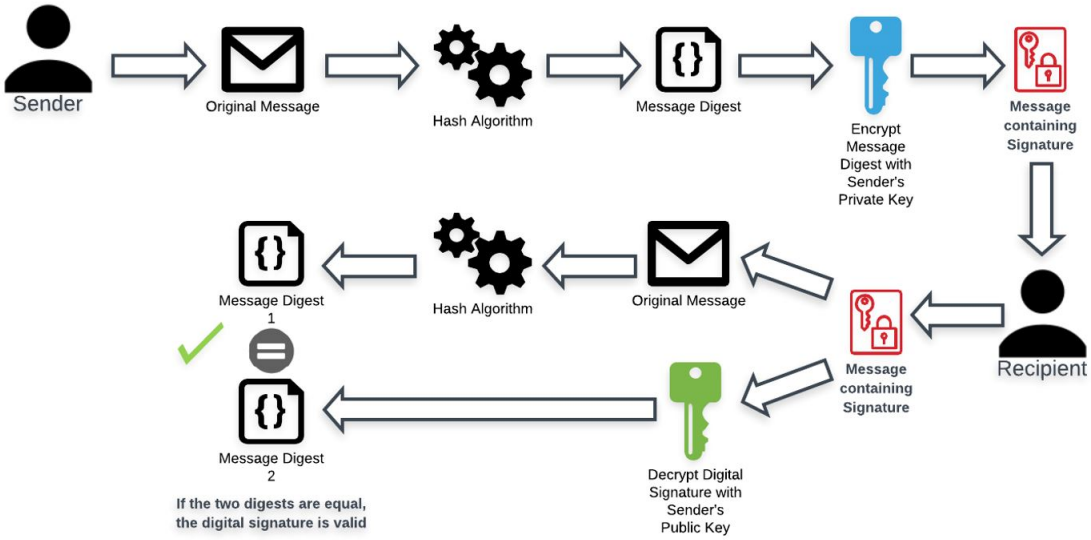


Figure 2.10. Signing and Verification of a Digital Signature

2.1.6 DoD Public Key Infrastructure

The advent of public-key algorithms solved the limitations of symmetric-key distribution and management in providing scalability, authentication, and non-repudiation through a framework called PKI [16]. PKI connects trusted-third party across widely distributed organizational boundaries, thus allowing the secure distribution of public keys [16]. Since the public key are managed by the trusted-third party and also bound to a particular user, the private key possessed by the users ensure that integrity, authentication, and non-repudiation are maintained through digital signatures [16]. The secure management of private keys is vital to the security of PKI. There are two primary ways that private keys can be stored: locally on the client's device or through a remote hardware device such as a Universal Serial Bus (USB) token or smart card [39]. PKI contains the following functional components:

1. A **CA**. The CA is a trusted-third party or entity trusted by all participants in the network to perform certification responsibilities for achieving public-key authentication [40]. The primary services that the CA provides are: certificate issuing (e.g., create and sign certificates), certificate renewal (e.g., publish unexpired certificates), certificate revocation (e.g., maintains certificate revocation lists (CRLs), and certifi-

cation storage (e.g., maintain archive of expired certificates issued by the respective CA) [16], [40]. In general, once trust is established, a CA can issue or revocate certificates for users and other CAs by signing with its own private key [16]. Likewise, users and other CAs can verify the authenticity of the CA in question via its respective public key [16]. The principal top-level CA in a PKI is known as the root CA [16]. Since there is no higher trusted entity to verify the certificate of the root CA, it issues its own "self-signed" certificate and publishes its own public key to assert trust [16]. The safeguarding of a CA's private key is paramount and represents a vulnerability in PKI that must not be compromised [16].

2. A **registration authority (RA)**. The RA – a collection of hardware, software, and/or human operators – is trusted by the CA to register and authenticate the identity of PKI users [16]. CAs maintain their own list of RAs and mutually authenticate each other with respective public and private keys [16]. Similar to the CA, an RA's private key must also be heavily guarded [16].

3. A **repository**. The repository is a database that provides a means to store, distribute, manage, and update the status of digital certificates [16]. The CAs submits active certificates and an updated list of revoked certificates, known as a CRL, to the repository servers [16].

4. An **archive**. The archive is a database that stores and maintains a record of CA issued certificates and associating certificate-related information that could be used to determine the validity of a digital signature for future dispute resolution [16].

Given the vulnerabilities of password-based authentication, the DoD viewed PKI technology and public cryptography as critical elements of the DoD IA Defense-in-Depth strategy for enterprise imperatives as well as warfighting [41]. In August of 1997, the Deputy Secretary of Defense (DEPSECDEF) released a memorandum soliciting input from the Under Secretaries of Defense in an effort to identify requirements for the design, development, and implementation of a DoD PKI to provide integrity, authentication, non-repudiation, and confidentiality for all programs and applications on DoD networks [42]. As stated in the memorandum:

> The Department of Defense is taking major steps in reforming its paper-based processes. It is our plan to move from traditional paper-based processes into an environment where data is moved electronically between users. As part of this effort, we have developed a position paper for the Department on digital signatures and commercial practices that I want to share with you. Jointly developed by my office and the Assistant Secretary of Defense for Command, Control, Communications, Computers and Intelligence (ASD(C3I)), the Defense Information Systems Agency (DISA) and the National Security Agency (NSA), this document serves to identify the baseline for the Department's transition to a paperless environment. [42]

In May of 1999, the DEPSECDEF released another memorandum detailing the DoD policies for the development and implementation of a DoD-wide PKI that was later updated by the DoD CIO in August of 2000 in order to better align milestones with the "Smart Card Adoption and Implementation" memorandum, released in November of 1999 [41]. In November of 2000, the DON published its PKI implementation plan, providing a roadmap for Navy and Marine Corps planners to execute the DoD PKI policy [43]. The initiation of smart card deployment and its supporting infrastructure under DoD PKI began in 2004 with the issuance of Homeland Security Presidential Directive-12 (HSPD-12) which dictated the use of a shared identification standard for promoting interoperability amongst Federal authentication mechanisms [44]. The following year, Federal Information Processing Standard (FIPS) 201 was published which defined the Federal smart card credentialing standard known as the Personal Identity Verification (PIV) Card (known as the CAC within the DoD) [44]. The emergence of PKI within the DON saw aggressive, widespread adoption; by 2010, approximately 85% of users under the DON had migrated to authentication using the PKI-enabled CAC [45]. In an effort to better align with the PKI standards of the Federal Government and to streamline operational interoperability, a DoD memorandum was published in February of 2019 directing the replacement of the DoD Identity certificate with the DoD PIV authentication certificate by May of 2020 [46].

2.1.7 Common Access Card

The DoD CAC leverages smart card technology in order to provide authentication (digital signature), data integrity (digital signature), confidentiality (encryption), and non-

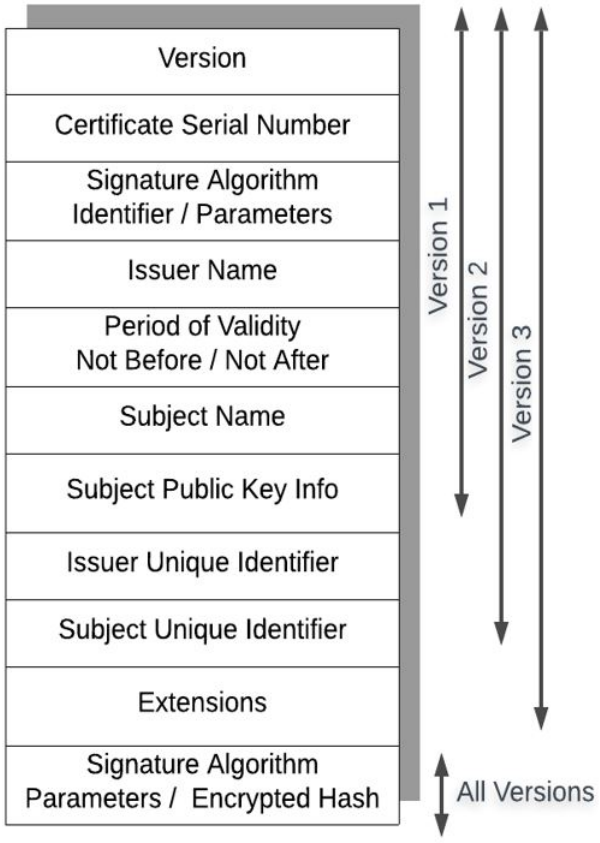


Figure 2.11. X.509 Certificate Fields. Adapted from [38].

repudiation (digital signature) [47]. In addition, the CAC serves as the standard identification card for access control to physical spaces (e.g., U.S. military bases, buildings, controlled areas) and DoD networks and systems. [48], [49]. CACs provide a strong 2FA mechanism requiring something that you know, a Personal Identification Number (PIN), and something that you have, physical possession of the CAC. CACs are issued and maintained by local RA infrastructures using the DoD's Defense Enrollment Eligibility Reporting System (DEERS) personnel database for identity verification and Defense Manpower Data Center (DMDC)/Real-time Automated Personnel Identification System (RAPIDS) for processing/credentialing [49]. CACs are issued to uniformed members of the Armed Forces, U.S. Public Health Service, National Oceanic and Atmospheric Administration (NOAA), DoD Civilians, and DoD Contractors [48].

The CAC possesses a 32-bit reduced instruction set computer (RISC) processor and

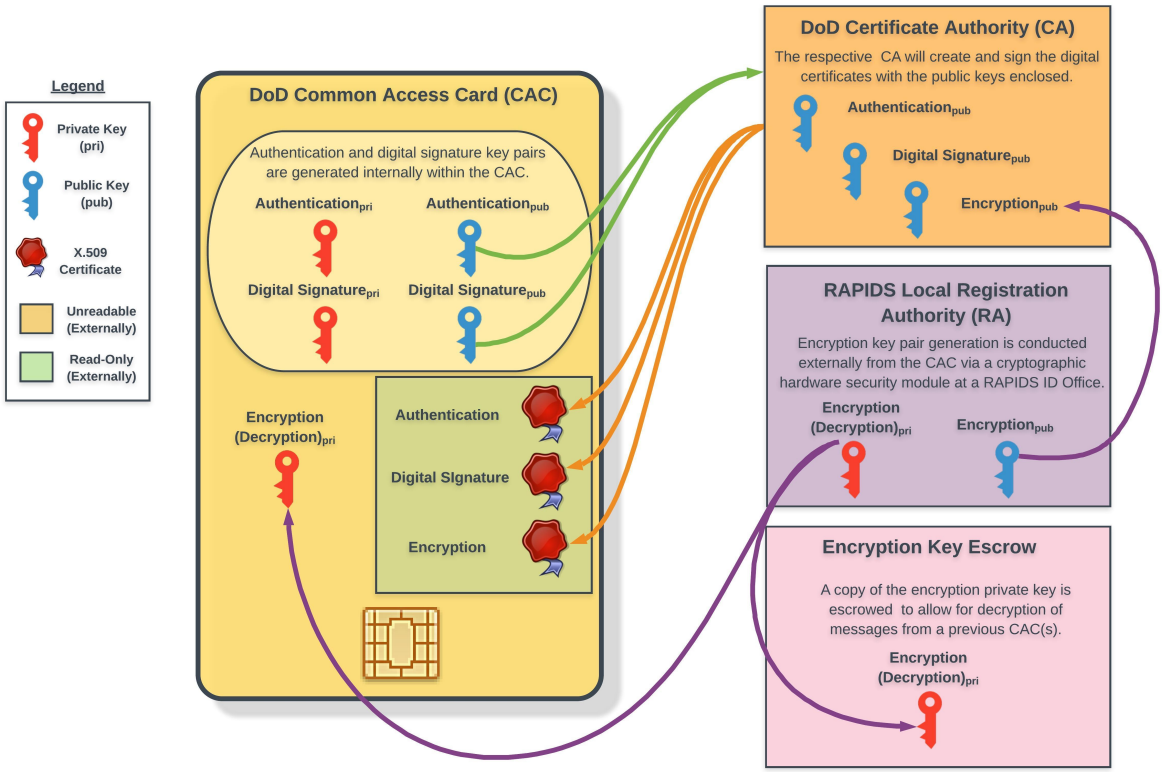
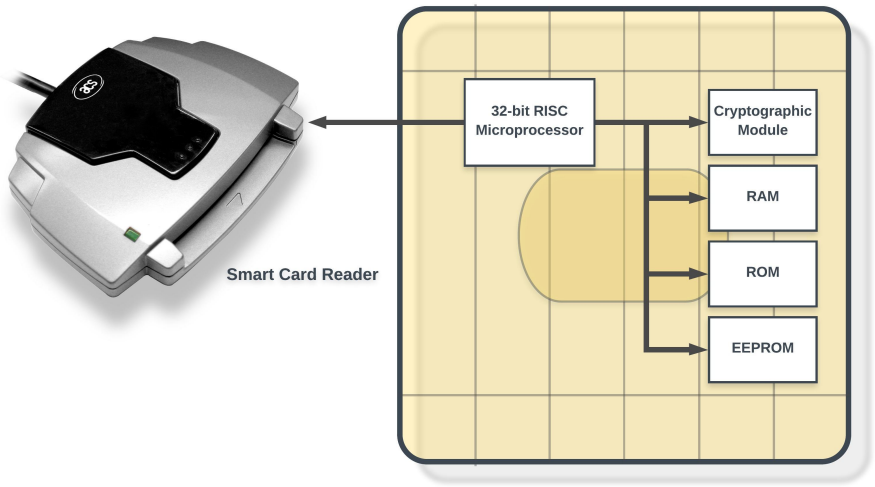


Figure 2.12. CAC Enrollment under DoD PKI. Adapted from [50].

additional cryptographic hardware that allows it to perform public-private key generation for the purpose of supporting PKI operations [47]. The CAC is capable of storing 144K of data on its tamper-resistant integrated circuit chip (ICC) which typically contains the client's RSA public-private key pairs for identity and/or PIV authentication, digital signature and encryption as well as their respective digital certificates among other personal/administrative information [48]. Figure 2.13 shows the CAC ICC hardware architecture which has been adapted from [51]. To further guard against host devices (which, by default, are assumed to be untrusted), the internal components of the CAC are designed such that the client's private keys are unable to be viewed or obtained [47]. As such, all PKI challenge-response operations to verify the validity of the private keys are performed within the CAC making it impossible for the host device to possess any knowledge of the private keys [47]. As an added level of physical security, the PIN protects the CAC in the event of theft [47]. The authentication and digital signature key pairs are generated internally within the CAC. The e-mail encryption key pair, however, is generated externally at a RAPIDS office using a

DoD-approved hardware security module (HSM). The resulting private key is imported onto the client's CAC and copied/escrowed to allow for future retrieval in order to allow for decryption of messages from previous a CAC(s). The client's public keys are sent to the respective CA who verifies the client's identity. Once the client's identity is validated, the CA encrypts the client's information and the client's public key with its own private key to create a digitally signed certificate that is then stored in a certificate database, archived, and imported onto the client's CAC.



CACs are passive devices which require a smart card reader to provide power and an interface for half-duplex communication through a serial input/output (I/O) interface [52]. The CAC employs a black-box model where an input is fed through the card reader, processed by the CAC's microprocessor, and the resulting output is returned to the card reader [52]. The memory mapping unit (MMU) of the microprocessor's operating system (OS) creates a hardware-software firewall that prevents user (application-layer) processes from viewing or accessing the private keys stored on the CAC. Thus, the CAC is considered a secure storage device because it prevents the card reader from having direct access to the CAC's protected memory spaces [52].

Figure 2.13. Simplified CAC ICC Architecture

2.1.8 Mobile Device Management

The rise of mobile technology has been recognized as a key capability enabler for joint force combat operations [53]. Globally integrated operations, secure and non-secure com-

munications, and cloud-enabled C2 can all leverage the application of mobile technology to greatly increase collaboration and the dissemination of information [53]. Given the surge in DoD CMDs, the implementation of a consolidated Mobile Device Management (MDM) architecture and Mobile Application Store (MAS) at the DoD enterprise level is necessary to provide secure mobile device operation and maintenance in cost-effective manner [53]. The DoD's MDM and MAS architecture seeks to minimize duplication, cost, and downtime by providing enterprise management for the following [53]:

- Policy enforcement for end user devices through the establishment of end user permissions for approved functions at the user and application-level
- Malware detection
- Over-the-Air (OTA) software distribution
- Remote data wiping
- Remote device configuration management
- Asset/property management for data and key protection
- Distribution, update, and deletion of mobile applications

The DoD's MDM architecture is a decentralized system hosted across several Defense Information Systems Agency (DISA) Defense Enterprise Computing Centers (DECCs) and accessed through a web portal for those with administrative profiles [53]. While the Bring Your Own Device (BYOD) or personally owned devices trend has seen rapid adoption across the commercial industry for business purposes, the current DoD policy prohibits the use of BYOD [53]. As stated by the DoD CIO, Teri Takai, in her CMD implementation plan memorandum,

> Despite the benefits, existing DoD policies, operational constructs, and security vulnerabilities currently prevent the adoption of devices that are unapproved and procured outside of official government acquisition. [53]

Simply said, the assumption of risk for BYODs exceeds the threshold which the DoD is willing to assume, in part primarily due to the inability to implement MDM in personal mobile devices [54]. Without MDM, the DoD is unable to implement any of the configuration controls required for ensuring mobile device security and preventing data breaches (i.e., remote wiping). Because sensitive information, such as private keys, are stored on

CMDs, MDM also allows the capability to sandbox applications (using virtual machines to partition the memory of mobile devices) such that only approved applications have access to the requested data [54].

2.1.9 Mobile Derived Credentials

Prior to 2016, DoD CMDs lacked a true capability to conduct secure e-mail and web browsing due to the hardware challenges associated with tethering a smartphone to a smart card [55]. Smart card readers with Bluetooth technology were tested for a period of time but found to be too cumbersome and costly [55]. MicroSDs SmartCard-HSMs were also tested but proved to be limited in functionality since not all leading mobile platforms supported SD cards [55]. The need for smart card reading infrastructure also meant that organizations would have to incur additional operating costs since smart card readers were not typically bundled with device purchases. Overall, existing hardware tethering solutions were found to be inadequate; in addition, they did not provide a positive experience for end users [55]. Advances in mobile telecommunications have paved the way for radio frequency (RF) communications with smart cards over a contactless surface using Near Field Communication (NFC) technology [44]. As implied, the user would be required to place the smart card in close proximity to the NFC-enabled device, but existing technologies and standards had not made it practical nor possible for use [44]. This set in motion the DoD's motivation to develop and implement a set standard for providing National Information Assurance Partnership (NIAP) certified U.S. Government CMDs a capability to securely store mobile credentials without the use of smart card readers [55].

In September of 2014, the DoD CIO issued a memorandum directing the DoD PKI Project Management Office (PMO) to conduct research in order to design an enterprise service for implementing derived PKI credentials on unclassified CMDs [55]. A derived credential, as defined in NIST Special Publication 800-63-3, is an alternative to CAC-based authentication designed specifically for mobile devices and is

> A credential issued based on proof of possession and control of an authenticator associated with a previously issued credential, so as not to duplicate the identity proofing process. [56]

Less than a year later, on October of 2015, the NSA and DISA hosted an Industry Day in which the initial requirements and capabilities for the Purebred program were announced [55]. In August of 2016, Purebred reached initial production capability with support for user encryption key recovery following in December of that year [55]. What is the Purebred program?

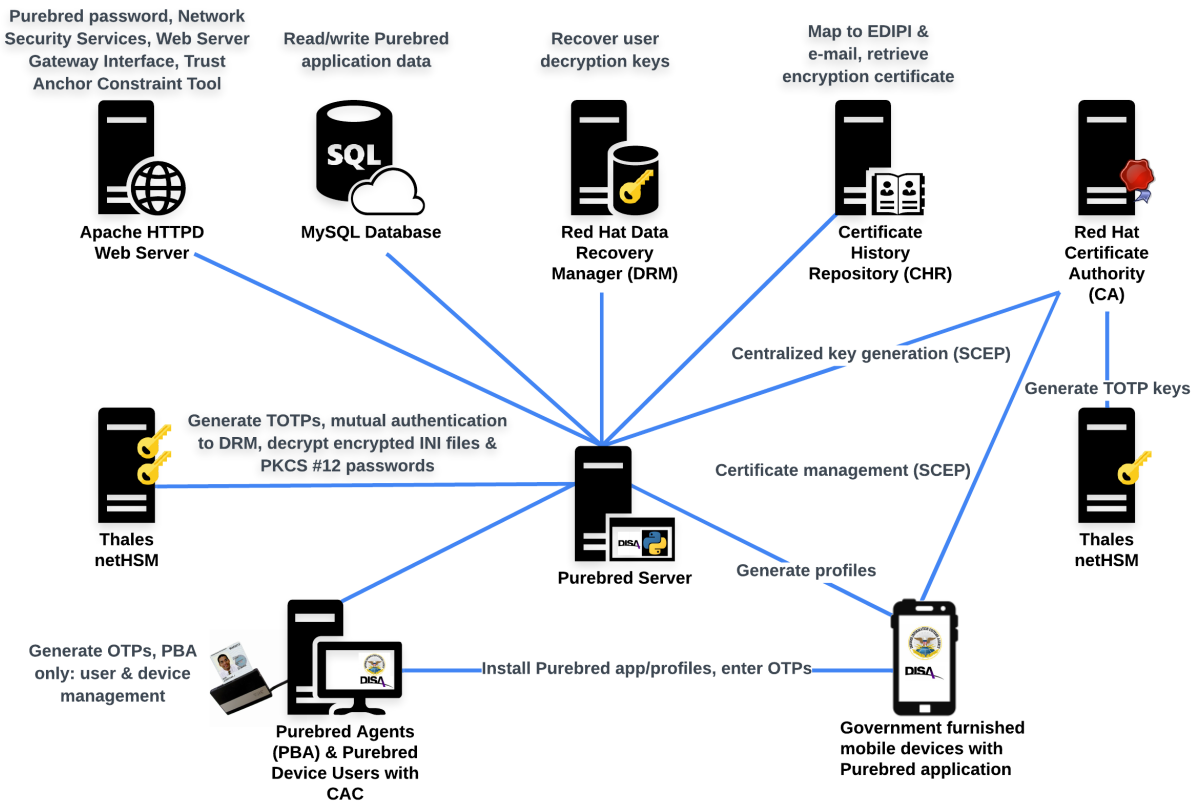
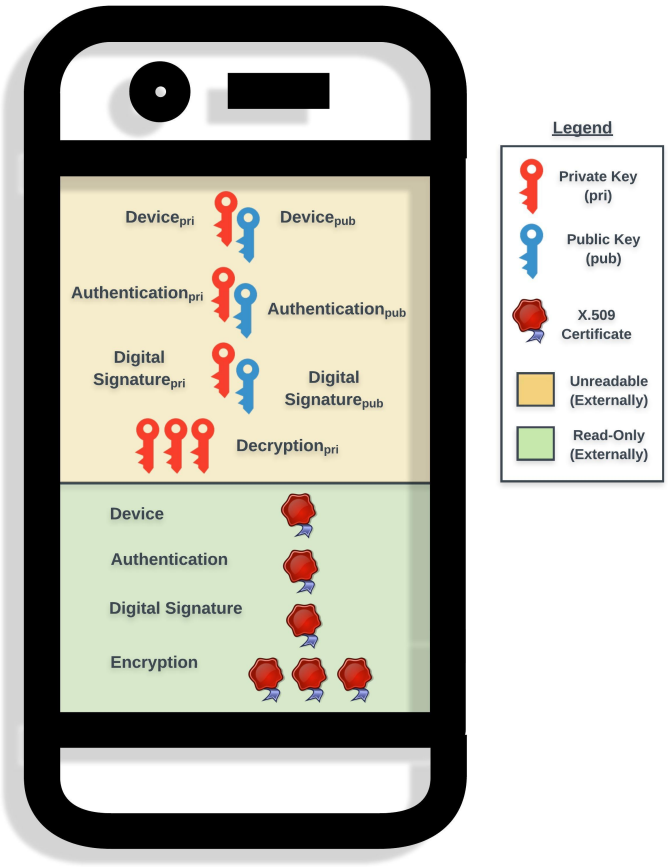


Figure 2.14. System Architecture for Purebred. Adapted from [57].

Using Apple's OTA Profile Configuration and Delivery Protocol with Key Management Extensions (KMEs) to distribute Simple Certificate Enrollment Protocol (SCEP) and Public Key Cryptography Standards (PKCS) #12 payloads, Purebred provides derived credentialing for clients on unclassified mobile platforms to access remote content, sign encrypted e-mails, and securely browse the Internet without the use of a CAC [8]. The derived credentials are escrowed private keys and X.509 public key certificates that maintain the same expiration datetime as their respective CAC certificates, either generated from a

DoD CA that supports SCEP (i.e., authentication certificate, digital signature certificate) or retrieved from a certificate repository (i.e., e-mail encryption certificate) [8].



Authentication and digital signature key pairs are generated internally within the device. In the current Purebred configuration, up to (3) decryption keys may be retrieved from the Red Hat Data Recovery Manager. The respective e-mail encryption certificate(s) are retrieved from the Certificate History Repository [57].

Figure 2.15. Purebred Mobile Device Credentials

SCEP is utilized as the backbone PKI communications protocol for certificate management (i.e., certificate requests, CRL queries, certificate renewals) while PKCS #12 defines the storage syntax for the files containing personal identity information which include private keys and digital certificates [58], [59]. Purebred is comprised of a key management server along with a host of application that supports both smartphones and tablets for all major platforms (e.g., iOS, Android, Microsoft, Blackberry) to include the USB platform, YubiKey [55].

Another key breakthrough of Purebred is its ability to separate key management from device management, an issue which mobile ecosystems have struggled to decouple [55]. Maintaining separate management structures allows the enterprise PKI system to remain centralized while enabling the decentralizing of mobile instances. Since device management varies according to the operational scenario, it was DISA's intent to maintain vendor neutrality in order to support the varied use cases of each service and agency [55].

The credentialing workflow for Purebred is typically a one-time enrollment process, similar to how often a new CAC is generated, unless something unexpected happens to the credentialed mobile device [55]. The credentialing process begins either in-person or remotely via a Purebred Agent (PBA), a trusted DoD PKI administrator that has been granted the ability to credential and enroll mobile devices [55]. Though the mobile device are Government Furnished Equipment (GFE), additional processes are required to cryptographically verify and establish trust [55].

First, the mobile device generates a RSA key pair to create a temporary self-signed device certificate that is submitted to the Purebred portal (a Wi-Fi connection is required) [8]. The PBA's electronic data interchange personal identifier (EDIPI) is required to link the enrollment session of the user to the respective PBA [8]. The PBA then generates a pre-enrollment OTP using the PBA's EDIPI and the unique identifier of the device on the Purebred server which is entered into the Purebred Registration application to prevent spam by securing the new device's public key [8]. The OTP generator and validator for the entire enrollment process leverages a NIST SP800-108 key derivation function (KDF) to generate time-based one-time passwords (TOTPs) [8]. The KDF uses two symmetric keys which are stored in a HSM: the first key is associated to human viewable OTP values while the second key uses OTP values from the SCEP payloads which are not viewable by humans and encrypted using server-authenticated TLS [8].

Next, the PBA compares the certificate hash of the device serial number on the Purebred database to the hash displayed on the mobile device, this ensures the integration of human verification to prevent MITM attacks or impersonations [55]. The PBA authorizes the device enrollment by generating and submitting an enrollment OTP along with the client's digital signature via SCEP which then allows the Purebred CA to credential the device with a new device certificate under DoD PKI [8]. Figure 2.16 and Figure 2.17 show the

enrollment and credentialing process as displayed on the Purebred application.

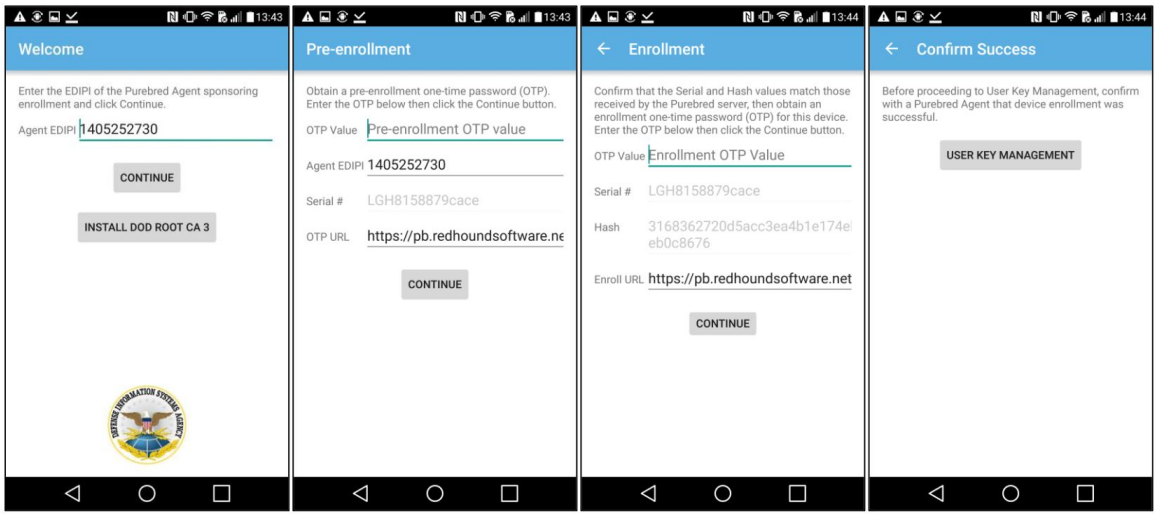


Figure 2.16. Enrolling a DoD Mobile Device with Purebred. Source: [55].

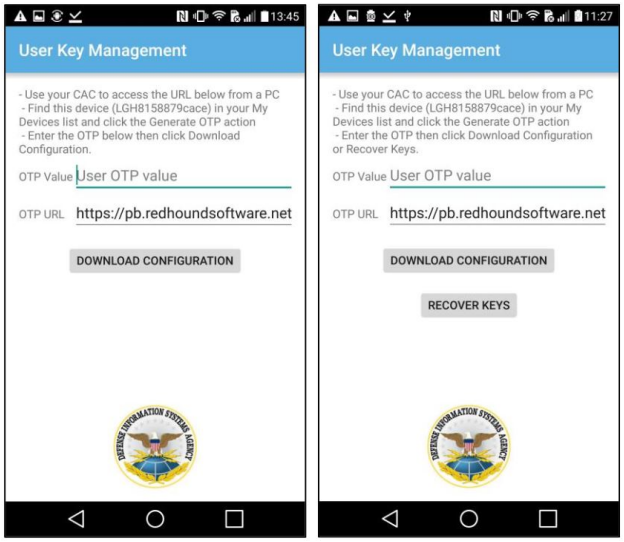


Figure 2.17. Importing Purebred Derived Credentials. Source: [55].

Once the device is enrolled, the user can then generate their own OTPs on the Purebred portal [8]. The OTPs serve as a challenge password for authentication in the SCEP protocol allowing the retrieval of the client's identity/PIV authentication certificate, digital signature certificate and e-mail encryption certificate(s) as well as the corresponding decryption key(s) (up to three) [55] [8].

2.1.10 Purebred Key Management

While Purebred can support both centralized and distributed (on-device) key generation, the RSA key pairs in practice are generated and stored internally within the mobile device using platform-specific application programming interfaces (APIs) [8]. See Table 2.1 [57]. The keys may be protected by vendor-specific hardware components (i.e., key attestation) and/or software depending on the mobile platform [8]. Key attestation is a way for the mobile device to prove to the CA that the asymmetric key it possesses is stored in secure hardware (i.e., Trusted Execution Environment (TEE), Secure Element (SE)) trusted by the CA [60]. Software protection is typically implemented by storing the encrypted keys in containers isolated from application processes, allowing only system processes to carry out the challenge-response protocol required for cryptographic operations. This ensures that it is impossible for application processes to extract or view the keymat.

Table 2.1. Purebred Platform-specific Key Generation

Platform	Key Generation
iOS	Software: Apple KeyChain API
Android	Software: Android Keystore API
Microsoft	Hardware: Universal Windows Platform APIs on Surface Pro; Yubikey using Yubikey APIs
Blackberry	Software: OpenSSL

Adapted from [57].

Though Purebred can support hardware-back mechanisms for key storage, it is not currently being utilized due to the lack of a CA capable of issuing hardware derived certificates and the lack of MDM support (i.e., for Android, the MDM application claims the one and only DeviceOwner function leaving Purebred without a delegated certificate installer feature) [57]. Decryption private keys are securely imported from a key escrow system via a mutually authenticated TLS connection since the same decryption key used on the mobile device is the same key used on CAC-enabled computer workstations or laptops [8].

2.2 Mobile Ad-hoc Networks

Advances in networking and portable computing have produced distinct types of wireless networks classified according to two primary criteria: the number of hops it takes a

packet to travel across the wireless network (one or multiple hops) and whether wireless network requires existing infrastructure (i.e., base station) to operate [61]. Table 2.2 shows examples of wireless networks based on this set of criteria.

An emerging class of wireless networks exists which does not rely on pre-existing infrastructure but rather on a collection of mobile nodes which act as independent routers within the network to transmit data to the final destination [61]. This class of networks, which can form rapidly changing self-healing topologies, is known as a MANET. MANETs have numerous applications in the civilian sector (i.e., first responders, IoTs, personal area networks (PANs)) but its autonomous behavior, small footprint, and decentralized management make it particularly beneficial for military use at the tactical edge where speed/tempo and mobility is desired.

Table 2.2. Classification of Wireless Networks

Network Classification	**Example(s)**
Single-hop, Infrastructure-based	WiFi, 3G
Single-hop, Infrastructure-less	Bluetooth
Multi-hop, Infrastructure-based	Sensor/mesh networks
Multi-hop, Infrastructure-less	Mobile ad-hoc networks

Adapted From [61].

Independent of conventional structure, MANETs are advantageous in disaster areas and combat zones where backbone infrastructure or access points are non-existent, destroyed, or impractical [62]. In addition, the dynamic routing capabilities of MANETs can be a vital enabler for extending line-of-sight (LOS) and beyond line-of-sight (BLOS) communications (via airborne methods such as Unmanned Aerial Vehicles (UAVs), Battlefield Airborne Communications Node (BACN)-equipped aircraft, communications satellites) in situations where traditional terrestrial communications may be degraded due to factors such as inclement weather or obstructions in the terrain. Figure 2.18 illustrates how the Adaptive Networking Wideband Waveform (ANW2) MANET can be extended BLOS.

Despite their advantages and applications, MANETs are still faced with their own set of challenges and planning considerations when it comes to employment, including:

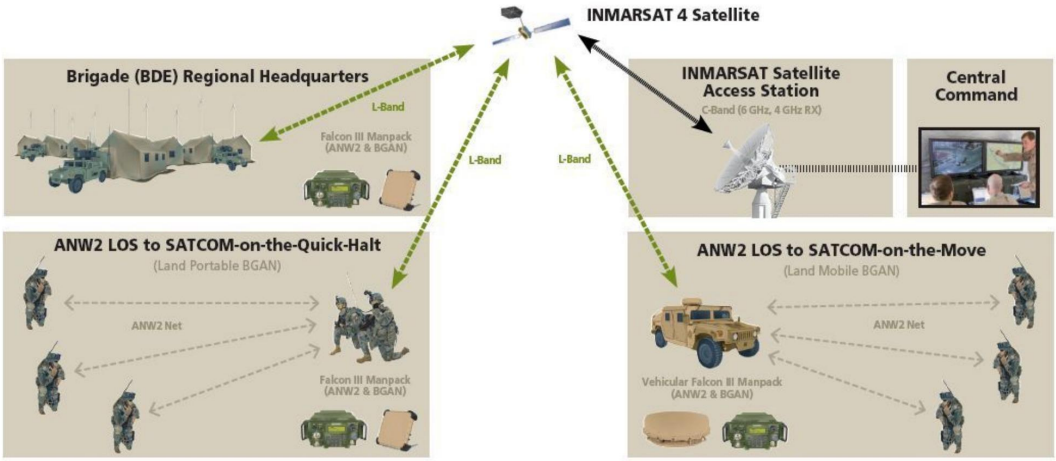


Figure 2.18. Extending ANW2 across multiple sites via the Inmarsat Broadband Global Area Network (BGAN) SATCOM Terminal. Source: [63].

- **Network Management:** no centralized trusted authority to administrate/monitor network operations, lack of network management framework
- **Resource Constraints:** nodes have limited computing capability, data storage, battery capacity and as a result, limited transmission power/range
- **Wireless Constraints:** by default, lower bandwidth than wired links, signal degradation due to obstructions/interference in wireless medium, longer delays, high bit error rate (BER)/packet loss, broadcast nature of MANETs means nodes in range of sending node will receive the transmission
- **Mobility:** constantly changing topologies, discovery/configuration of new nodes, high frequency of link updates due to arbitrary node connections/disconnections poses challenges in network convergence
- **Scalability:** frequent unpredictable topological changes poses challenges for dynamic routing and network configurations, identification of emergent behaviors can be difficult to determine
- **Security/Privacy:** dynamic nature of nodes makes physical security difficult, wireless medium operates in the open and can potentially be accessible by malicious actors i.e., MITM

- **Interoperability:** integration of heterogeneous devices is challenging due to issues such as software compatibility, use of proprietary/standard communications protocols, service support, etc.

2.2.1 USMC MANET Employment

The demand for commanders in the field to gain access to real-time battlefield information as well as their desire to extend situational awareness down to the squad-level has fueled the widespread distribution of SDRs to troops at the tactical edge under the DoD's Joint Tactical Radio System (JTRS) program [64]. As opposed to traditional hardware-defined radios, SDRs leverage software to perform digital signal processing of RF signals, enabling a single radio the ability to receive and transmit multiple types of radio protocols, otherwise known as waveforms. The Software Communications Architecture (SCA), developed under the JTRS program, serves as an international open standard framework for military SDRs, defining the foundation for which to instantiate and manage waveforms in a way that supports multi-national interoperability, data security, and software portability while minimizing costs [64].

With improvements in mobile ad-hoc technology, DoD wideband waveforms (such as ANW2, Soldier Radio Waveform (SRW), Wideband Networking Waveform (WNW), Tactical Scalable MANET X (TSM-X)) have started to replace legacy narrowband waveforms (i.e., amplitude modulation (AM)/frequency modulation (FM), very high frequency (VHF)/ultra high frequency (UHF) LOS, Single Channel Ground and Airborne Radio System (SINCGARS)) due to their ability to effectively and efficiently transmit voice, video, and data on-the-move across a multitude of operational environments. These waveforms add redundancy while limiting single points of failure by taking advantage of the self-healing nature of MANETs [65]. Likewise, the flexibility to move about and communicate from a vehicle has allowed field commanders the ability to lead their troops from the front, thereby reducing the decision cycle through real-time information access [65].

Developed by Harris Corporation, ANW2 is a proprietary waveform being used for secure MANETs across the Marine Corps. The workhorses of ANW2 networking are the Harris Falcon III AN/PRC-117G manpack radio and its handheld version, the Harris Falcon III AN/PRC-152A. Typically employed at the platoon-level and below, ANW2 was designed

(a) AN/PRC-117G Manpack Radio. Source: [66]. (b) AN/PRC-152A Handheld Radio. Source: [67].

Figure 2.19. USMC ANW2-Capable Radios

for supporting scalable, high-bandwidth data communications in both stub and/or transit network configurations [64], [68]. Once the mission plan (Communications Programming Application (CPA)) is configured, radios within the ANW2 network will automatically route between each other [69]. The formation and self-synchronization (no GPS required) of a ANW2 subnet can be completed in less than 30 seconds with subnet healing and joins being completed within five seconds [64]. Depending on the transmission bandwidth and operating environment, data rates of 50 kbps up to 5 Mbps can be expected, see Table 2.3. Backhaul or network integration is achieved through a radio configured with a RedLAN connection which allows routing outside of the ANW2 network [69]. Figure 2.20 shows a typical ANW2 topology with a RedLAN connection.

The limitations of the AN/PRC-117G's transmission range limits its practical use beyond the platoon-level [68]. Operators must also use a cable to tether the radio to an auxiliary device (i.e., laptop) in order to provide data capabilities, proving cumbersome and a challenge for personnel mobility [70]. The AN/PRC-117G is also relatively heavy (12 lbs), "bulky (3.7 H x 7.4 W x 8.8 D inches), and expensive (over $30K/radio)" [70]. Another performance setback for ANW2 stems from its use of Time Division Multiple Access (TDMA) which in effect reduces the available bandwidth across all the radios in the MANET due to sharing of the same frequency band [70]. The handheld AN/PRC-152A,

while almost 10 lbs lighter (2.7 lbs) and smaller (10.25 H x 3.0 W x 2.5 D inches), suffers from a lower transmission range (5W transmitter) while still being relatively expensive ($13K/radio) [70]. Overall, the cost of equipping the force has limited the distribution and availability of these radios across the Fleet Marine Force [70].

Table 2.3. ANW2 Data Rates

Bandwidth	Data Rates
1.2 MHz	50 kbps to 2.8 Mbps
5 MHz	200 kbps to 5 Mbps

Source: [71].

The current DoD approach to MANET authentication is largely based on symmetric (NSA Type 1) encryption. Type 1 encryption refers to NSA-developed or approved algorithms (i.e., AES-256, FIREFLY, HAVEQUICK) which protect classified information (up to Top Secret) on products used by the U.S. Government, their contractors, federally sponsored non-U.S. Government activities, and North Atlantic Treaty Organization (NATO) allies [72]. In this scheme, authentication between the sender and receiver is achieved through the use of a pre-shared key which is used to both encrypt and decrypt the plain text and respective cipher text. Using an external fill device (which receives the keys from the KDC), the keys are transferred via DS-101 (current NSA key transfer protocol) to the supporting SDR [72]. While symmetric cryptography is computationally fast and efficient for data encryption, key distribution and management still remains a fundamental challenge [16]. To protect these keys from compromise, cryptographic key rollovers (key changes) are frequent and manually performed (via a key fill device)– a redundant, time-consuming task. As such, symmetric encryption is not very scalable for larger tactical networks.

2.3 Software-defined Networks

With the intent of widespread access and global information sharing, traditional IP networking infrastructure was designed to be decentralized. Despite the widespread adoption of traditional networking, the explosive growth of the Internet and IoT devices have spawned massive network infrastructures that have become very complex and difficult to

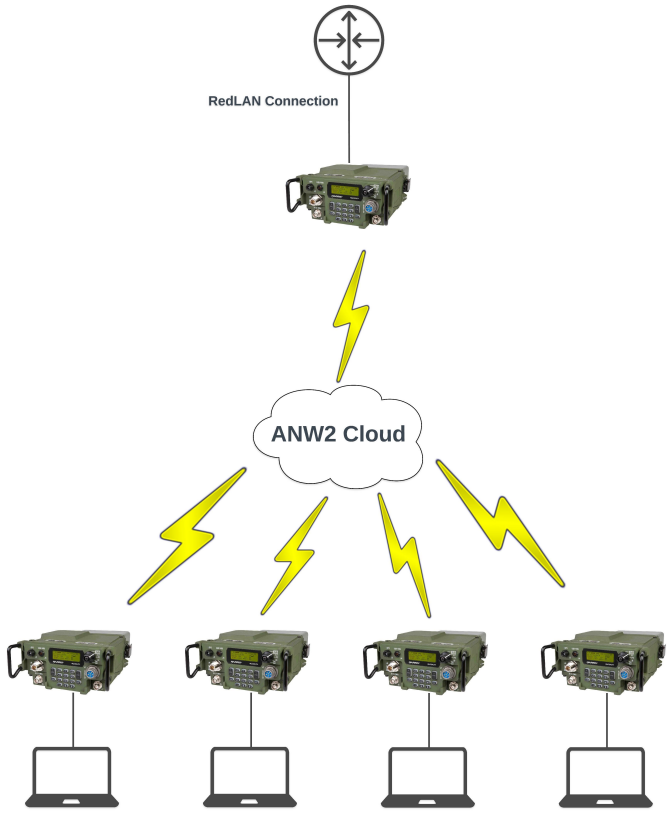


Figure 2.20. ANW2 Network with RedLAN Connection. Adapted from [69].

manage [73]. In order for administrators to implement network policies, individual network devices, network applications, and middleboxes (i.e., firewalls, intrusion detection systems, load balancers, deep packet inspection tools) have to be manually configured using low-level vendor-specific programming languages [73]. Adding to this complexity, traditional networks inherently lack the capability to dynamically reconfigure and respond to network faults and load changes which consequently requires more manual administration [73].

To further exacerbate the situation, traditional networks are vertically integrated meaning that the control plane, which makes protocol-based decisions on how to manage network traffic, and the data plane, which sends the traffic according to the control plane's decision, are attached within the same networking devices [73]. The end result is that the static infrastructure of traditional networking makes future innovation difficult as it lacks both the resiliency and ability to adapt to ever-changing network conditions [73]. Fortunately, the

innovations of the information age have created a digitally connected society where tasks which were once defined by hardware have now been replaced by software. Leading this paradigm shift in the networking domain is an emerging approach to network infrastructure virtualization known as SDN [73]. In accordance with [73], SDN is defined by the following:

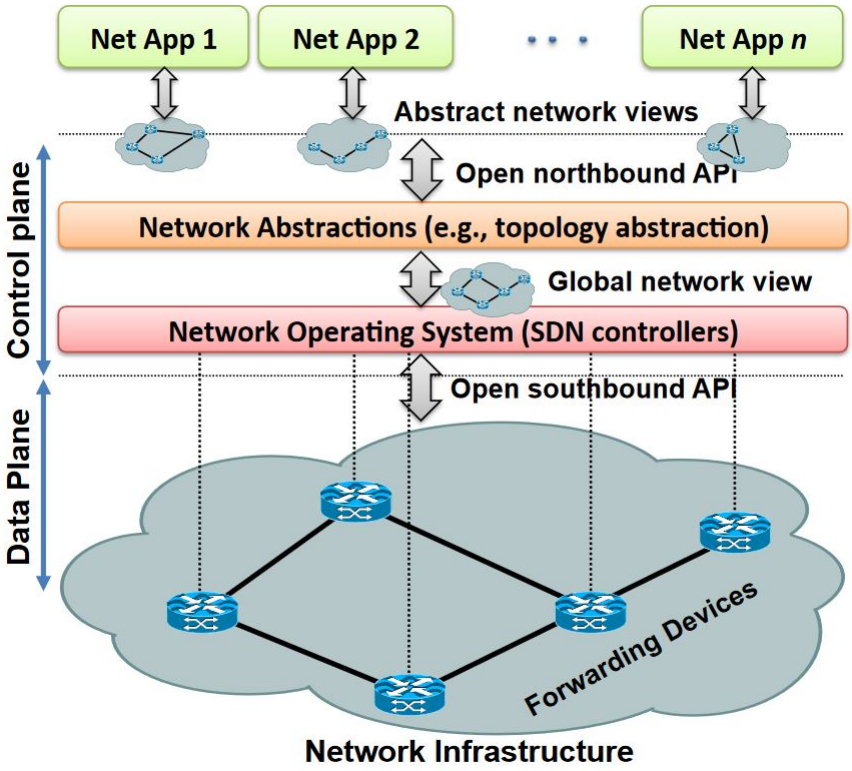
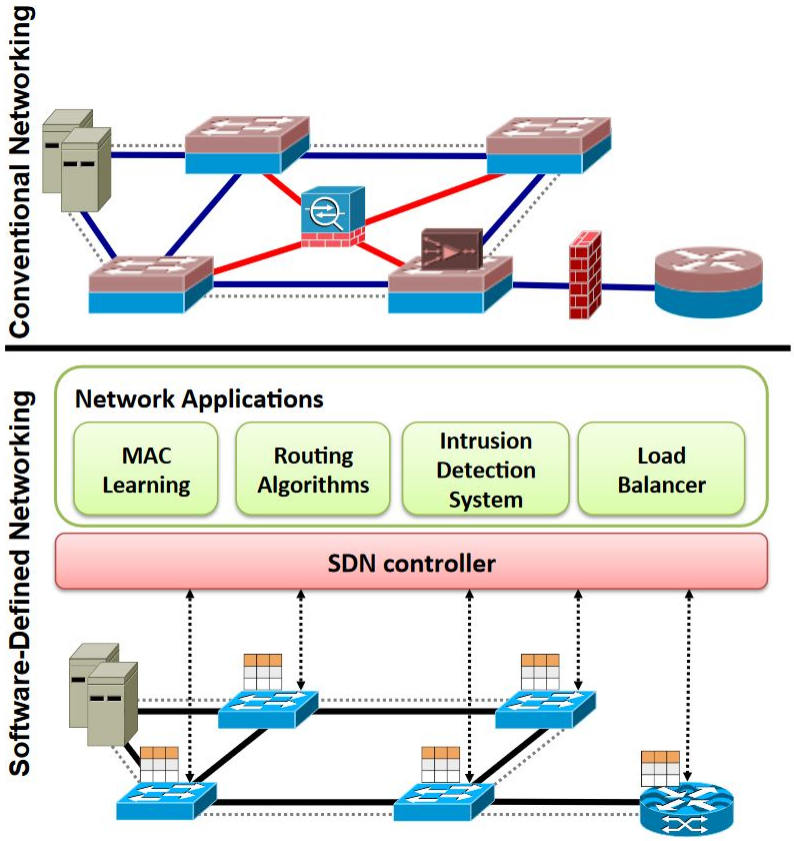


Figure 2.21. SDN Architecture. Source: [73].

1. **Separated control and data plane:** SDN disassociates the vertical integration of traditional networking by separating the control plane from the data plane [73]. In other words, network devices no longer possess control functionalities and only serve as packet forwarding elements [73].
2. **Flow-based forwarding:** Instead of traditional destination-based forwarding, forwarding is based on a sequence of packets (flows) which are stored in a flow table and act as a matching criterion with instructions for action [73]. Flow programming allows for flexibility through the aggregated abstraction of the data plane components (i.e., switches, routers, firewalls) [73].



SDN allows middlebox services to be abstracted away as SDN controller applications, thereby simplifying network administration and management [73].

Figure 2.22. Traditional Network versus SDN. Source: [73].

3. **Logically centralized control logic:** The control plane's logic is executed via a logical SDN controller, also called the network operating system (NOS) [73]. SDN controllers serve as the strategic brains of the network by managing the flows going to the switches and routers based on the abstracted network view [73].
4. **Programmable Network:** Software (network applications) running on the SDN controller allows network programmability to permeate across the data plane giving SDNs an unprecedented advantage over traditional networking infrastructure [73].

The separation of the data and control plane in SDNs allows three primary advantages over traditional networks: centralized management, scalability, and improved security [74].

1. **Centralized Management:** The SDN controller provides network administrators

an abstracted view of the entire network while also giving them the capability to manage the entire network from one location (locally or remotely); thus, improving automation and eliminating individual device management.

2. **Scalability:** Centralization administration allows for network resources and virtualized services to be provisioned on-demand, circumventing manual installations and configurations.
3. **Improved Security:** The entire network security policy can be centrally enforced to ensure compliance (i.e., software patching) and to dynamically respond to external threats (i.e., network segmentation). Network Function Virtualization (NFV) reduces vendor lock-in.

2.3.1 Software-defined Mobile Ad-hoc Networks

While still at its infancy stage, SD-MANET is an evolving, new field of research that integrates the principles of SDN to establish and manage a wireless multi-hop network of peer-to-peer nodes in an infrastructureless environment [75]. This networking paradigm creates unique challenges, especially as it pertains to the tactical environment:

- The employment of the SDN over a MANET requires a TCP between the SDN controller and each forwarding node for control messages (i.e., OpenFlow) [75]. However, the connection between MANET nodes may be unreliable due to the dynamic nature of MANETs [75].
- SDN control message sizes may be too large for MANETs in a tactical environment which typically exhibit low bandwidth and intermittent links [75].
- Recently proposed SD-MANET architectures, such as vehicular ad-hoc networks (VANETs), assume a single-hop link from the forward node to the SDN controller over a separate channel (i.e., cellular) [75]. However, in tactical MANETs, the SDN controller should be able to communicate with all nodes in the network using multi-hop.
- Similar research also assumes a base station to host the SDN controller and a location service (i.e., GPS) for tracking forward nodes is available [75]. For tactical situations, the SDN controller would most likely be hosted on mobile nodes (i.e., mobile operations center, vehicle-mounted node, foot-mounted node) in the network since the environment is expected to be infrastructureless. Additionally, location services

should not be expected at each node.

2.4 Related Work

While little industry support exists in the area of SD-MANETs, the field has started to gain traction among researchers seeking to combine the benefits that each paradigm offers. Lacking a dedicated mechanism for global network and resource management, SDNs offers a breakthrough concept by enabling programmability on an otherwise traditionally static network infrastructure. However, the centralization of the SDN controller introduces challenges in link reliability and network performance when managing highly dynamic and mobile networks. Little work has been conducted regarding the application of SDNs in the wireless domain, as the majority of research is concentrated on applications in wired infrastructures such as Internet service provider (ISP) networks and data centers [76].

Nobre et al. was the first research to evaluate the application of a SDN architecture in battlefield networking (BN) [77]. The study defined BN as the concept of independent, dispersed tactical networks integrated as part of a larger network, typically through satellites. Although the work did not explicitly focus on MANETs, its vision of applying SDN technology to improve battlefield communications and security serves as the catalyst for this thesis.

Poularakis et al. proposed architecture designs for tactical SD-MANETs [76]. Specifically, the authors focused on where and how to organize the SDN controllers in the network (globally, locally, or on mobile nodes) while addressing the advantages and disadvantages of each location.

In [78], Poularakis et al. argued that the centralized nature of SDN created problems with SD-MANET robustness. The study introduced methods of hybrid control logic to add redundancy and flexibility by pushing control logic (data forwarding decisions) to the mobile nodes: migrating to a distributed routing protocol between nodes during poor network conditions, allowing nodes in a cluster to determine routes independently, and storing forwarding rules on the mobile nodes.

Zouridaki et al. explored a distributed CA-based PKI scheme employing threshold-based cryptography for MANETs based on ECC in order to provide authenticated and

encrypted information exchange [79]. The research used a clustering of nodes to reduce overhead and enable scalability while ECC was adopted due to its computational performance (reduced key size). The study's proposed PKI architecture was shown to be compatible with current smartcard technology. While SDN was not employed in this research, the work highlights areas that may be improved with the hybridization of SDN.

To the extent of the author's knowledge, this thesis is the first work to explore a CA-based PKI scheme for tactical SD-MANETs.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
Design

This increasingly complex security environment is defined by rapid technological change, challenges from adversaries in every operating domain, and the impact on current readiness from the longest continuous stretch of armed conflict in our Nation's history. In this environment, there can be no complacency—we must make difficult choices and prioritize what is most important to field a lethal, resilient, and rapidly adapting Joint Force. America's military has no preordained right to victory on the battlefield.

—Jim Mattis, 26th U.S. Secretary of Defense [2]

The objective of this chapter is to develop an architectural design for ORION, an SD-MANET authentication scheme that employs SDN controllers to provide on-demand, heterogeneous device-user authentication (in addition to its traditional flow routing responsibilities). An implementation for ORION is developed in order to evaluate its strengths, weaknesses, and viability as a future USMC network mobility model. The technical requirements of the Purebred infrastructure are examined in order to develop a framework for ORION that will support the intent of the NDS, CPG, and the future of authentication in tactical edge networks.

3.1 Technical Requirements

ORION leverages the current Purebred infrastructure to support credentialing when back-end connections are available in order to take advantage of existing system solutions and functions that would otherwise not be available in the absence of a back-end. See Figure 3.2 for an overview of ORION's conceptual architecture design. This section focuses on two major technical requirements of ORION:

1. In the event that a mobile device is lost, destroyed, or compromised, ORION shall have the capability to facilitate timely revocation of device derived credentials by a

Trusted Agent (TA). Compromise or expiration of CAC credentials shall also result in the revocation of credentials pertaining to ORION.

2. ORION shall be capable of enrolling and credentialing new mobile devices in the absence of a back-end connection. An expedited method shall be offered that emphasizes automation of this process to the maximum extent possible while adhering to NIST guidelines for Authenticator Assurance Level (AAL) 3.

Supplementary topics that will be addressed include decryption key escrow and system interoperability.

3.2 Consideration of Purebred Features

The following is an outline of fundamental Purebred solutions that will be taken into consideration when designing ORION in order to determine how certain features of Purebred can be built upon and adapted for use in a mobile ad-hoc tactical environment [57] [80]:

1. Purebred integrates into the DoD PKI enclave;
2. Purebred's source code is available for government review;
3. Purebred supports both centralized and distributed key generation;
4. Purebred supports decryption key recovery;
5. Purebred authenticates and authorizes all parties involved in the provisioning process (i.e., device, people, services);
6. Purebred uses NIST-approved cryptographic algorithms and key sizes;
7. Purebred supports NIAP-validated or in-evaluation devices;
8. During registration, users must demonstrate possession and control of CAC per NIST SP800-157;
9. Purebred facilitates automated revocation of derived (software) credentials when associated CAC is revoked.
10. Purebred avoids passing private keys, passwords, challenge values, etc. through MDM in plaintext form;
11. Purebred enrollment includes a live agent in the process for human verification;
12. Purebred supports remote registration, reducing labor and allowing users to register without having to visit a provisioning facility;
13. Purebred supports modern certificate enrollment protocols (i.e., Enrollment over

Secure Transport (EST));

14. Purebred performs certificate validation in accordance with RFC 5280, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile";
15. Purebred supports Personal Identity Verification (PIV) Authentication, Digital Signature, and E-mail Encryption credentials along with a Device credential for each authorized device;
16. Derived PIV Authentication and Digital Signature certificates feature the same expiration (notAfter) value as the CAC credentials they were derived from.

3.3 Assumptions

The following assumptions are made in order to set conditions for the ORION design structure:

- Internet connection may be unavailable in tactical environment– limited/no connection to back-end services;
- ORION is platform agnostic– cross platform differences (i.e., iOS, Android) will not be evaluated;
- ORION Registration app has been pre-installed on all smartphones prior to deployment;
- MDM configurations have been installed prior to deployment;
- SDN is currently deployed to support routing in the SPAN;
- SDN controller possesses the means to wirelessly communicate with devices within its own network (i.e., Wi-Fi Direct, Bluetooth, Apple's Multipeer Connectivity framework);
- Ad-hoc links possess sufficient bandwidth to support ORION;
- ORION CA key pair generation will utilize FIPS approved cryptographic modules and methods;
- Backup copies of the ORION CA private key will be created and stored using FIPS approved methods;
- Use of biometric sensors and processing conforms with NIST standards for AAL 3;
- All key pairs use a RSA 2048 bit modulus.

3.4 Hierarchical Chain of Trust

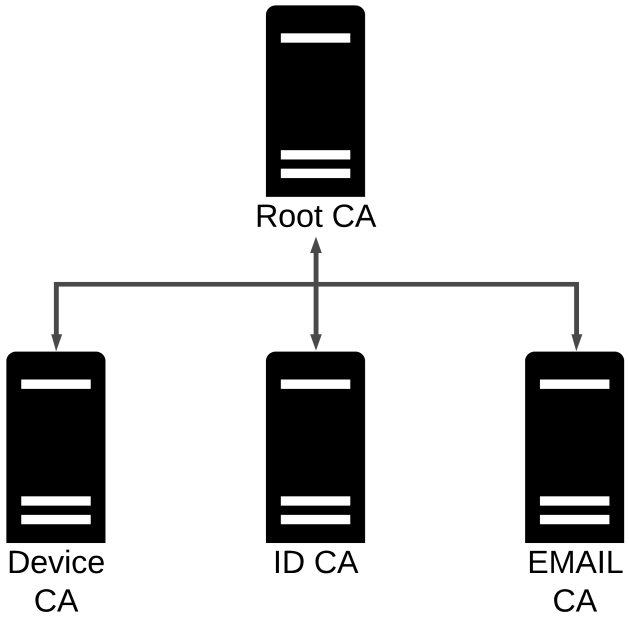


Figure 3.1. ORION CA Hierarchy

The DoD Root CA, maintained by the NSA, is the ultimate authentication authority for all public key certificates created within the DoD PKI hierarchy. With regards to the notion of trust, the DoD Root CA serves as the common trust anchor for all intermediate CAs subordinate to the Root CA. The DoD Root CA generates its own key pairs and signs its own public certificate as well as the certificates of its subordinate CAs. The subordinate CAs, in turn, use the DoD Root CA's signature on their certificate to prove to their subordinates that they are trusted authorities within the PKI hierarchy. The ORION CA will ideally serve as an subordinate CA underneath a DISA-managed intermediate CA to provide a security partition and trust barrier for the ORION domain. The ORION CA will manage all aspects of subscriber public key certificates within its hierarchy.

3.5 Trusted Agents

ORION TAs, who serve as functional equivalents to Purebred Agents, are individuals with role-based authorization to execute Subscriber and device identity validation during

the registration process on behalf of the ORION CA. TAs ensure that there is a human audit in the registration process to ensure visual vetting of the individual and/or device. That is, a TA assists in authenticating the Subscriber to the CA. TAs do not have privileged permissions to perform any functions on behalf of the CA or access to the CA. TAs roles are identified on their public key certificates by a Subject field Common Name (CN) of "TA" followed by their EDIPI (i.e., CN = TA.1234567890).

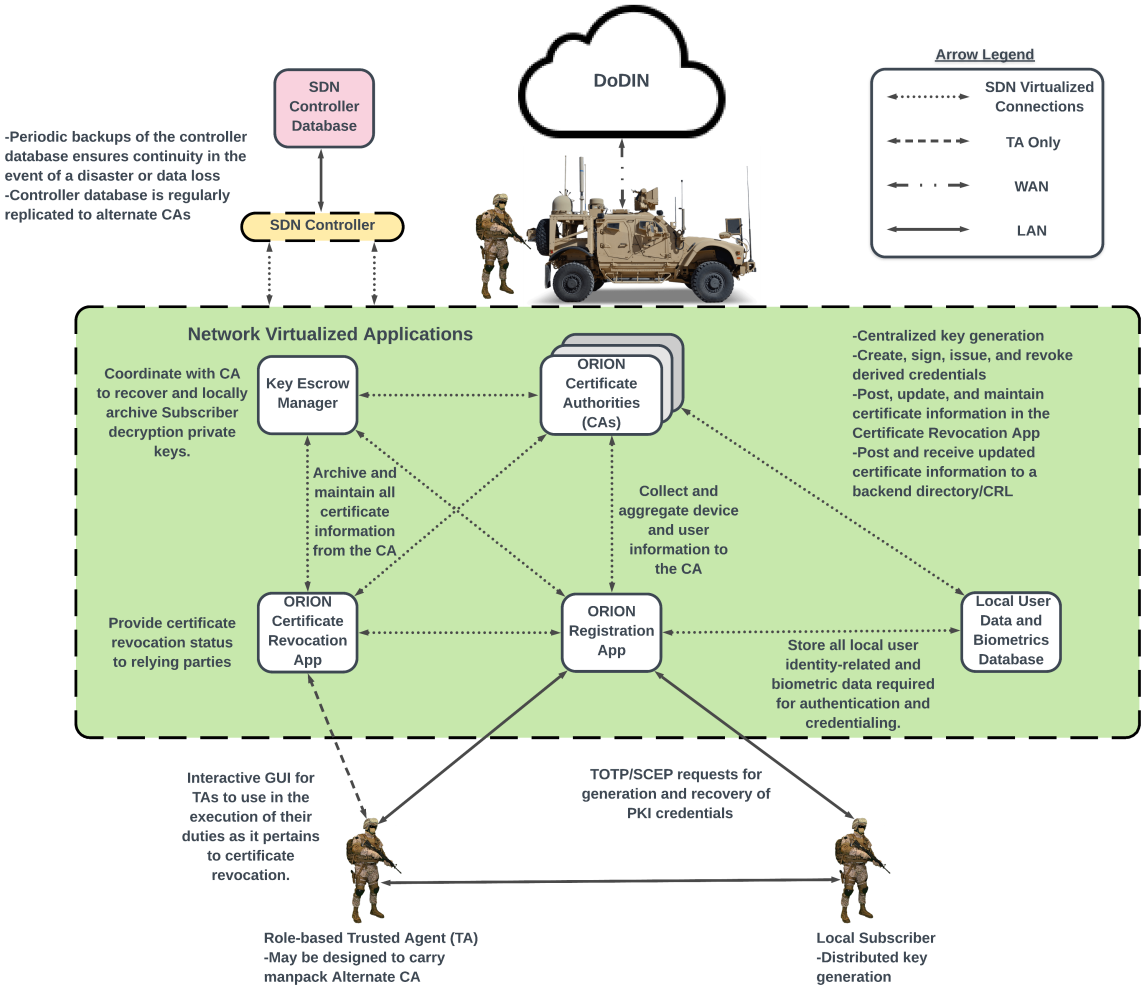


Figure 3.2. ORION Conceptual Design

3.6 Subscribers

ORION Subscribers are end users identified by their associated public key certificate Subject field and must demonstrate proof of possession of the respective certificate's corresponding private key. Subscribers must possess a current and valid U.S. Government issued CAC in order to be actively enrolled in the system.

3.7 System Architecture

ORION's architecture is defined by the OpenFlow open-source standard and consists of a software-defined controller responsible for provisioning and managing virtual software applications required for proper functioning of the system. ORION applications and responsibilities include the following:

1. ORION CAs
    - Create, sign, issue, and revoke public key certificates for respective class type (i.e., Device, Identity CA, E-mail CA);
    - Post and receive updated certificate information to a back-end directory/CRL;
    - Post, update, and maintain certificate information in the Certificate Revocation App;
    - If CAC credentials are expired/revoked, perform automated revocation of ORION credentials pending connection to back-end database (i.e., DMDC)
2. ORION Registration App
    - Collect and aggregate device and user information to the CA;
    - Facilitate time-based one-time password (TOTP) requests for generation and recovery of PKI credentials;
    - Demonstrate possession of valid CAC.
3. ORION Certificate Revocation App
    - Archive and maintain all certificate information posted from the CA;
    - Provide certificate revocation status to relying parties;
    - Provide an interactive graphical user interface (GUI) for TAs to use in the execution of their duties as it pertains to certificate revocation.
4. User/Biometric Database
    - Store all local user identity-related and biometric data required for authentication/credentialing.

5. Key Escrow Manager
    - Coordinate with CA to recover and locally archive Subscriber decryption private keys.
6. SDN Controller Database
    - Store periodic backups of the controller configuration to ensure continuity in the event of a disaster or data loss.

3.8 Credential Registration

Per [81], the CAC is the primary DoD PKI credential for logical authentication to unclassified networks, systems, servers, and applications as it meets the criteria for AAL 3 in accordance with NIST SP800-63-3, "Digital Identity Guidelines". While the CAC is optimized for use in traditional wired environments (i.e., desktops, laptops), its inadequacies in the wireless mobile environment have led to a revisal of FIPS 201-1, "Personal Identity Verification (PIV) of Federal Employees and Contractors" [44]. To facilitate authentication of mobile devices as well as cost-efficiency, the updated FIPS 201-2 seeks to leverage the existing DoD PKI infrastructure and the trusted identity of a current, valid CAC holder to provision an additional credential called a derived credential [44]. When compared to regular employment of a smart card reader and CAC for secure authentication, derived credentials are a more pragmatic solution for "CAC-enabling" mobile devices. NIST SP800-157, "Guidelines for Derived Personal Identity Verification (PIV) Credentials", further states that in order to provision derived credentials, the user must first prove possession of a valid CAC [44]. As such, the Purebred registration process requires a computer with a smart card reader in order to ensure that the user requesting new credentials can demonstrate CAC possession before derived credentials are issued.

To align with existing policy, ORION leverages the same prerequisite of demonstrating CAC possession (or equivalent thereof to a max practical extent) during the registration process. Additionally, credentials derived for use in ORION will ideally be an entirely new set of credentials separate from Purebred– this is analogous to having separate credentials for garrison and tactical. For one, it would allow implementation of the principle of least privilege, ensuring user segmentation and security compartmentalization from the Purebred and garrison DoD PKI domain. Moreover, it would enable greater flexibility and control for use in the local tactical environment (i.e., certificate revocation).

3.9 Identity and Device Trust

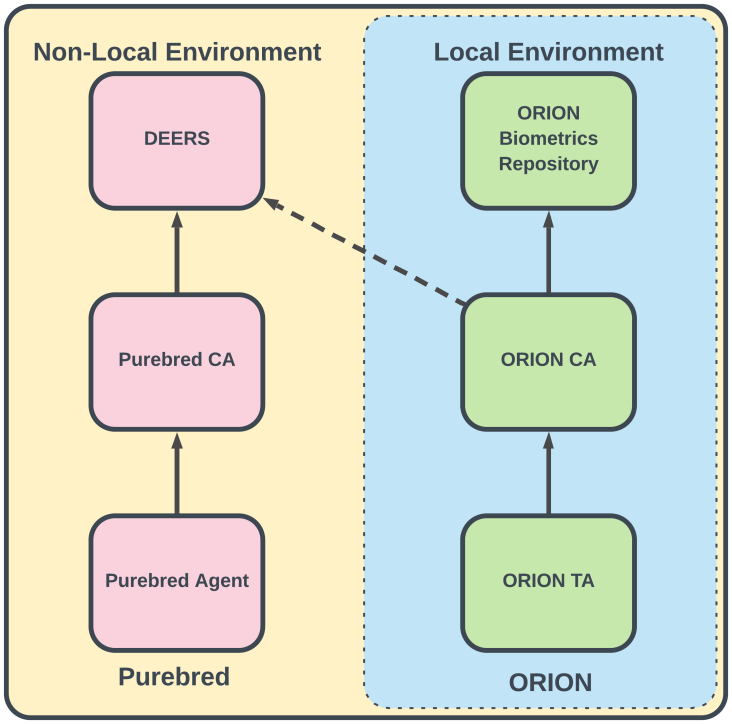


Figure 3.3. ORION Identity Verification Architecture

Utilizing a similar identity assurance model to Purebred, ORION implements identity attestation through DEERS when a stable back-end connection is available for establishing verification services. In the absence of a backhaul connection, ORION will default to its local biometric database which contains the fingerprint and facial data of CAC owners collected during the CAC registration process. These biometric modalities have been physically verified at a DEERS ID support office during capture to belong to the enrollee and thus, serve as an alternative approximation to a DEERS verification. Biometric authentication will also be utilized during log-in for participating devices. Additional rationale for biometric use is detailed in the following subsection.

3.10 Use of Biometrics

Biometric authentication proves "something that you are" and is measured by physical characteristics (i.e., fingerprint, iris, facial) and/or behavioral characteristics (i.e., how you type, how you walk) [82]. The use of biometric modalities in ORION will conform with NIST SP800-63B, "Digital Identity Guidelines: Authentication and Lifecycle Management", which states that biometric authentication may be implemented in AAL 3 so long as "the biometric sensor and subsequent processing methods meet the performance requirements" of the aforementioned publication. Biometric authentication has seen limited use in tactical application for reasons which include [82]:

- Biometric False Match Rates (FMRs) provided limited confidence in Subscriber authentication when used by itself;
- Biometrics is vulnerable to spoofing attacks;
- Biometric comparison uses probabilistic means vice deterministic techniques;
- Proliferation of biometric technology is limited and the technology itself is not fully mature;
- Biometric characteristics are considered open secrets (e.g., facial images can easily be obtained online or by camera, fingerprints can be lifted and copied, iris patterns can be captured using high resolution images).

Despite these limitations, when combined with other authentication methods, biometric technology is considered secure [82]. Since most modern smartphones are factory equipped with biometric sensors and subsequent processing software, costs associated with equipping/upgrading devices are eliminated. Additionally, biometrics provides a potential defense measure against device post-compromise in current tactical radio networks. For instance, if a programmed radio is lost during a convoy, any individual, including the adversary, who acquires the radio can listen in on internal communications because there is no authentication mechanism to prevent unauthorized use the device. The compromised communication window can potentially last hours or more if the losing unit fails to quickly recognize the loss and execute a timely cryptographic key rollover. The consequences may be severe, posing significant risk to life and jeopardizing mission success (i.e., positional compromise, operational compromise). Device biometric authentication for log-in provides a solution to closing this vulnerability gap.

In an effort to leverage and provide stronger confidence in biometric authentication, the following authentication requirements are proposed and adapted from [82]:

- Biometrics shall only be used in combination with a MFA solution that incorporates a physical authenticator in order to certify that the user proves "something you have" (i.e., associated smartphone during log-in, CAC possession during enrollment);
- The biometric system shall allow no more than 3 consecutive failed authentication attempts. Once that limit is reached, an exponentially increasing delay starting at 30 second will be imposed following each successive failed attempt;
- Alternative methods of authentication may also be offered as a substitute to the current biometric method upon reaching the failure limit (i.e., a different modality method, PIN, password);
- Biometric comparison for device log-in shall be performed locally on the user's device;
- Biometric comparison for the registration process shall be performed remotely on a central biometrics verifier database;
- Biometrics shall be used during all stages of the enrollment process to prevent repudiation of the enroller and to ensure that the entire credentialing process is being conducted by the same individual.

3.11 Dynamic Passwords

ORION CAs emulate Purebred's use of time-limited and context-limited OTPs values during the registration process to provide proof of trustworthy credential usage with mobile devices that it communicates with [8]. Per [82], AAL 3 requires safeguard against replay attacks. Thus, the use of TOTPs is considered replay resistant as the values are only valid within a limited, specified window. As stated in [8],

> The one-time password entered into the Purebred Registration app serves to signal the user's authorization of a request while authenticating selected attributes contained in the request.

Just as in Purebred, TOTPs are also used in ORION to authorize issuance of device and all derived certificates. A KDF based on NIST SP800-108 is used to generate the symmetric key

for the TOTP algorithm [80]. As added security feature, ORION requires that Subscribers authenticate using biometrics before a TOTP value can be generated. See Figure 3.4 for an overview of the CA's TOTP generation and verification process.

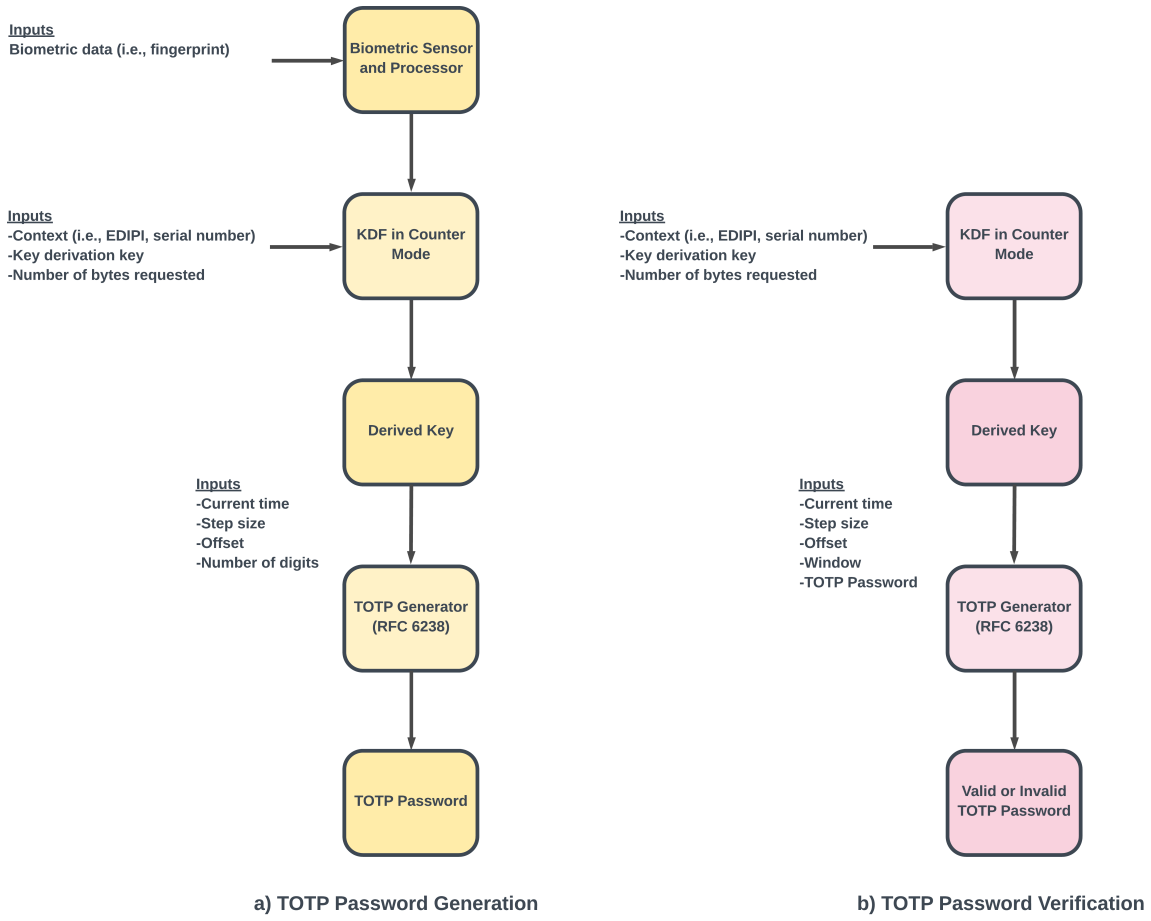


Figure 3.4. Multi-factor TOTP Generation and Verification. Source: [80].

3.12 OpenFlow Security

The OpenFlow protocol serves as the main southbound protocol for ORION's control plane allowing switch flow tables to be configured properly in order to manage the forwarding of packets. To maximize bandwidth and facilitate interoperability, securing the southbound communications channel between the SDN controller and its switch(es)

with TLS is an optional feature. However, it is highly recommended that TLS be enabled for southbound communications between the control and the forwarding plane. This is especially important as unsecure communications between the controller and switch is vulnerable to eavesdropping and MITM attacks [83]. Ensuring that the controller and its switch(es) are mutually authenticated via TLS provides a mechanism for encrypting control traffic [83]. As a result, this security measure also protects ORION against MITM attacks or any malicious impersonation of its switch(es) or controller [83].

3.13 Full and Expedited Registration

ORION provides the option for two modes of device registration and credentialing which may be situationally dictated based upon the needs of the using entity:

1. Full registration: requires TA witnessing, CAC demonstration/verification and/or biometric authentication, and supplemental authentication using TOTPs;
2. Expedited registration: eliminates manual operations; registration is entirely automated through biometrics. TA oversight is removed along with CAC demonstration/verification and TOTP authentication.

The expedited mode is less secure, but may be necessary in urgent situations. It is recommended that the use of expedited registration provides more restrictive services when compared to a full registration (i.e., limited access in terms of credential validity period or limited functionality in terms of post-registration services provided). Nevertheless, the use of expedited mode still conforms to NIST guidelines for AAL 3 as authentication is accomplished using a multi-factor cryptographic device (i.e. biometric-activated smart device with secure processor) [82]. The following excerpt was taken from NIST SP800-63B:

> ...a biometric is recognized as a factor, but not recognized as an authenticator by itself. Therefore, when conducting authentication with a biometric, it is unnecessary to use two authenticators because the associated device serves as "something you have," while the biometric serves as "something you are. [82]

3.13.1 Full Device Enrollment Process

The full device enrollment process has been adapted from [84] and depicts the steps necessary for the Subscriber to provision a device credential which is required for obtaining new derived credentials. The full device enrollment process consists of a pre-enrollment phase followed by a final enrollment phase with TA vetting and CAC or biometric verification. See Figure 3.5 for the full device enrollment flow diagram.

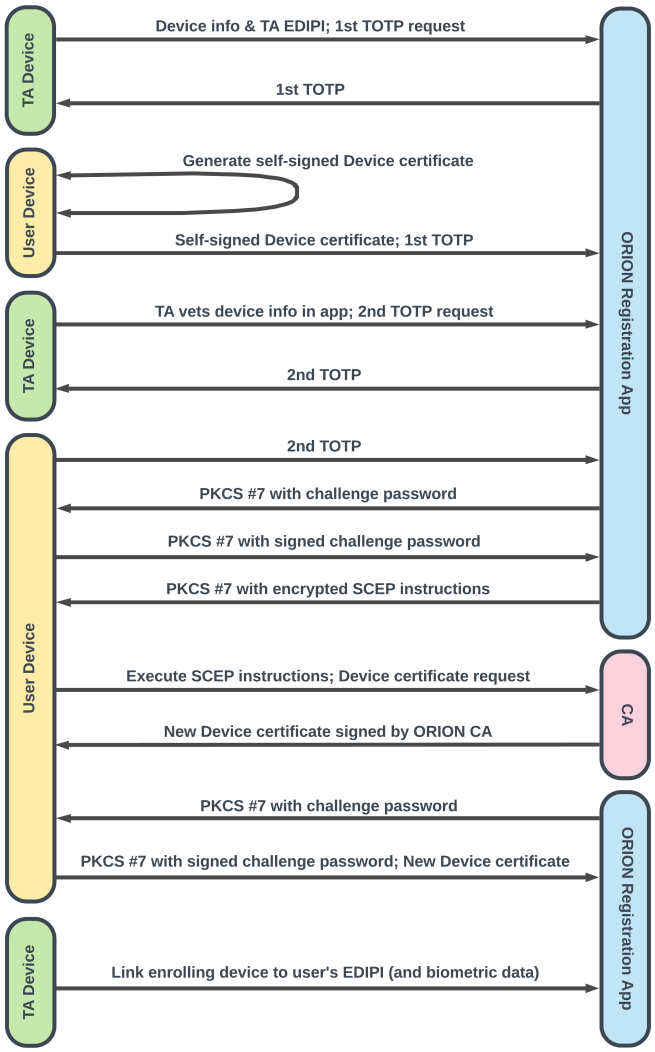


Figure 3.5. Full ORION Device Enrollment. Adapted from [8].

1. TA Device The TA establishes a mutually authenticated TLS with the ORION Regis-

tration App using the TA's own device and the TA's token.

2. 

The TA obtains a TOTP from the app by entering the enrolling device's serial number and their own EDIPI for future authorization association. The TOTP is signed using the app's private key.
3. 

The Subscriber, on their enrolling device, accesses the ORION Registration App. The enrolling device generates a self-signed certificate. The device's self-signed public key certificate, certificate hash, and other device identifiers are posted to the app along with the TOTP value generated by the TA. The Subscriber's device is now connected to the app via Server Side TLS.
4. 

The app verifies that the TOTP is valid and if so, the device is added to the list of eligible devices for enrollment in the app.
5. 

The TA visually verifies that the device information presented in the app matches the information on the enrolling device.
6. 

Once the TA has completed the verification, the TA prompts the app for a second TOTP.
7. 

The TOTP is entered on the enrolling device to enable device pre-enrollment.
8. 

The app sends a challenge password to the enrolling device using the pre-existing Server Side TLS session.
9. 

The enrolling device responds to the challenge by signing it with its device private key.
10. 

The app verifies the signature with the enrolling device's corresponding public key and sends an encrypted SCEP instructions (containing the enrollment TOTP) using the enrolling device's self-signed public key certificate.
11. 

The enrolling device generates a new key pair and uses the information in the SCEP to craft a Certificate Signing Request (CSR) for the CA in order to obtain a formal device certificate.
12. 

The app sends another challenge password which the enrolling device now signs with the new private key.
13. 

The enrolling device sends the signed challenge and CA-issued device public key certificate to the app. The app verifies the validity of the device's public key certificate.
14. 

To associate the enrolled device with the Subscriber, the TA verifies the

Subscriber's identity by confirming their CAC. Within the app, the TA assigns the device to the Subscriber by linking the device with the Subscriber's EDIPI. The TA may also associate the device to a biometric modality belonging to the Subscriber as an addition to the EDIPI.

15. User Device At this point, the Subscriber is now ready to obtain ORION credentials and recover decryption keys.

3.13.2 Expedited Device Enrollment Process

The expedited device enrollment process, as shown in Figure 3.6, removes the TA vetting process and the use of TOTP authentication. Device credentialing is accomplished through two biometric authentication steps– the first step initiates the registration process and the second step confirms device enrollment. The procedure required to provision an expedited device credential is as follows:

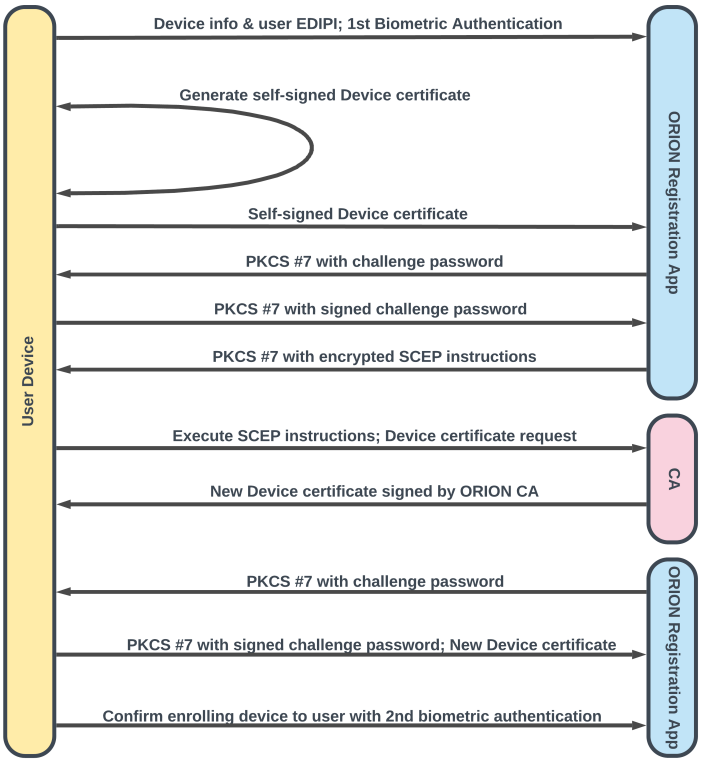


Figure 3.6. Expedited ORION Device Enrollment

1. User Device The Subscriber enters the enrolling device's serial number and their own EDIPI. The Subscriber sends the information to the Registration App and initiates device pre-enrollment via biometric authentication.
2. User Device The enrolling device generates a self-signed certificate. The device's self-signed public key certificate, certificate hash, and other device identifiers are posted to the app. The Subscriber's device is now connected to the app via Server Side TLS.
3. Registration App The app sends a challenge password to the enrolling device using the pre-existing Server Side TLS session.
4. User Device The enrolling device responds to the challenge by signing it with its device private key.
5. Registration App The app verifies the signature with the enrolling device's corresponding public key and sends an encrypted SCEP instructions using the enrolling device's self-signed public key certificate.
6. User Device The enrolling device generates a new key pair and uses the information in the SCEP to craft a CSR for the CA in order to obtain a formal device certificate.
7. Registration App The app sends another challenge password which the enrolling device now signs with the new private key.
8. User Device The enrolling device sends the signed challenge and CA-issued device public key certificate to the app. The app verifies the validity of the device's public key certificate.
9. User Device To complete the device enrollment process, the Subscriber associates their device to the newly CA-generated device credential when prompted via biometric authentication.
10. User Device At this point, the Subscriber is now ready to obtain ORION credentials and recover decryption keys.

3.13.3 Full Credentialing Process

The full Subscriber credentialing process has been adapted from [84] and depicts the steps necessary to provision new derived identity, signature, and encryption credentials. The Subscriber credentialing process is contingent on trust in the enrolling device. Therefore, the Subscriber must first successfully enroll the device (receive a CA-issued device public key certificate) before the Subscriber credentialing process can be initiated. See Figure 3.7 for the full credential enrollment flow diagram [8].

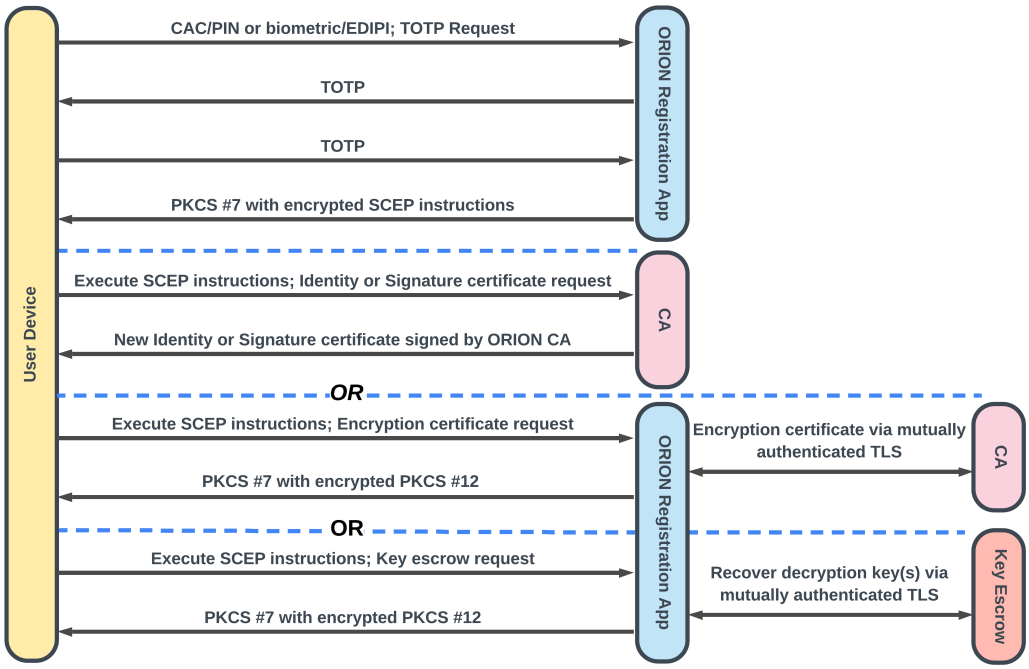


Figure 3.7. Full ORION Subscriber Credential Enrollment. Adapted from [8].

1. User Device The Subscriber accesses the ORION Registration App and presents their CAC/PIN or biometric/EDIPI in order to obtain a TOTP.
2. Registration App The app verifies that the Subscriber's EDIPI in the CAC certificate matches the EDIPI linked to the user's device or that the biometric data matches the EDIPI assigned to the device.
3. User Device The Subscriber submits the TOTP and the app generates an encrypted SCEP instruction (using the Subscriber's device public key certificate) that is then sent back to the Subscriber.
4. User Device The enrolling device decrypts the SCEP payload containing a TOTP to generate an identity or signature CSR to the CA through an established Server Side TLS. Key pairs for identity and signature credentials are generated locally on the enrolling device.
5. E-mail encryption certificate: After a successful TOTP exchange, the key pairs will be generated by the Registration App who will negotiate on behalf of the Subscriber to obtain an encryption certificate. The resulting decryption private key and encryption

certificate will be sent in an encrypted PKCS #12.

6. Key escrow: The escrow of decryption key(s) will be executed by the Key Escrow Manager via the Registration App in the same manner.

3.13.4 Expedited Credentialing Process

The expedited credentialing process, as shown in Figure 3.8, removes the use of TOTP authentication. The credentialing process is authorized through the use of biometric authentication. The procedure required to provision expedited Subscriber credentials is as follows:

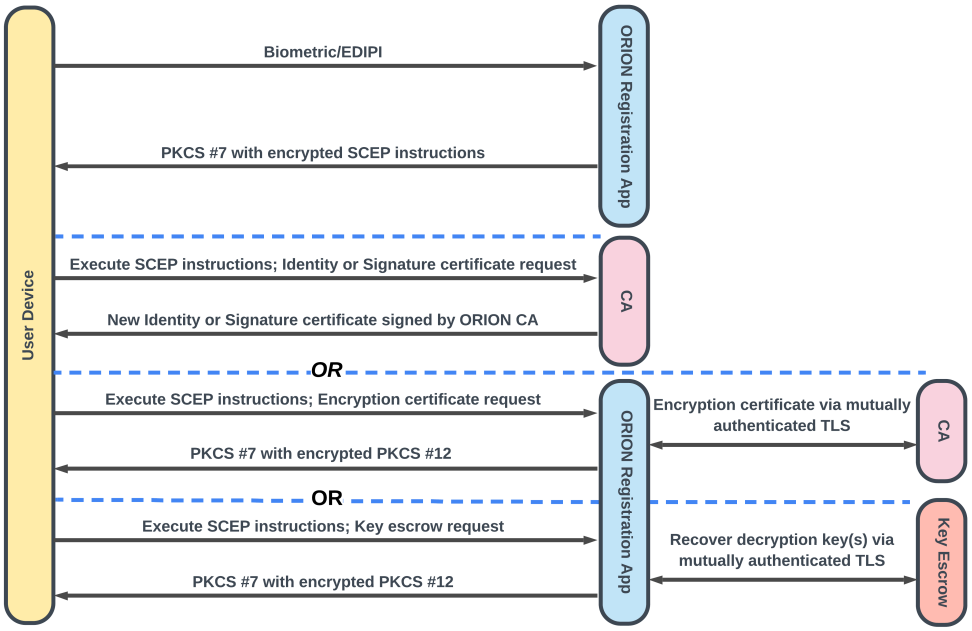


Figure 3.8. Expedited ORION Subscriber Credential Enrollment

1. User Device The Subscriber accesses the ORION Registration App, types in their EDIPI, and performs biometric authentication to initiate the selected credentialing process.
2. Registration App The app verifies that the Subscriber's EDIPI in the biometric database matches the EDIPI linked to the user's device.
3. Registration App The app generates an encrypted SCEP instruction (using the Subscriber's device public key certificate) that is then sent back to the Subscriber.

4. 

The enrolling device decrypts the SCEP payload containing a TOTP to generate an identity or signature CSR to the CA through an established Server Side TLS. Key pairs for identity and signature credentials are generated locally on the enrolling device.
5. E-mail encryption certificate: The key pairs will be generated by the Registration App who will negotiate on behalf of the Subscriber to obtain an encryption certificate. The resulting decryption private key and encryption certificate will be sent in an encrypted PKCS #12.
6. Key escrow: The escrow of decryption key(s) will be executed by the Key Escrow Manager via the Registration App in the same manner.

3.14 Credential Revocation

Certificate revocation duties, other than certificate expiration, are delegated from the CA to the TAs and executed via the Certificate Revocation App. Any subscribers, or authorized parties (as defined in a Certification Practice Statement (CPS)), may request revocation assistance from a TA.

3.14.1 ORION Revocation App

The design considerations for the ORION Revocation App takes into account three primary methods for managing revocation: CRLs, Online Certificate Status Protocol (OCSP), and OCSP Stapling.

Certificate Revocation Lists

A CRL is simply a list maintained by the CA, or an entity designated by the CA, which details all revoked certificates by serial number and the reason why the certificate was revoked [85]. CRLs are not efficient when it comes to scalability as the CRL will continue to grow over time along with its supporting infrastructure [86]. In addition, CRLs updates are only published periodically and thus, a dilemma exists where a client may allow an already revoked certificate [86]. On the other hand, CRLs are susceptible to availability issues. In the event that a client is unable to request the CRL, the client would not be able to make a decision regarding the validity of a certificate [86].

Online Certificate Status Protocol

OCSP is a more modern approach to CRLs as it circumvents the issue of outdated CRLs by enabling the client to on-demand request the status of a certificate's validity [86]. For this reason, the OCSP is more efficient when compared to CRLs as it does not require the client to download the entire CRL [86]. However, this makes OCSP a prime target for Distributed Denial of Service (DDoS) attacks as the service is completely reliant on its request-response protocol [86]. A privacy concern for OCSP is that its usage allows the CA to identify the time of request and the specific certificate that was queried [86]. Similarly, the client's certificate is also linked to the transaction as well as their IP address [86]. While this ability to log events may not be a concern for government use, there are security implications as the client requests are not protected from replay or MITM attacks [86].

OCSP Stapling

To overcome to privacy and security concerns of OCSP, OCSP Stapling introduces a TLS extension that enables the server-side to automatically request the status of a certificate and append the time-stamped, CA-signed result to the client-server handshake [86]. This protects against MITM, replay attacks, and resolves the privacy concerns of regular OCSP because the server is now responsible for fulfilling the OCSP request vice the client [86]. However, OCSP Stapling does not solve the availability issue when employed on an open network as a failure in the OCSP server would cause the service to fail [86].

Revocation Design Considerations

With regards to the ORION framework, OCSP Stapling may be the best option of the three as it improves the scalability and time inaccuracy issues in traditional CRLs while providing more security against MITM and replay attacks than regular OCSP. Availability remains an important concern and would require further research to determine its viability and effectiveness when hosted on a SDN controller. As long as the service is hosted as an internal SDN application it would be executing within the controller's container. Thus, it would be incumbent upon the controller to ensure security of the service and even then, a failure of the controller would still cause the service to shutdown. Separately, the MITM attack surface may be an area that the TLS feature in OpenFlow could facilitate in securing as it encrypts and authenticates soundbound communication from the controller to the switch(es). Pending additional experimentation, traffic from the ORION CAs and OCSP

application would theoretically be secure since the entities are again virtually hosted within the controller.

3.14.2 Causes for Revocation

In their capacity, TAs reserve the ability to revoke any certificate issued by the ORION CA only. This authority does not extend to those certificates issued on a CAC or by Purebred. Per [80], the following circumstances require certificate revocation, but is not limited to:

- Subscriber's private key is lost, stolen, or suspected of compromise;
- Subscriber is suspected of fraud or other adverse behavior;
- Subscriber is no longer affiliated with any component of its distinguished name;
- Subscriber dies, deserts, becomes a prisoner of war or is classified as missing in action;
- Subscriber leaves the DoD;
- Subscriber forgets the password and no recovery is possible;
- Subscriber violates the Subscriber agreement;
- Subscriber leaves the organization that sponsored it in the PKI, and does not return its hardware token (in the case where one has been issued);
- Subscriber or authorized party (as defined in a CPS) asks for Subscriber's certificate to be revoked;
- One of the certificates associated with a shared private key is revoked, then all certificates using that private key must also be revoked and the private key may not be used for any other certificates;
- Any certificate issued (directly or indirectly) based on a request signed with a key that subsequently determined to have been compromised at the time the request was made.

3.14.3 Revocation Requests

Should any of the revocation requirements be met, the first available TA will be immediately alerted by the individual(s) aware of the circumstance(s). The TAs will review and, pending legitimacy, approve or reject the revocation request. The revocation request procedure is as follows:

1. Notified TA authenticates and approves revocation request;
2. TA accesses Certificate Revocation App;
3. CA authenticates TA's authenticity by verifying possession of respective TA's role-based private key. CA authenticates to TA by decrypting associated request with TA's public key certificate.
4. Mutual TLS connection is established between the TA's device and the CA;
5. TA selects and confirms associated certificates for revocation within the Certificate Revocation App. Additionally, TA indicates reason for revocation;
6. CA completes revocation process.

3.14.4 Revocation due to External Circumstances

Should a Subscriber's CAC credentials be revoked for any reason, the Subscriber's corresponding ORION credentials will be automatically suspended, pending revocation, so long as there is a stable back-end connection to a DoD CRL server. In the absence of a back-end connection, an out-of-band communication method (i.e., long-haul radios) may be used to contact a tactical unit in order to direct a certificate revocation(s).

3.14.5 Emergency Revocation

The capability to conduct on-demand, remote revocation of all local ORION credentials within a SD-MANET should be available to TAs for circumstances such as those which pose a threat to safety, force preservation, actual (or suspicion of) ORION compromise, or other reasons as defined in a CPS.

3.15 Key Escrow

The risk-return trade-offs of locally replicating and storing non-ORION private decryption keys cannot be justified given the liability and the consequences of a potential compromise. However, ORION will be capable of performing key escrow functions for decryption keys created in the process of generating ORION E-mail Encryption credentials.

3.16 Credential Profile and Management

ORION will provide three Subscriber certificates (Authentication, Digital Signature, and E-mail Encryption) in addition to a Device certificate. See Figure 3.9 for a layout of a typical certificate profile.

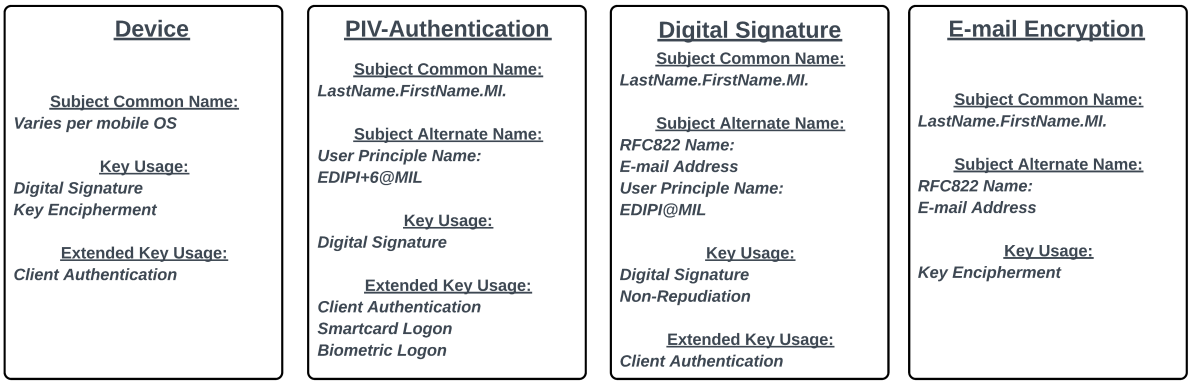


Figure 3.9. Typical ORION Subscriber Certificate Profile. Adapted from [80].

To address the concern of device certificate management in tactical environments (i.e., manual verification of certificate expirations on an individual device), the Army developed a certificate-monitoring tool called Public Key Infrastructure in a Tactical Environment (PKITE) which automatically collects and displays all device certificate information to a web database [7]. This allows soldiers to check all certificate statuses and to receive an alert on their devices when a certificate is nearing expiration [7]. The ORION Certificate Revocation App should emulate a similar function to PKITE to leverage automation and to enable scalability.

3.17 Joint and Coalition Interoperability

In a joint/coalition environment, the requirement to securely communicate amongst entities must be predicated upon a common trust anchor. To facilitate trust with entities outside of ORION, but within the DoD domain, trust is established through the concept of CA chaining. However, CA chaining does not allow for secure communications with CAs outside of the "chain". To establish trust with an external CAs, the ORION CA will execute cross certification (also known as federated PKI) in order to authenticate users belonging

to a different domain. In cross certification, the ORION CA first establishes trust with the other CA by performing an out-of-band verification of the other CA's public key certificate. Each CA will than sign the hash of the other CA's public key certificate to produce a cross certificate that can be distributed amongst its respective users. See Figure 3.10 for an overview of the cross certification process.

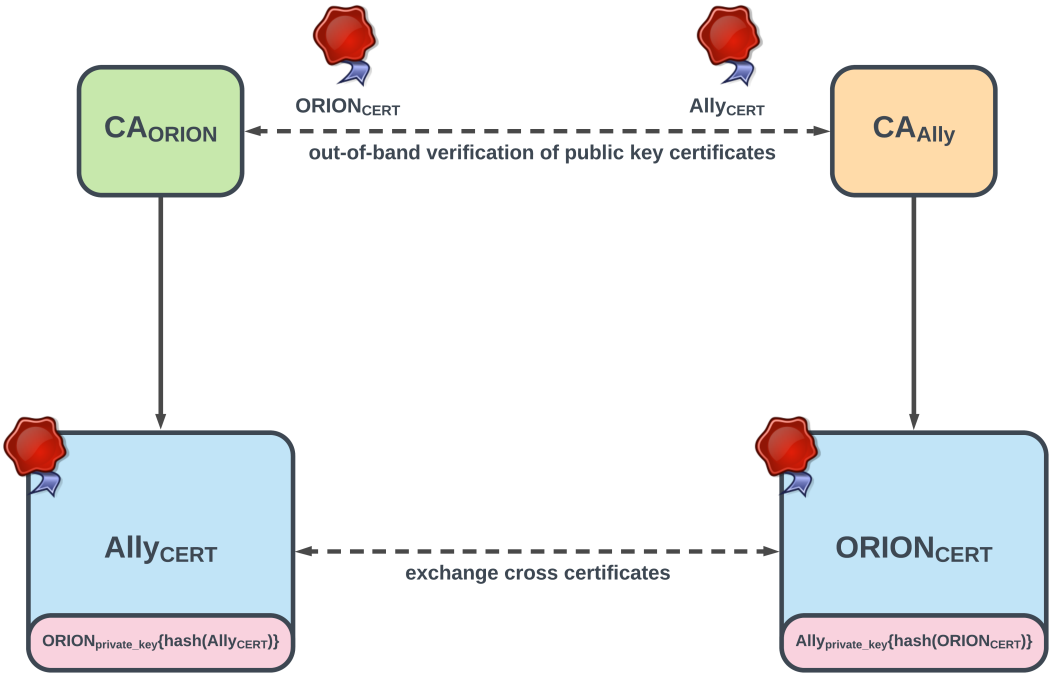


Figure 3.10. Facilitating Coalition Interoperability with CA Cross Certification

3.18 Backup and Restoration

To facilitate survivability and resiliency, the ORION system architecture includes an encrypted database for the SDN controller configuration. Periodic backups of the controller database ensures continuity in the event of a disaster or data loss. The controller database is regularly replicated to alternate CAs to ensure that there is not a single point of failure. Only designated alternate CAs are authorized to perform a full SDN infrastructure restoration after successfully decrypting the database.

3.19 Summary

The purpose of this chapter was to develop an architectural design for ORION in order to evaluate its potential as an authentication scheme for mobile ad-hoc communications in tactical edge networks. The proposed design achieves two primary objectives:

1. ORION shall have the capability to facilitate on-demand revocation of device derived credentials through a TA.
2. ORION shall have the capability to provision derived credentials without back-end support. Additionally, a method is provided to automate device credentialing (that will limit full functionality to the device or decrease the validity period of derived credentials) without the assistance of a TA.

The use of biometrics (as an alternative authentication factor) is recommended and approaches to key escrow, framework interoperability, and service recovery are discussed. The emulation of ORION's framework and functions is analyzed in the following chapter to assess its strengths, weaknesses, and viability as part of a future USMC network mobility model.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Implementation

The Marine Corps' reliance on information networks to enable timely decision-making in support of operations creates a vulnerability we must mitigate. As the Marine Corps matures the tactics, techniques, and procedures required to support a persistent information network, it must continue to explore and rehearse methods to maintain combat effectiveness in a communications degraded environment. Networks must incorporate resiliency characteristics that will support the continued operation of key functions while defensive cyberspace measures are implemented, and provide appropriate protection of information without undue sacrifice of functionality. A persistent information network is essential for MAGTF operations.

—Marine Corps Concept for Cyberspace Operations ***[87]***

4.1 Configuring the Virtual Environment

ORION's functions were emulated and evaluated in a virtual environment using a hardware device (laptop computer) and open-source software to simulate ORION's deployment in a tactical environment. Figure 4.1 provides an abstracted view of the desired virtualized implementation. The scope of this emulation experiment centered on implementation of ORION's device registration and credentialing process; specifically, the expedited mode since the full registration and credentialing process already exists in Purebred.

4.1.1 System Specifications

The following were the laptop computer system specifications used to develop ORION's proof-of-concept model:

System Operating System: Windows 10 Pro
Manufacturer/Model: Acer Aspire E5-576G
Processor: Intel Core i5-8250U @ 1.6 GHz

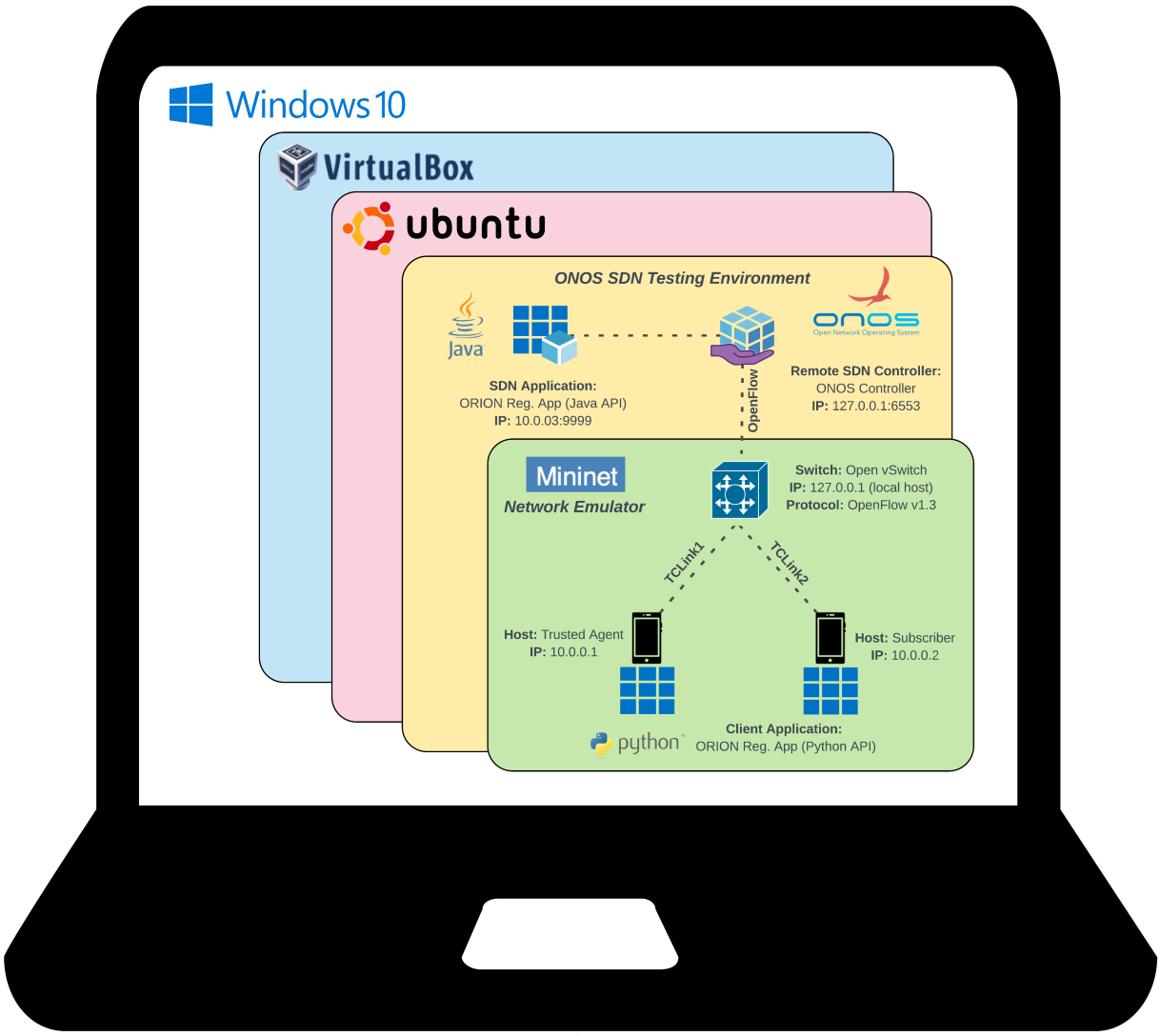


Figure 4.1. Software Abstraction of ORION's Experimental Design

Graphics Coprocessor: NVIDIA GeForce MX150 with 2.00 GB Dedicated GDDR5 VRAM
Installed Memory (RAM): 8.00 GB Dual Channel Memory DDR4
Hard Drive: 256 GB SSD
System Type: 64-bit Operating System, x64-based processor
Hypervisor: Oracle VM VirtualBox 6.0.4r128413 (Qt5.6.2)
Virtual Machine Operating System: Ubuntu 19.10
SDN Operating System: Open Network Operating System (ONOS) 2.4.0

Network Emulator: Mininet 2.2.2 [88]
Packet Analyzer: Wireshark 3.0.5 [89]
Programming Languages: Python 2.7.17 & Java 1.8.0

4.1.2 Mininet

The Mininet network emulator was chosen for use as a test bed to virtualize ORION's network environment due to its flexibility and built-in support for SDN, OpenFlow, custom topologies, and network testing. Network topologies were created using software vice hardware and behaved similarly to their discrete hardware counterparts [90]. Mininet used "a set of features built into Linux that allow a single system to be split into a bunch of smaller "containers", each with a fixed share of the processing power" [90]. This feature provided Mininet with superior performance and host scalability when compared to other "emulators which use full virtual machines" [90].

The Mininet network was composed of isolated hosts, emulated links, and emulated switches [90]. Mininet hosts were user-level Linux processes running in virtualized containers that provided network interfaces, ports, and routing capabilities [90]. Mininet links rates were configurable and managed by the Linux Traffic Control (`tc`) utility [90]. Mininet switches used the default Linux bridge or Open vSwitch to perform packet switching responsibilities across the different interfaces [90].

Mininet provided a command line interface (CLI) for managing the network and individual nodes which was called by running the `CLI()` constructor [90]. To run programs in Mininet hosts, the `cmd()` method was used to send input to the host's bash shell process [90]. The terminal emulator (`xterm`) was invoked to provide a virtualized CLI GUI experience for executing commands directly from the host computer. Additionally, Mininet allowed use of the Python API for developing prototype scripts. The RemoteController class served as a proxy for connecting Mininet to an external SDN controller running independently (i.e., in a LAN,a separate virtual machine (VM), or on a laptop) of the Mininet network [90].

Topology Design

The following Python code was used to design a simple network topology in Mininet for experimentation. It consisted of one SDN controller, one OpenFlow supported switch, and two hosts (simulating a TA and a Subscriber end device). For research purposes, such a simple topology was sufficient for emulating the client-server interaction between the Subscriber and the ORION Registration App. The TA host device was initially included for ORION Revocation App development. However, the TA host was not utilized as the app implementation was left for future work.

orion_topo.py

```
#!/usr/bin/python

from mininet.topo import Topo
from mininet.net import Mininet

class ORION(Topo):
    def __init__(self):

        # initialize topology
        Topo.__init__(self)

        # add hosts
        trusted_agent = self.addHost('ta')
        subscriber = self.addHost('client')

        # add switch
        orion_switch = self.addSwitch('switch', dpid='1775
            ')

        # add links
        self.addLink(trusted_agent, orion_switch)
        self.addLink(subscriber, orion_switch)
```

```
22
23 topos = {'orion': (lambda:ORION())}
```

Topology Build

Executing the script below on the Linux CLI resulted in a build of the Mininet network topology within the Ubuntu VM. Network Address Translation (NAT) functionality was enabled and served as an intermediary router between the SDN controller and the Mininet hosts in order to facilitate communications between the controller subnet and the Mininet subnet. The official Internet Assigned Numbers Authority (IANA) registered TCP ports for the OpenFlow protocol are 6633 and 6653 [91]. The output of the Mininet topology script:

```
$ sudo mn --custom orion_topo.py --topo orion --mac --
   controller=remote,ip=127.0.0.1,port=6653 --switch ovs,
   protocols=OpenFlow13 --nat
*** Creating network
*** Adding controller
*** Adding hosts:
client ta
*** Adding switches:
switch
*** Adding links:
(client, switch) (ta, switch)
*** Configuring hosts
client ta
*** Starting controller
c0
*** Starting 1 switches
switch ...
*** Starting CLI:
mininet>
```

4.1.3 ONOS

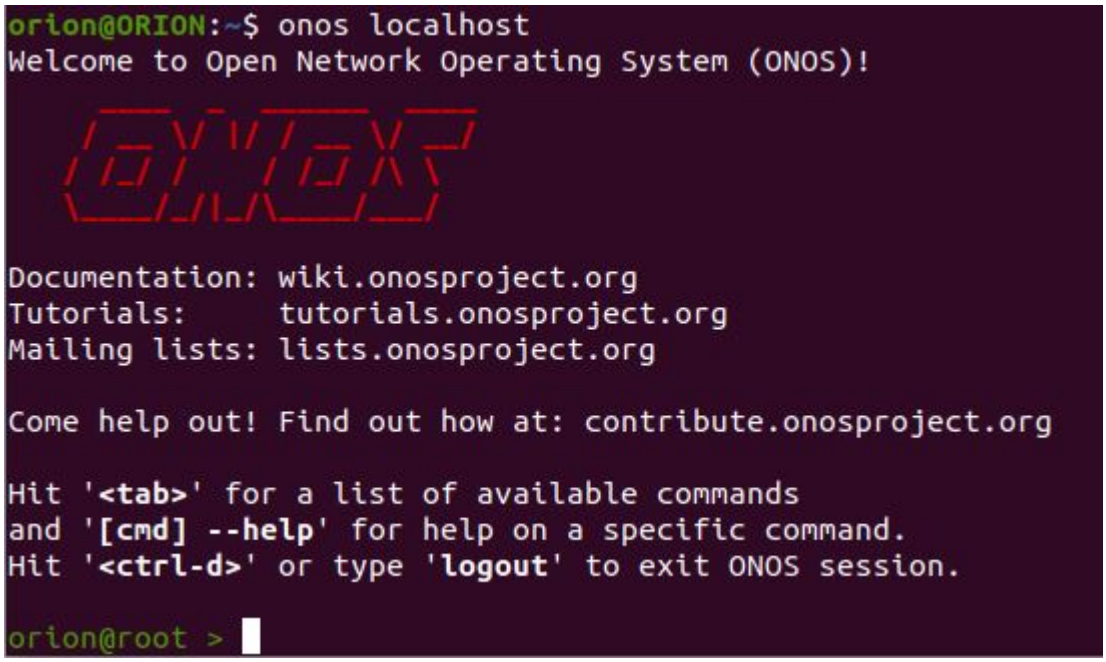


Figure 4.2. ONOS CLI

Many open source SDN controllers exist for building SDN and NFV solutions. Due to its robust support and popularity within the open source community, the ONOS controller was chosen for use in this experiment to manage the control plane of ORION's SDN and its associated network components (i.e. switches, links, software applications). The ONOS platform was analogous to a server OS that responds to client requests, and was also responsible for the management and configuration of the SDN [92]. The ONOS kernel and its system applications were constructed in Java and loaded into an Apache Karaf Open Services Gateway initiative (OSGi) container, enabling modules to be installed and executed in a single Java Virtual Machine (JVM) [92]. The ability to run ONOS within a JVM allowed it to be interoperable across a variety of OS platforms [92]. ONOS included its own CLI (which was accessed with the `onos localhost` command) as well as a web-based GUI for interaction with the platform ecosystem (by typing the following URL into a web browser: `localhost:8181/onos/ui`; see Figure 4.3). After downloading the ONOS tarbal from the online ONOS repository, Bazel (an open-source build tool developed by Google) was used to install and start the ONOS server from the CLI. Upon execution, the ONOS server log runs continuously in the foreground until the server is terminated.

(a) Login Screen

(b) Interface Layout

Figure 4.3. ONOS GUI

OpenFlow and Reactive Forwarding

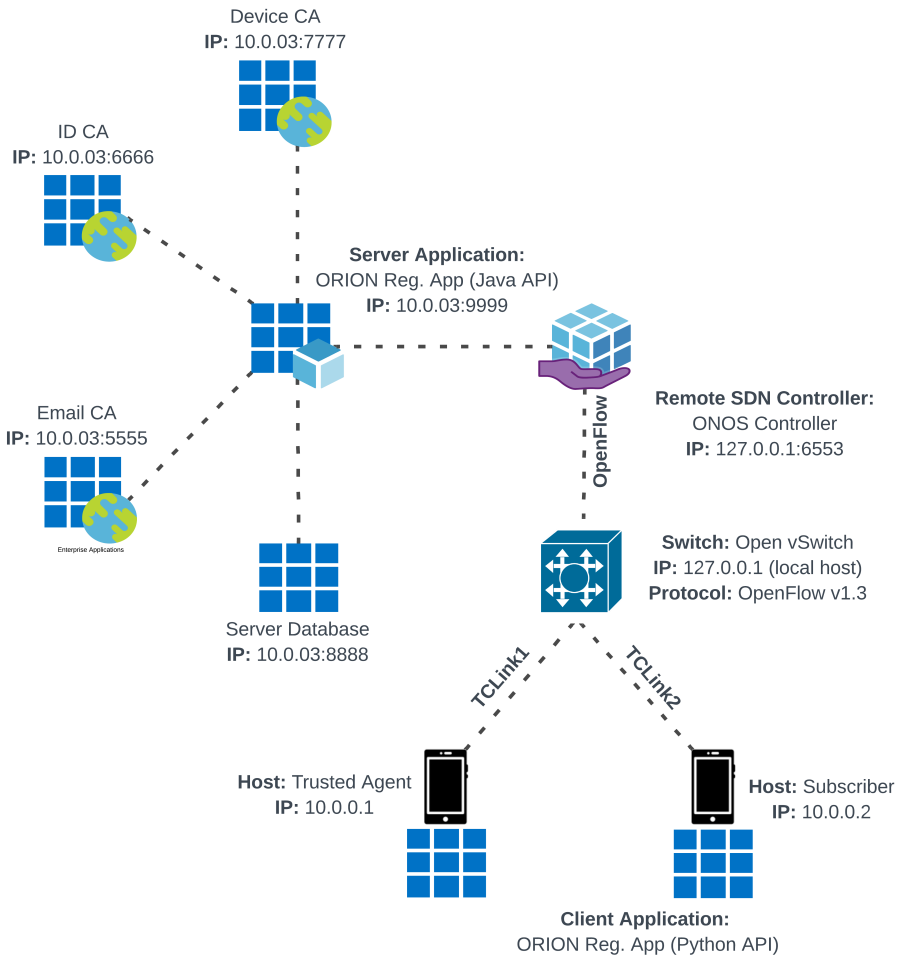


Figure 4.4. Expedited Registration and Credentialing Emulation Topology

After starting the ONOS instance and Mininet test topology, the `openflow` (OpenFlow) and `fwd` (Reactive Forwarding) applications which were preloaded on ONOS (but not installed by default) had to be activated. Since flows were not installed on the data-plane, traffic was initially unable to be forwarded appropriately. Activating both applications installed on demand flow forwarding and was achieved via the ONOS CLI with the following command:

```
onos> app activate fwd
Activated org.onosproject.openflow
Activated org.onosproject.fwd
```

To verify that OpenFlow and Reactive Forwarding was correctly installed, the following command may be typed to view all installed SDN applications:

```
onos> app -a -s
```

Next, a ping test with the `pingall()` command confirmed successful connectivity between the Mininet hosts and the NAT interface.

```
mininet> pingall
*** Ping: testing ping reachability
client -> ta nat0
ta -> client nat0
nat0 -> client ta
*** Results: 0% dropped (6/6 received)
```

4.2 ORION Registration App

Due to ONOS' dependence on the Java platform, the server-side of the ORION Registration App was constructed using the Java programming language. See Appendix A for the GitHub repository hyperlink containing all source code. First, a new ONOS application was generated using the `onos-create-app` tool which employs the Apache Maven Archetype, a build software that allows projects to be rapidly prototyped. Once the new application structure was generated, the .java file was modified to build the ORION Registration App. The `mvn clean install` command was invoked to compile the .java file into an .oar file that was then stored in the local Maven repository. The `onos-app` tool was used to install and deploy the .oar application into the ONOS instance; the exclamation mark tells ONOS

to immediately activate the ORION Registration App upon installation. The `onos-app` tool was also used to install re-compiled applications without restarting the entire ONOS server by replacing the `install` keyword with `reinstall`.

```
$ onos-app localhost install! target/FILENAME.oar
```

4.2.1 OpenSSL

The OpenSSL software library was selected for establishing the ORION PKI build. OpenSSL is a robust, commercial-grade cryptography toolkit that supports open-source implementations of the TLS and Secure Sockets Layer (SSL) protocol [93]. The OpenSSL library enjoys wide spread support from the open-source cryptography community and its application is well documented. Specifically, as it pertained to the experimentation, the included command-line binary was useful for performing a plethora of cryptographic operations such as generating private keys, signing public key certificates, encrypting/decrypting data, importing/exporting a variety of PKCS file formats, etc. In addition, the OpenSSL API offered developer-level customization capabilities for facilitating efficient scripting of cryptography related tasks.

4.2.2 Expedited Device Registration and Credentialing

The client-side of the application was constructed using the open-source Python programming language. Python was chosen for its extensive standard library, OS compatibility, and syntax readability which facilitated management of the source code and simplified the application development process. As an interpreted programming language, the source code is not compiled after editing and thus can be executed immediately. This improved the prototype development time and allowed the coding efforts to be directed toward building the application's core functionalities. See Figure 4.2.2 for an example implementation of ORION's expedited device registration and credentialing process.

Biometric Verification

To expedite the build process, the biometric verification was simulated using a simple password challenge and response method. To expound, the client provided the biometric

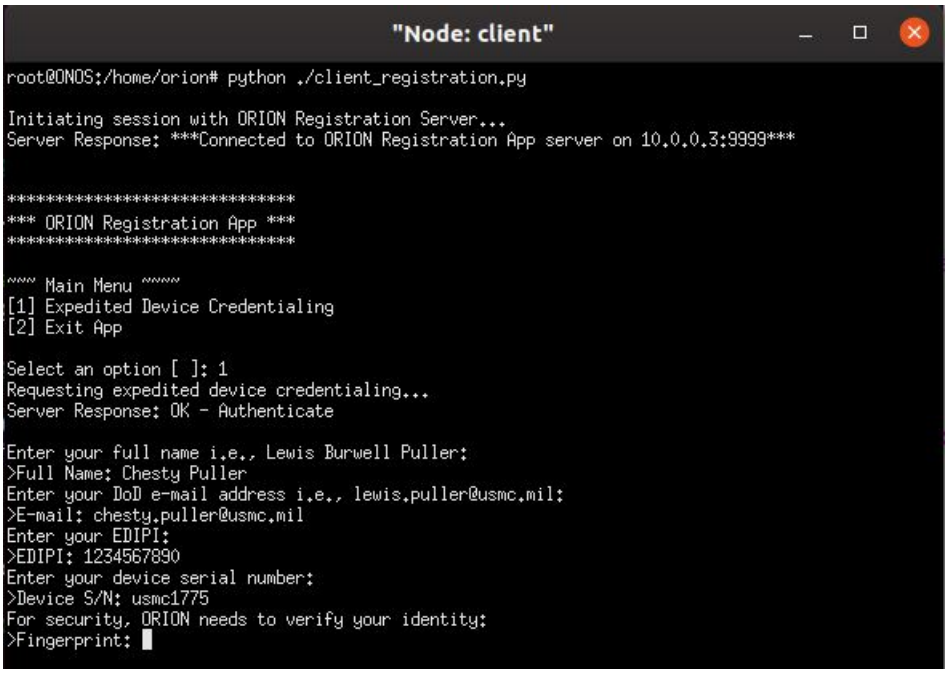


Figure 4.5. ORION Registration App Graphical User Interface

input in the form of a password. The server simulated the verification process by comparing the client's input password against the server's database stored password that was linked to the client's EDIPI. The server accepted the client's password if it matched the password stored in its database.

Process Threading

To optimize the performance of the registration and credentialing process, threading was used to control the execution and timing of concurrent processes. In contrast to a timer or sleep function, the implementation of process threading was a more elegant and efficient way of managing application execution flow. Instead of pausing the main program for a specified period of time, threading was more accurate as the main process would only wait the minimum time required for the sub-process to finish executing. For the client-side, the `thread` module was used to create a Python class (`thread.Thread`) that represented a threaded process. To start the thread, the `start()` method was invoked to initiate the process and the `join()` method was invoked to block further execution of the main program until the threaded process had been terminated. Similarly, on the server-side, the Java implementation involved the creation of a Process class in order to return an instance

of Process's subclass that could then be controlled by invoking the method to execute the process, `Runtime.getRuntime().exec()`, or invoking the method to wait for the process to terminate, `waitFor()`.

Encoding Compatibility

It was important to note the character encoding differences between Python 2.x (American Standard Code for Information Interchange (ASCII)) and the Project Object Model (POM) Extensible Markup Language (XML) file encoding (Universal Transformation Format 8-bit (UTF-8)) used by Apache Maven to build the Java project. While Python 3.x used the Unicode Standard UTF-8 encoding for file read and writes, Python 2.x used ASCII encoding by default. Since different encoding schemes generated unique character-to-byte mappings (e.g., different bytes corresponded to the same character and vice-versa), the client-side Python source code was modified to adopt the UTF-8 encoding scheme in order to prevent corrupted read and writes that would have resulted from encoding and decoding in different standards.

Simple Certificate Enrollment Protocol

SCEP was used as a means to automate the distribution of public certificates without the need for end user involvement. The use of a SCEP Gateway API Uniform Resource Locator (URL) was simulated using the ORION Registration App which provided the client with an encrypted PKCS #7 file containing the socket pair information needed to interact with the respective CA along with a hash of the CA's public key which was used to authenticate the CA once a connection was established.

PKCS #7

PKCS #7, now known as the Cryptographic Message Syntax (CMS), was used to digitally sign, digest, authenticate, or encrypt the contents of a message and serves as the foundation for the Secure/Multipurpose Internet Mail Extensions (S/MIME) standard [94]. For the experiment, PKCS #7 files, denoted by the file extension .p7, were mainly used to encapsulate and encrypt SCEP instructions using the receiver's public certificate. Additionally, PKCS #7 was also used to encrypt the client's encryption private key for transport and to sign the server's encrypted challenge message.

The following OpenSSL command was used to generate a PKCS #7 with public certificate encrypted contents:

```
$ openssl smime -encrypt -in IN_FILE -outform pem -out ENCRYPTED_FILE.p7
    PUBLIC.crt
```

The following OpenSSl command was used to decrypt a public certificate encrypted PKCS #7 file using the recipient's corresponding private key:

```
$ openssl smime -decrypt -inform pem -in ENCRYPTED_FILE.p7 -inkey PRIVATE
    .key -out DECRYPTED_FILE
```

The following OpenSSl command was used to sign the server's PKCS #7 containing a challenge message using the sender's private key:

```
$ openssl smime -sign -nodetach -in DECRYPTED_FILE -out SIGNED.p7 -
    outform pem -inkey PRIVATE.key -signer PUBLIC.crt
```

PKCS #12

PKCS #12 is a cryptography standard developed by RSA Laboratories that defines a message syntax for importing and exporting personal identity information [59]. PKCS #12 was specifically used in ORION's implementation for the transfer of public certificates. The standard, denoted by the file extension .pfx or .p12, supports a password-based privacy and security mode that was used to protect the file contents [59]. For the experiment, the password derived symmetric key was used to encrypt the file data and integrity was provided through a MAC that was also derived from the same password.

The following OpenSSl command was used to package and encrypt a public certificate for export:

```
$ openssl pkcs12 -export -nokeys -out OUT_FILE.pfx -in IN_FILE.crt -
    password pass:PASSWORD
```

The following OpenSSl command was used to extract the contents of an encrypted PKCS #12 file containing a public certificate:

```
$ openssl pkcs12 -in IN_FILE.pfx -out OUT_FILE.crt -nokeys -password pass
    :PASSWORD
```

Figure 4.2.2: ORION Registration & Credentialing Implementation

```
Initiating session with ORION Registration Server...
Server Response: ***Connected to ORION Registration App server on
    10.0.0.3:9999***


******************************
*** ORION Registration App ***
******************************

~~~ Main Menu ~~~~
[1] Expedited Device Credentialing
[2] Exit App

Select an option [ ]: 1
Requesting expedited device credentialing...
Server Response: OK - Authenticate

Enter your full name i.e., Lewis Burwell Puller:
>Full Name: Chesty Puller
Enter your DoD e-mail address i.e., lewis.puller@usmc.mil:
>E-mail: chesty.puller@usmc.mil
Enter your EDIPI:
```

```
>EDIPI: 1234567890
Enter your device serial number:
>Device S/N: usmc1775
For security, ORION needs to verify your identity:
>Fingerprint: password

Sending user and device information
Server Response: OK - Verified

Generating 2048-bit RSA private key

Generating self-signed X.509 public certificate

Creating p12 containing self-signed public certificate

Checking if server is ready for file transfer...
Server Response: OK - Send

Preparing to transfer: PULLER.CHESTY.1234567890.pfx
Completed transfer: PULLER.CHESTY.1234567890.pfx
Server Response: OK - Received

Server Response: challenge.p7 737

Establishing new file stream connection with the server...
Server Response: ***Connected to ORION Registration App server on
    10.0.0.3:8888***

Received: challenge.p7 of size: 737 bytes

File stream with the server closed...

challenge.p7 successfully decrypted with PULLER.CHESTY.1234567890.key;
    server challenge stored as decrypted.txt
```

```
signed.p7 containing decrypted.txt successfully encrypted and signed with
    PULLER.CHESTY.1234567890.key

Checking if server is ready for file transfer...
Server Response: OK - Send

Preparing to transfer: signed.p7
Completed transfer: signed.p7
Server Response: OK - Received

Server Response: scep.p7 749

Establishing new file stream connection with the server...
Server Response: ***Connected to ORION Registration App server on
   10.0.0.3:8888***

Received: scep.p7 of size: 749 bytes

File stream with the server closed...

scep.p7 successfully decrypted with PULLER.CHESTY.1234567890.key; server
   challenge stored as scep.txt

Establishing connection with ORION DEVICE CA...
Server Response: ***Connected to ORION SCEP DEVICE CA on 10.0.0.3:7777***

Sending GETACERT request to DEVICE CA...

Server Response: caCert.pfx 1112

Received: caCert.pfx of size: 1112 bytes

hash of CA.crt: c7820aeb
```

```
DEVICE CA public certificate hash successfully verified

Connection with DEVICE CA closed...

Generating RSA private key, 2048 bit long modulus (2 primes)
.....................................................................+++++
.............+++++
e is 65537 (0x010001)
Done.

Generating CA Certificate Signing Request...

Done.

Establishing connection with ORION DEVICE CA...
Server Response: ***Connected to ORION SCEP DEVICE CA on 10.0.0.3:7777***

Server Response: OK - Send

Preparing to transfer: PULLER.CHESTY.1234567890.csr
Completed transfer: PULLER.CHESTY.1234567890.csr
Server Response: OK - Received

Receiving Device Certificate from ORION DEVICE CA...

Server Response: deviceCert.pfx 1072

Received: deviceCert.pfx of size: 1072 bytes

Device Certificate extracted from P12

Connection with DEVICE CA closed...

Server Response: challenge2.p7 688
```

```
Establishing new file stream connection with the server...
Server Response: ***Connected to ORION Registration App server on
    10.0.0.3:8888***

Received: challenge2.p7 of size: 688 bytes

File stream with the server closed...

challenge2.p7 successfully decrypted with PULLER.CHESTY.1234567890.device
    .key; server challenge stored as decrypted2.txt

signed2.p7 containing decrypted2.txt successfully encrypted and signed
    with PULLER.CHESTY.1234567890.device.key

Checking if server is ready for file transfer...
Server Response: OK - Send

Preparing to transfer: signed2.p7
Completed transfer: signed2.p7
Server Response: OK - Received

Authenticate your identity to complete the process:
>Fingerprint: password

Processing.....
Server Response: OK - Verified

+++++ ORION Device Enrollment Complete - Total Elapsed Time:
    1.54713320732 Seconds +++++

+++++ Device serial: usmc1775 is now credentialed to user EDIPI:
    1234567890 +++++
```

```
Press [Enter] to continue to expedited credentialing

For security, ORION needs to verify your identity:
>Fingerprint: password
Server Response: OK - Verified

Server Response: scep.p7 701

Establishing new file stream connection with the server...
Server Response: ***Connected to ORION Registration App server on
    10.0.0.3:8888***

Received: scep.p7 of size: 701 bytes

File stream with the server closed...

scep.p7 successfully decrypted with PULLER.CHESTY.1234567890.device.key;
    server challenge stored as scep.txt

Establishing connection with ORION ID CA...
Server Response: ***Connected to ORION SCEP ID CA on 10.0.0.3:6666***

Sending GETACERT request to ID CA...

Server Response: IDcaCert.pfx 1144

Received: IDcaCert.pfx of size: 1144 bytes

hash of IDCA.crt: c1fc9a8a
ID CA public certificate hash successfully verified

Connection with ID CA closed...

Generating RSA private key, 2048 bit long modulus (2 primes)
```

```
.............................+++++
............+++++
e is 65537 (0x010001)
Done.

Generating CA Certificate Signing Request...

Done.

Establishing connection with ORION ID CA...
Server Response: ***Connected to ORION SCEP ID CA on 10.0.0.3:6666***

Server Response: OK - Send

Preparing to transfer: PULLER.CHESTY.1234567890.piv.csr
Completed transfer: PULLER.CHESTY.1234567890.piv.csr
Server Response: OK - Received

Receiving PIV Authentication Certificate from ORION ID CA...

Server Response: pivCert.pfx 1088

Received: pivCert.pfx of size: 1088 bytes

PIV Authentication Certificate extracted from P12

Connection with ID CA closed...

+++++ PIV Authentication Credential Received +++++

Server Response: scep.p7 701

Establishing new file stream connection with the server...
Server Response: ***Connected to ORION Registration App server on
```

```
    10.0.0.3:8888***

Received: scep.p7 of size: 701 bytes

File stream with the server closed...

scep.p7 successfully decrypted with PULLER.CHESTY.1234567890.device.key;
    server challenge stored as scep.txt

Establishing connection with ORION EMAIL CA...
Server Response: ***Connected to ORION SCEP EMAIL CA on 10.0.0.3:5555***

Sending GETACERT request to EMAIL CA...

Server Response: EMAILcaCert.pfx 1152

Received: EMAILcaCert.pfx of size: 1152 bytes

hash of EMAILCA.crt: ca0664bc
EMAIL CA public certificate hash successfully verified

Connection with EMAIL CA closed...

Generating RSA private key, 2048 bit long modulus (2 primes)
.....+++++
....+++++
e is 65537 (0x010001)
Done.

Generating CA Certificate Signing Request...

Done.

Establishing connection with ORION EMAIL CA...
```

```
Server Response: ***Connected to ORION SCEP EMAIL CA on 10.0.0.3:5555***

Server Response: OK - Send

Preparing to transfer: PULLER.CHESTY.1234567890.signature.csr
Completed transfer: PULLER.CHESTY.1234567890.signature.csr
Server Response: OK - Received

Receiving Digital Signature Certificate from ORION EMAIL CA...

Server Response: signatureCert.pfx 1088

Received: signatureCert.pfx of size: 1088 bytes

Digital Signature Certificate extracted from P12

Connection with EMAIL CA closed...

+++++ Digital Signature Credential Received +++++

Server Response: scep.p7 701

Establishing new file stream connection with the server...
Server Response: ***Connected to ORION Registration App server on
    10.0.0.3:8888***

Received: scep.p7 of size: 701 bytes

File stream with the server closed...

scep.p7 successfully decrypted with PULLER.CHESTY.1234567890.device.key;
    server challenge stored as scep.txt

Establishing connection with ORION EMAIL CA...
```

```
Server Response: ***Connected to ORION SCEP EMAIL CA on 10.0.0.3:5555***

Sending GETACERT request to EMAIL CA...

Server Response: EMAILcaCert.pfx 1152

Received: EMAILcaCert.pfx of size: 1152 bytes

hash of EMAILCA.crt: ca0664bc
EMAIL CA public certificate hash successfully verified

Connection with EMAIL CA closed...

ORION Registration App is generating new key pair and Encryption CSR...

Server Response: encryptionCert.pfx

Establishing file connection with ORION Registration App...
Server Response: ***Connected to ORION Registration App server on
    10.0.0.3:8888***

Sending Encryption Certificate request

Server Response: encryptionCert.pfx 1088

Received: encryptionCert.pfx of size: 1088 bytes

Encryption Certificate received: PULLER.CHESTY.1234567890.encryption.crt
Closing file connection...

Server Response: encryptionKey.p7 3102

Establishing new file stream connection with the server...
Server Response: ***Connected to ORION Registration App server on
```

```
    10.0.0.3:8888***

Received: encryptionKey.p7 of size: 3102 bytes

File stream with the server closed...

encryptionKey.p7 successfully decrypted with PULLER.CHESTY.1234567890.
    device.key; stored as encrypted.key

writing RSA key
encrypted.key successfully decrypted: PULLER.CHESTY.1234567890.encryption
    .key

+++++ Encryption Key/Credential Received +++++

*** Subscriber Credentialing Completed - Total Elapsed Time:
    2.08721208572 Seconds ***
```

4.3 Summary

The purpose of this chapter was to implement the architectural design of ORION for further evaluation. The system configuration and specifications of the experiment were outlined and the scope of the implementation was established. Specifically, the expedited mode of the ORION Registration App was developed for both device registration and user credentialing. Mininet was used to emulate an experimental ORION network topology which included two hosts, a switch, and two links that connected the hosts to the switch. The ONOS controller was selected to manage the SDN environment and all server-side applications were developed and virtualized on the controller using Java. The client-side application was prototyped in Python. OpenSSL was used to build out the PKI authentication scheme and to execute any cryptography-related tasks. Relevant Tactics (Tools), Techniques, and Procedures (TTP) used to facilitate and optimize the implementation were also discussed.

CHAPTER 5:
Evaluation

MCDP 1 points out that a significant advantage can be gained by being first to exploit a development in the art and science of war. A military that is slow to exploit technological advances and adapt new ways of fighting opens itself to catastrophic defeat. As we continue to reap the benefits of technological progress in many warfighting areas, we must capture the full potential inherent in automation. Automation can mitigate risk, reducing the exposure of humans to harm, and reduce the workload on personnel. As machines advance from performing repetitive tasks to dynamic workloads, it will free people to focus on the things they do uniquely or best. The challenge, as machines become more capable and autonomous, is how to put people and things together in the most effective pairings for the mission at hand.

—Marine Corps Operating Concept [95]

The evaluation is divided into three sections. First, methods of validating correct behavior and function of the ORION Registration App are demonstrated. Next, functional timing test are conducted under adversarial network conditions characterized by low bandwidth and high packet loss probabilities; the details of the results are highlighted and discussed. Finally, a summary of the evaluation is presented to outline the expected performance of the ORION framework and its deployment in tactical edge networks.

5.1 Validation of Correctness

This section discusses the methods used to perform unit testing and to evaluate the expected behavior of the ORION Registration App. The following tests were performed using OpenSSL to view the details of the desired file in order to check for correctness.

1. Check the validity of a private key:

```
$ openssl rsa -in PULLER.CHESTY.1234567890.device.key -check
RSA key ok
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAre4qpoShVyxWyYFP/YnRINKj9Q13AnnWUu4ktS2gZiZj6eRQ
D/bC0km99BpM+vpfbmpT9nMQOC8hdIOSc6xjy6hIkKZF8JnUnfyoF9rire69VZkz
EkTqgGYkGh6cJLlF6JTYE2jUFF0FskibEP5nTvrIncFdiEMgI7YqoN826uts+CTf
CgI0F1qzvWJZSIwPpdG6xA0J44DpgzvfU8aiojp3HNoUvNhu2HPmBanxYwwxhq5x
k351I6fj6pP/EqCF2zOF4NCCg41PUHKnkpeU5ptUdGw2pf+38kWF32K/+kbkXZ/6
Dfb+gc33GPYGtyjZjVdPMpAQ7B509Bp7drSZnQIDAQABAoIBAEByq+Mf6hi/GTl5
bQZpdtBOvOoknyCPvNbl3zwzN+gh+YvFSdVgfjglkvlMdZHaDFqBk7MSV8Q50o+1
4SJzklb4eEsBEZFhruMLp82PEceZWxbIuZ/fYXVKY7458Wm6LSlNXfNzOIKDJYU2
Wh8l5Wnii0vJRrsjBFm3Onx5uYO6NDeeMUfEKUZLiSm3ZDQy3wOYjiYwYVPoXxbI
FAfsEHGIkoaVWH9kJQqm1LqZ4LFWreq4i7Xsy0WfMqZu6nJgEBeH3Pjroz0bb9/0
VpGQ3uCg4VNvie2O4ZISCw6OtmauQI6nDnQyKXoUbu0tIRPUAxRLAU/QQmzR8V4k
18QHRYECgYEA1RxVNA1HZkwA9RTVGDdKZeGDvMHrmDqY3R9lg+lhnLd7UkH3LF70
UgUGkOBw1ift2lxHHokfBJ2Renu7lbzoUkUaZmBto9jInf/avfApVIX+PBiZoorN
2G8oP4CaOE/rJCOFTI8jAe/0U1feCqUDSjZKYw7fcwirXamtWc7qLHUCgYEA0O85
oDa8u41QOHe3O6HRKwbkWAHkpK39phmhiJPClt9kkTFU2r0kHltK7iglt3hv28Sr
ULH76UQ83awOa0LNTS+3KYd2xXFkXw1VbgVcqbJLhbI3DLyFFED/30ORBvR7eBGL
ADGyZ0soL/ytENl2PPu8R+ijep/6NyBcn695s4kCgYBIkPlCXDMv1jmNOqbFNC6V
OpFNOQssLufWg9ZhcJrYbnIDtIqiBUNUmnl9468IfVg0vq9I6pAxViYveuqJNmY1
fWpb6gHrhOnrQzAM9TZdnbLQa+AExRcK1+7wkK3y9uOuzpmZVDQreLXNm2H0ZfQU
1R/HU00rMoyHvDtXFWoP8QKBgEWUNUuejvGimOGFNLceJ2s9y1Nnpf4Vl0/2Xy39
jyLxWCtmE00ZCes2Gdj/87eK2y2D7E1eSHcha2ejAdZEk9wtlMC2xR3xqixhY17V
WEuG/dGTTPPn4CWjk9KFTMnVJz0QEotJAZGWnNaGPhhLYy3h44zK845Fsm0jUNU4
+b2JAoGBAJkHPfi/cxw6bYWSHoAhdbDcLL+PDBOuOu8OUutbrwZmXeALTXf8TKyb
xNCaWrdIBa1x+ux5xOmS0PLLRmmsJzv7nAPl2ZxyMP2r06J3+mDAlxOUQdrufyha
RMpcbi81ZHnwgWhy9AJv2beFgH1QTmxMLBPUekXg2qKI9NngyZa7
-----END RSA PRIVATE KEY-----
```

2. Check the validity of a CSR:

```
$ openssl req -text -noout -verify -in PULLER.CHESTY.1234567890.
   csr
verify OK
Certificate Request:
   Data:
      Version: 1 (0x0)
      Subject: C = US, OU = USMC, O = U.S.GOVERNMENT, CN =
         PULLER.CHESTY.1234567890, emailAddress = chesty.
         puller@usmc.mil
      Subject Public Key Info:
         Public Key Algorithm: rsaEncryption
            RSA Public-Key: (2048 bit)
            Modulus:
               00:c1:94:11:a0:e2:8e:4c:90:3d:f2:eb:d9:c3:6e:
               b2:c3:56:a2:a3:e5:3c:c9:40:66:4d:5e:05:56:c6:
               4a:63:52:5a:7a:19:02:55:c2:82:01:4a:43:4d:e0:
               aa:6e:6a:e3:f0:60:bb:bf:fd:19:0e:7a:4d:cb:92:
               20:57:bb:07:9c:7a:3e:4e:9d:15:8a:c3:45:b4:20:
               a8:2c:53:a5:96:f7:54:cf:cf:cf:aa:ac:7c:7c:a2:
               f1:aa:5e:95:34:01:23:2f:4a:f7:87:a1:7b:d8:27:
               3b:71:88:9b:6b:60:56:cd:0b:a3:98:73:7e:cf:9d:
               37:7b:c2:16:df:11:cb:82:10:d9:eb:7c:87:62:f3:
               9d:a7:21:63:fb:be:8c:1a:ec:d6:19:67:90:73:f8:
               81:26:50:ef:2a:69:ea:29:2e:42:41:47:ca:3b:bb:
               ee:16:3b:b6:13:7e:21:23:31:60:ee:5f:6c:50:fc:
               73:f3:40:82:3c:b4:60:02:99:71:9c:91:3f:56:93:
               ec:0f:a7:80:5a:03:db:8b:42:9f:9c:b5:0f:e7:ce:
               4a:e8:e2:fe:a0:8c:d7:d6:bb:d7:fc:e7:29:b4:da:
               e4:ea:d2:8d:e3:16:6f:24:e9:5b:55:6d:fe:fd:44:
               f2:2a:31:87:7a:6d:58:fc:89:a8:df:89:c8:0b:8b:
               e0:c7
            Exponent: 65537 (0x10001)
      Attributes:
```

```
        Requested Extensions:
            X509v3 Key Usage:
                Digital Signature
    Signature Algorithm: sha256WithRSAEncryption
         44:8d:79:06:ee:51:27:13:72:67:8c:ed:52:09:5e:d9:c2:9e:
         4e:b0:f1:08:b3:9d:5f:73:d6:4a:e0:27:df:4a:23:04:e7:60:
         9d:6c:eb:47:b3:39:cb:b0:88:47:bf:cf:cc:b2:18:24:42:e1:
         38:bd:df:80:bf:6d:03:ef:51:51:4b:4b:83:cb:97:04:39:ce:
         73:2c:d4:df:1c:99:43:ae:f9:37:e6:2a:cb:c4:b8:00:2a:62:
         c3:b5:32:a3:9b:1a:52:24:39:b1:02:8d:1e:d2:49:19:5a:02:
         07:a2:d6:00:0d:8c:de:67:1b:67:78:15:40:39:70:f1:99:d3:
         04:d4:c9:17:34:17:cd:b2:da:29:b2:ad:f4:bd:04:39:e6:ce:
         ac:88:82:d2:67:be:0f:08:fd:60:1f:0c:c4:d5:a3:3d:1d:fa:
         c8:e5:34:31:19:9d:3e:91:a7:2d:06:bf:68:fc:05:d1:2e:6f:
         39:7a:16:ed:b1:5a:2b:91:0c:02:44:c7:b3:7d:10:2b:a7:0b:
         a2:88:57:fa:95:a4:e4:b6:ef:5b:fe:f8:1e:73:53:21:9f:2b:
         80:ea:6f:97:7a:ce:10:21:5e:a3:f1:fd:07:e3:12:a8:b5:94:
         cc:22:02:8f:bd:f9:36:c0:cf:42:af:ea:e2:be:2e:39:be:40:
         b8:02:19:76
```

3. Check the validity of a public certificate:

```
$ openssl x509 -in PULLER.CHESTY.1234567890.device.crt -text -
    noout
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            5d:9a:7d:5d:00:9a:e6:28:69:6d:f7:0a:68:b2:f5:52:21:90:
                f7:1e
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = US, O = U.S. Government, OU = USMC, CN = DOD
            DEVICE CA
        Validity
            Not Before: Jun 9 04:34:57 2020 GMT
```

```
            Not After : Jun 9 04:34:57 2021 GMT
        Subject: C = US, OU = USMC, O = U.S.GOVERNMENT, CN =
            PULLER.CHESTY.1234567890, emailAddress = chesty.
            puller@usmc.mil
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:c1:94:11:a0:e2:8e:4c:90:3d:f2:eb:d9:c3:6e:
                    b2:c3:56:a2:a3:e5:3c:c9:40:66:4d:5e:05:56:c6:
                    4a:63:52:5a:7a:19:02:55:c2:82:01:4a:43:4d:e0:
                    aa:6e:6a:e3:f0:60:bb:bf:fd:19:0e:7a:4d:cb:92:
                    20:57:bb:07:9c:7a:3e:4e:9d:15:8a:c3:45:b4:20:
                    a8:2c:53:a5:96:f7:54:cf:cf:cf:aa:ac:7c:7c:a2:
                    f1:aa:5e:95:34:01:23:2f:4a:f7:87:a1:7b:d8:27:
                    3b:71:88:9b:6b:60:56:cd:0b:a3:98:73:7e:cf:9d:
                    37:7b:c2:16:df:11:cb:82:10:d9:eb:7c:87:62:f3:
                    9d:a7:21:63:fb:be:8c:1a:ec:d6:19:67:90:73:f8:
                    81:26:50:ef:2a:69:ea:29:2e:42:41:47:ca:3b:bb:
                    ee:16:3b:b6:13:7e:21:23:31:60:ee:5f:6c:50:fc:
                    73:f3:40:82:3c:b4:60:02:99:71:9c:91:3f:56:93:
                    ec:0f:a7:80:5a:03:db:8b:42:9f:9c:b5:0f:e7:ce:
                    4a:e8:e2:fe:a0:8c:d7:d6:bb:d7:fc:e7:29:b4:da:
                    e4:ea:d2:8d:e3:16:6f:24:e9:5b:55:6d:fe:fd:44:
                    f2:2a:31:87:7a:6d:58:fc:89:a8:df:89:c8:0b:8b:
                    e0:c7
                Exponent: 65537 (0x10001)
    Signature Algorithm: sha256WithRSAEncryption
        42:6f:4f:1b:6f:a9:8b:d7:ab:40:48:27:c0:b0:e1:f4:e0:c9:
        c7:16:9f:49:7a:9e:d7:c6:97:ff:b2:8b:b1:ea:40:f8:ab:17:
        2a:2f:a9:72:02:ab:ac:42:8c:a1:73:75:d7:45:2e:20:eb:38:
        f8:a0:c9:b2:08:d4:4f:22:49:0e:48:f4:a0:29:b6:08:fa:ae:
        53:1e:00:02:7c:68:ea:f2:c9:9a:5c:4d:8a:68:59:63:d1:eb:
        6a:8d:a3:b2:8f:00:19:8f:ca:10:f4:85:e2:5f:cf:a7:4f:a0:
```

```
        9c:25:3a:ec:41:e3:3f:ff:f0:6f:b0:c6:56:9f:40:f4:5b:35:
        d6:f5:86:00:0f:33:c9:d2:8a:42:de:9c:81:57:6e:cf:33:23:
        41:1b:87:67:c4:49:a2:b7:66:75:3f:d9:d0:da:dd:a8:8d:32:
        25:d2:ed:ff:1c:49:c5:ef:65:7c:5e:b4:c6:25:c9:31:54:c2:
        df:c1:49:b8:78:23:e9:97:c1:80:de:0d:14:2f:43:a3:5e:fc:
        06:49:c3:df:78:01:1b:b4:57:b9:bf:6e:73:83:59:da:e1:1e:
        97:8a:58:b4:85:4b:7d:33:50:8c:95:fd:5d:ea:40:7d:7a:5c:
        5b:cb:4b:26:ea:ac:7e:c0:e4:0d:28:71:82:87:51:da:b3:a6:
        e8:48:99:7d
```

4. Check the validity of a PKCS #12 file:

```
$ openssl pkcs12 -info -in deviceCert.pfx
Enter Import Password:
MAC: sha1, Iteration 2048
MAC length: 20, salt length: 8
PKCS7 Encrypted data: pbeWithSHA1And40BitRC2-CBC, Iteration 2048
Certificate bag
Bag Attributes: <No Attributes>
subject=C = US, OU = USMC, O = U.S.GOVERNMENT, CN = PULLER.
   CHESTY.1234567890, emailAddress = chesty.puller@usmc.mil

issuer=C = US, O = U.S. Government, OU = USMC, CN = DOD DEVICE
   CA

-----BEGIN CERTIFICATE-----
MIIDVDCCAjwCFF2afV0AmuYoaW33Cmiy9VIhkPceMA0GCSqGSIb3DQEBCwUAME4x
CzAJBgNVBAYTAlVTMRgwFgYDVQQKDA9VLlMuIEdvdmVybm1lbnQxDTALBgNVBAsM
BFVTTUMxFjAUBgNVBAMMDURPRCBERVZJQ0UgQ0EwHhcNMjAwNjA5MDQzNDU3WhcN
MjEwNjA5MDQzNDU3WjB/MQswCQYDVQQGEwJVUzENMAsGA1UECwwEVVNNQzEXMBUG
A1UECgwOVS5TLkdPVkVSTk1FTlQxITAfBgNVBAMMGFBVTExFUi5DSEVTVFkuMTIz
NDU2Nzg5MDElMCMGCSqGSIb3DQEJARYWY2hlc3R5LnB1bGxlckB1c21jLm1pbDCC
ASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAMGUEaDijkyQPfLr2cNussNW
oqPlPMlAZk1eBVbGSmNSWnoZAlXCggFKQ03gqm5q4/Bgu7/9GQ56TcuSIFe7B5x6
Pk6dFYrDRbQgqCxTpZb3VM/Pz6qsfHyi8apelTQBIy9K94ehe9gnO3GIm2tgVs0L
```

```
o5hzfs+dN3vCFt8Ry4IQ2et8h2LznachY/u+jBrs1hlnkHP4gSZQ7ypp6ikuQkFH
yju77hY7thN+ISMxYO5fbFD8c/NAgjy0YAKZcZyRP1aT7A+ngFoD24tCn5y1D+fO
Suji/qCM19a71/znKbTa5OrSjeMWbyTpW1Vt/v1E8ioxh3ptWPyJqN+JyAuL4McC
AwEAATANBgkqhkiG9w0BAQsFAAOCAQEAQm9PG2+pi9erQEgnwLDh9ODJxxafSXqe
18aX/7KLsepA+KsXKi+pcgKrrEKMoXN110UuIOs4+KDJsgjUTyJJDkj0oCm2CPqu
Ux4AAnxo6vLJmlxNimhZY9Hrao2jso8AGY/KEPSF4l/Pp0+gnCU67EHjP//wb7DG
Vp9A9Fs11vWGAA8zydKKQt6cgVduzzMjQRuHZ8RJordmdT/Z0NrdqI0yJdLt/xxJ
xe9lfF60xiXJMVTC38FJuHgj6ZfBgN4NFC9Do178BknD33gBG7RXub9uc4NZ2uEe
l4pYtIVLfTNQjJX9XepAfXpcW8tLJuqsfsDkDShxgodR2rOm6EiZfQ==
-----END CERTIFICATE-----
```

5. Confirm that a public certificate matches its corresponding CSR and/or private key:

```
$ sudo sudo openssl x509 -noout -modulus -in PULLER.CHESTY
    .1234567890.device.crt | openssl sha256
(stdin)= 16
    c39c0fa336d7333d804458aa1aeaf1184a192115da9db26cfa36170f2a1d0d

$ sudo openssl x509 -noout -modulus -in PULLER.CHESTY
    .1234567890.device.crt | openssl sha256
(stdin)= 16
    c39c0fa336d7333d804458aa1aeaf1184a192115da9db26cfa36170f2a1d0d

$ sudo openssl req -noout -modulus -in PULLER.CHESTY.1234567890.
    csr | openssl sha256
(stdin)= 16
    c39c0fa336d7333d804458aa1aeaf1184a192115da9db26cfa36170f2a1d0d
```

5.1.1 Traffic Analysis

Wireshark was used to analyze the TCP traffic between the client and the ORION Registration App server. As depicted in Figure 5.1, network traffic between the client and server was not encrypted and the TCP communications could be viewed. This may

be a security concern as network transactions were visible, inviting the possibility of exploitation and malicious network-related attacks (i.e., spoofing, session hijacking, MITM, replay). However, it was assumed for purposes of this analysis that any data traversing a DoD network would be encrypted using NSA-approved Type 1 encryption hardware (i.e., TACLANE-micro KG-175D) and/or Commercial Solutions for Classified (CSfC)-approved, commercial-grade technology in accordance with NIST and federal government IA standards. While important to note, the topic of this subject was outside the scope of the ORION framework, and further study on viable DoD-approved encryption solutions was thus left to future work.

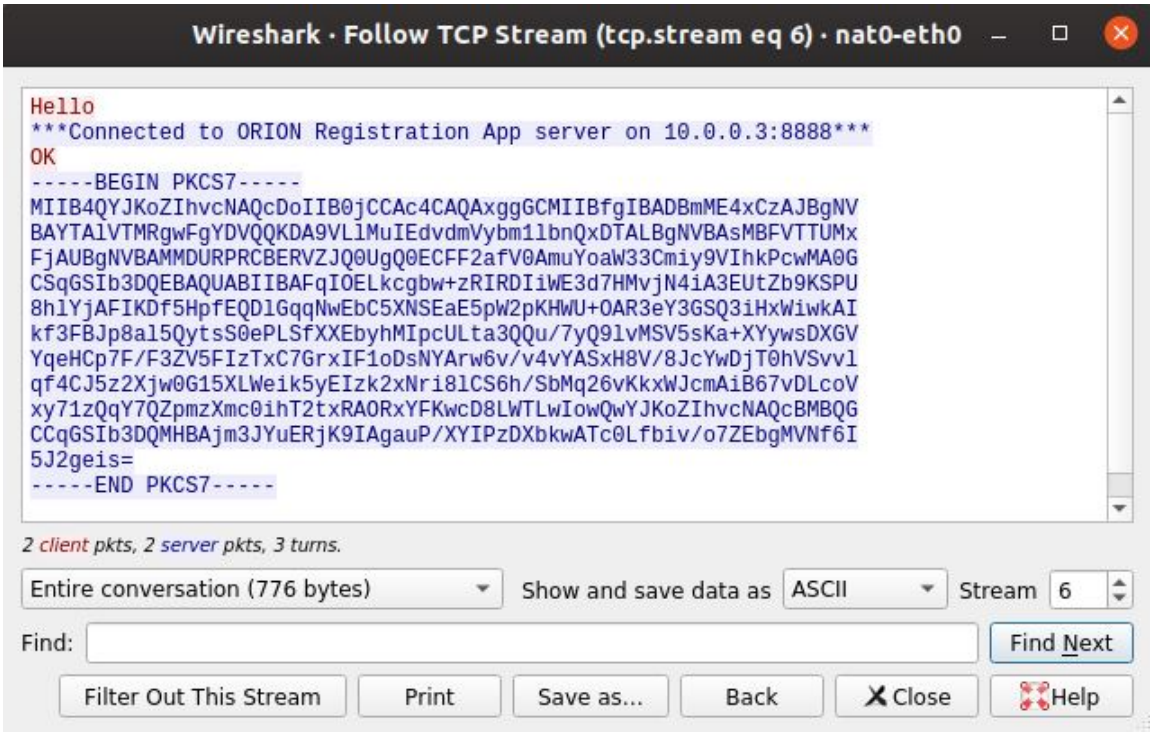


Figure 5.1. Wireshark Capture of Application TCP Stream

5.2 Network Performance Analysis

In order to model and evaluate the typical network characteristics serving a forward-deployed Marine unit, the Very Small Aperture Terminal (VSAT)-Small satellite communication (SATCOM) system was chosen as a candidate for identifying realistic network link thresholds. While many alternatives exist, the VSAT-Small is often employed by small

operational units as a means of reachback for access to the Marine Corps Enterprise Network (MCEN), making it a suitable candidate. With the understanding that bandwidth allocation and link performance are dependent upon a variety of factors (i.e., priority, mission, weather, beam angle), we considered the following network scenario to be a generalized baseline for a typical VSAT-Small link: 2Mbps bandwidth, 700ms latency, and 1% loss rate. By applying this baseline, the expedited registration and credentialing mode of the ORION Registration App was evaluated under a combination of varying bandwidth settings {128 kbps, 256 kbps, 512 kbps, 1 Mbps, 2Mbps} coupled with a fixed packet loss rate of 1%, and a latency of 700±50ms. Furthermore, the app was deployed in a highly constrained network condition to examine its feasibility and performance in a high loss (10%), high delay (700±50ms), low bandwidth (128 kbps) environment.

5.2.1 Traffic Control

The Linux tool `tc` was used to impair the network link conditions (e.g., bandwidth, loss, latency), and was configured within the Mininet client's `xterm` CLI. While the Mininet CLI offered traffic control capabilities, the Mininet `tc` feature was not as granular and customizable as the built-in `tc`. Of note, both `tc` packages could not be simultaneously used, and doing so would result in the following: `Error: Exclusivity flag on, cannot modify.` `tc` modifies the Traffic Control in the Linux kernel to shape transmission rates and to manage the scheduling of transmission packets [96]. By enabling traffic policing with the `ingress` queuing discipline, the client's expected link conditions were applied for automated testing. Additionally, `NetEm`, an internal feature of `tc`, allowed for the modification of delay and packet loss to be activated on the client's network interface.

Packet loss is typically attributed to transient events such as network congestion, buffer overflows, or random transmission bit errors. The pattern of packet loss is then generalized to be non-uniform in distribution, meaning that equal-probability events for defining packet loss would be inadequate in generating realistic test data. Rather, packet loss is more bursty in nature. A more complete method for emulating complex packet loss mechanisms incorporates the Gilbert-Elliot loss model and can be enabled on `NetEm` by specifying the `gemodel` parameter. The Gilbert-Elliot loss model states that packet loss can be more appropriately modeled as a stochastic 2-state Markov process to capture the observed loss pattern [97]. The Gilbert-Elliot model considers a good (`G`) and a bad (`B`) state where each

state has a probability for independent error events (1-p or 1-r) as dictated by its respective state dependent error rate (1-k or 1-h) [97]. See Figure 5.2 for a diagram of the Gilbert-Elliot model depicted as a 2-state Markov process. Network analysis using `gemodel` would likely produce more accurate results but was out of the scope for purposes of this evaluation. Thus, we leave further research in this area to future work.

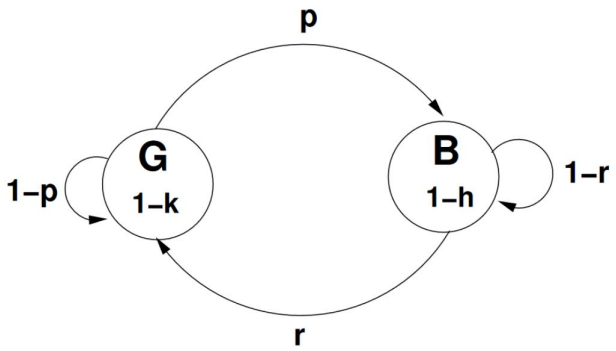


Figure 5.2. Modeling Gilbert-Elliot Loss Model as a 2-State Markov Chain. Source: [97].

For the evaluation, the `loss` parameter was enabled on `NetEm` using two input parameters after the `loss` keyword. The first parameter specified the overall packet loss rate; the second parameter specified the average number of consecutive losses correlated to the previous lost packet. This method of correlation was generalized to emulate packet burst losses. Network latency was emulated using the `delay` parameter in `NetEm`. To make the effects of delay more realistic, four parameters were used. The first parameter specified an overall delay of 700ms. The second parameter specified that delay value be uniformly distributed between {650ms-750ms}. The third parameter specified that the current packet delay would be 25% correlated with the delay of the previous packet. The `distribution` parameter was set to `normal` to model a non-uniformly distributed delay.

The following were example commands used to implement the desired bandwidth, loss, and latency characteristics:

```
$ sudo tc qdisc add dev client-eth0 root handle 1: tbf rate 1Mbit burst
    1600 limit 3200
$ sudo tc qdisc add dev client-eth0 parent 1:1 handle 10: netem loss 1%
```

```
    25% delay 700ms 50ms 25% distribution normal
```

The following example command was used to verify the configured `tc` settings:

```
$ sudo tc qdisc show dev client-eth0
qdisc tbf 1: root refcnt 2 rate 1 Mbit burst 1600b lat 12.8ms
qdisc netem 10: parent 1:1 limit 1000 delay 700.0ms 50.0ms 25% loss 1%
    25%
```

The following example command was used to delete the configured `tc` settings:

```
$ sudo tc qdisc del dev client-eth0 root
```

The network throughput tool, `iperf`, was included in Mininet and used to validate TCP bandwidth performance on the client-side, ensuring that the desired effects were achieved. Ping was used to validate that network delay settings were properly implemented.

```
mininet> iperf client nat0
*** Iperf: testing TCP bandwidth between client and nat0
*** Results: ['951 Kbits/sec', '1.06 Mbits/sec']
```

5.2.2 Functional Timing Tests

The application execution time of ORION's full expedited mode (device registration and credentialing) was evaluated by measuring the total time required to register and fully credential the device under varying network conditions. The constraints of the network link also served to assess and demonstrate the application's stability and resiliency in low bandwidth and high latency environments. The total time required to complete a full expedited ORION device credentialing process was composed of the sum of the registration and credentialing process. Time cost required to conduct biometric authentication was not added to the total time as the process was simulated; further research examining methods of biometric authentication and its effect on application performance were left for future work.

Bandwidth	Latency (ms)	Packet Loss	Registration (sec)	Credentialing (sec)	Total (sec)
2 Mb	700±50	1	26.45	35.37	61.82
1 Mb	700±50	1	26.46	35.54	62.01
512 kb	700±50	1	26.94	35.51	62.45
256 kb	700±50	1	26.12	35.02	61.15
128 kb	700±50	1	27.01	36.03	63.04
128 kb	700±50	10	27.92	38.17	66.09
128 kb	700±50	20	32.91	47.09	80.00
128 kb	700±50	30	57.52	68.45	125.97
128 kb	700±50	40	92.75	127.09	219.84

Table 5.1. Mean Execution Times for Device Registration and Credentialing

Low Bandwidth Timing Testing

To observe performance in low bandwidth links, the application was tested at different bandwidth levels {128 kbps, 256 kbps, 512 kbps, 1 Mbps, 2Mbps} at and below the baseline using a fixed packet loss of 1%, and a latency of 700±50ms. For each bandwidth setting, the experiment was run a total of 20 times to provide statistical significance and the total time was tabulated, separated by registration, credentialing, and the sum of both. The result of individual experiments are depicted on scatter plots (Figure 5.3, 5.4, and 5.5) for comparison. The results are also depicted on standard boxplots to illustrate the min, max, median, and interquartile timing range (Figure 5.6 and 5.7). The results of the low bandwidth testing showed that decreasing the bandwidth more than 93% (128 kb) below the baseline had relatively minimal effect on the total time required to register and credential a device. Although the total time slightly increased as bandwidth was lowered (+1.22 sec difference between 2Mb and 128 kb), the experiment showed that the application was able to perform flawlessly in low bandwidth environments with no unexpected failures. Overall, the average time required to fully register and credential a device under the ORION framework at and below baseline bandwidth conditions is slightly over one minute in most situations (61.82 sec at 2 Mb and 63.04 at 128kb).

High Loss Timing Testing

To observe the impact of increasing packet loss on ORION's performance, the application was tested at different loss levels {10%, 20%, 30%, 40%} below the baseline using a fixed bandwidth of 128 kb, and a latency of 700±50ms. The loss level represented the number of data packets lost per 100 packets sent by the client. For each packet loss setting,

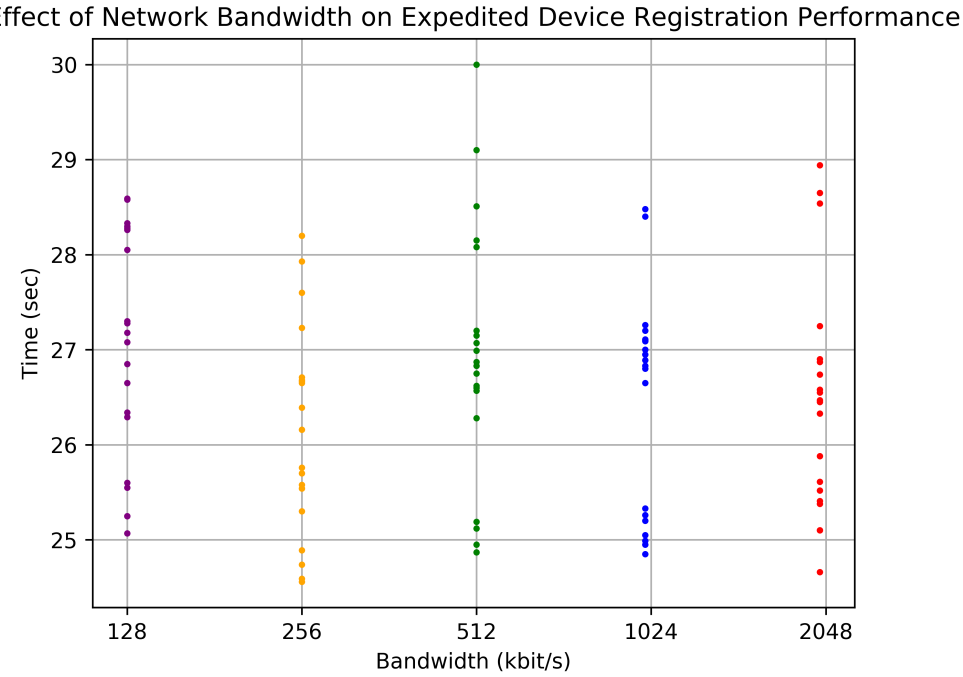


Figure 5.3. Bandwidth Effects on Device Registration

the experiment was run a total of 20 times. Compared to low bandwidth settings, packet loss had the most significant effect on timing performance as it directly impacted TCP performance in two ways. First, as the number of lost packets increased, the number of TCP retransmissions also increased. Second, loss of acknowledgement packets caused the TCP congestion control window to be halved, thus directly reducing the amount of throughput available for use. As packet loss increased, the range between the minimum and maximum execution time increased considerably as shown in Figure 5.9. Additionally, an exponential increase in total application execution time was observed as packet loss increased, see Figure 5.8. No application failures were observed. The experiment demonstrated that the ORION framework and application was capable of reliably performing in low bandwidth, high loss environments, albeit at an expectedly reduced performance level.

5.3 Summary

The ORION Registration App was evaluated for correctness in function and performance under emulated link conditions characteristic of tactical edge networks. The reliability and robustness of the application was tested by further constraining link properties,

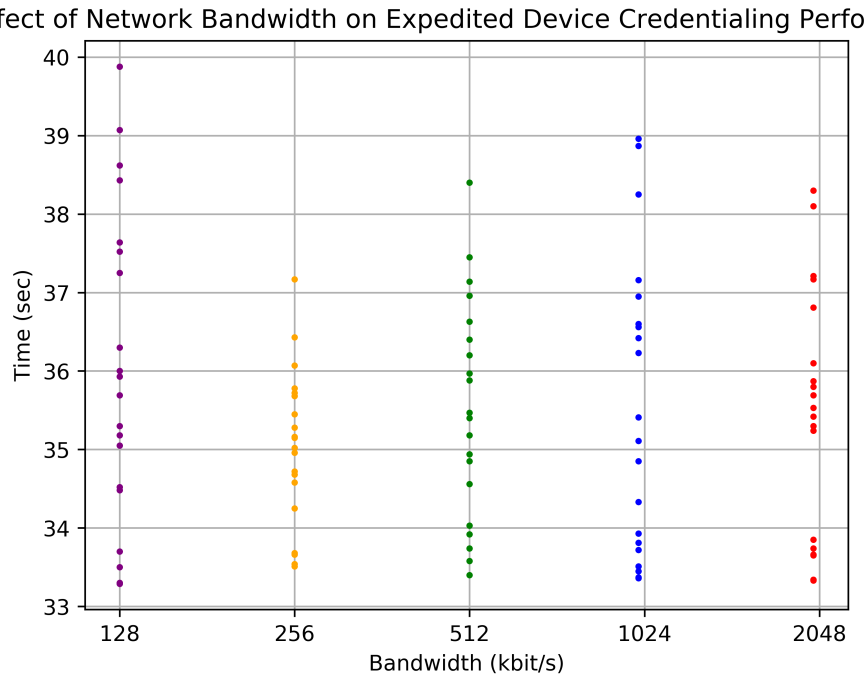


Figure 5.4. Bandwidth Effects on Device Credentialing

specifically bandwidth, packet loss, and latency, to measure the execution time required to register and credential a device using the app's on-demand expedited mode. The results showed that the developed app was able to provide expected functionalities even under the most extreme conditions with a trade-off of reduced performance. No unexpected failures or bugs were observed during the experimentation process. The results of the timing tests demonstrated the app's capability to support mobile PKI credentialing in network degraded environments. Through the implementation of the ORION Registration App, we also demonstrated that a SDN controller can be deployed to perform CA functionalities.

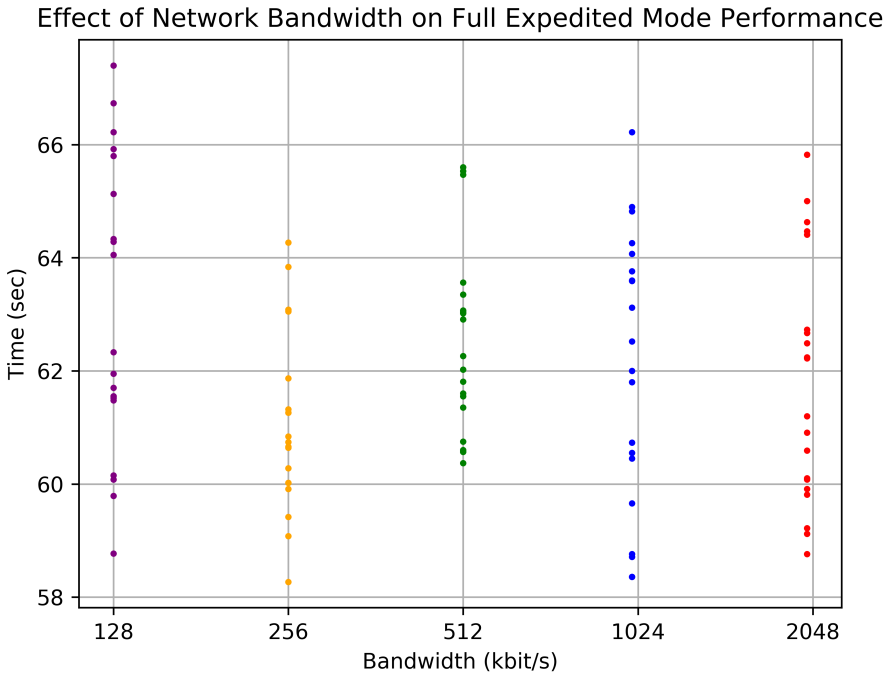


Figure 5.5. Bandwidth Effects on Full Expedited Mode

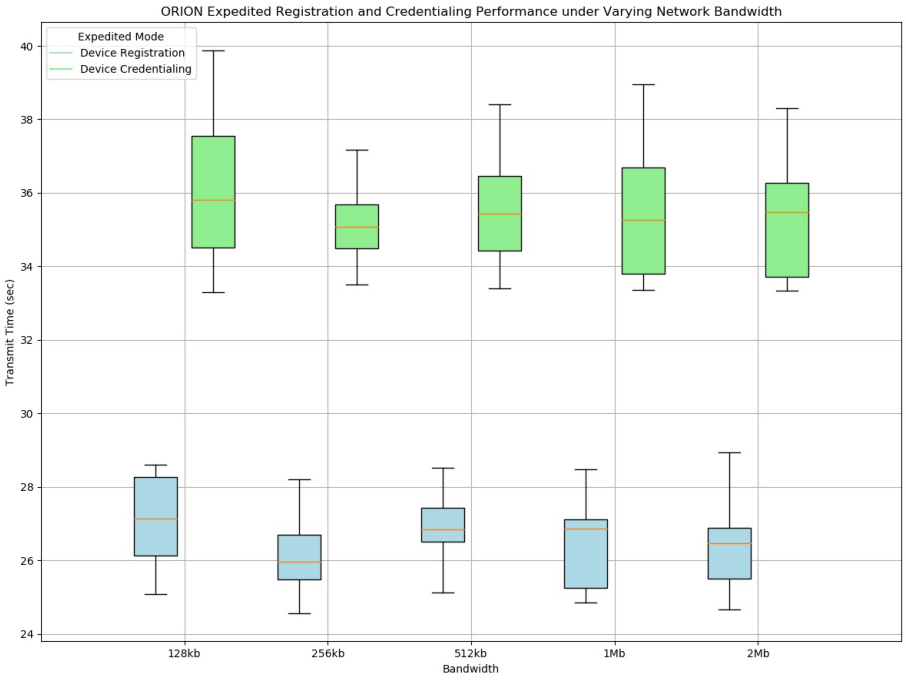


Figure 5.6. Device Registration and Credentialing Under Varying Bandwidth

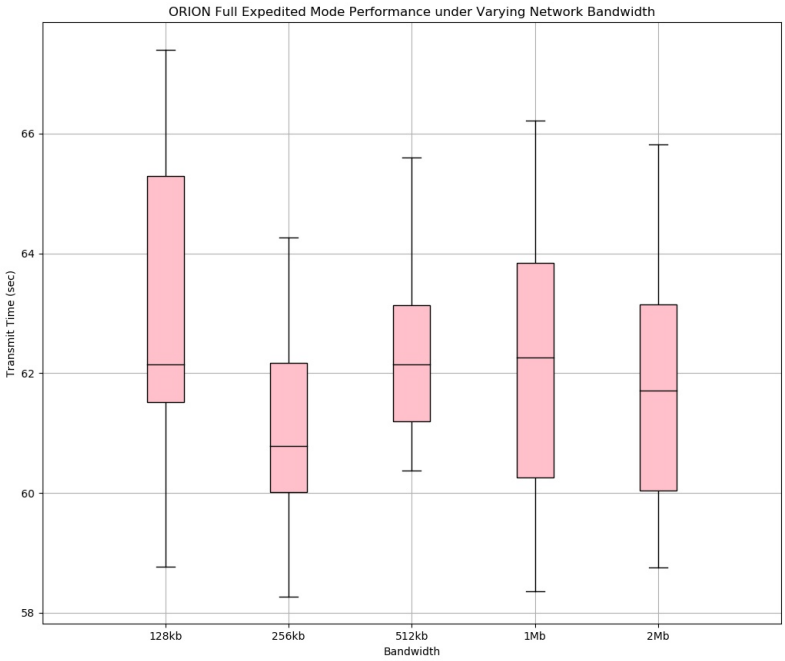


Figure 5.7. Full Expedited Mode Under Varying Bandwidth

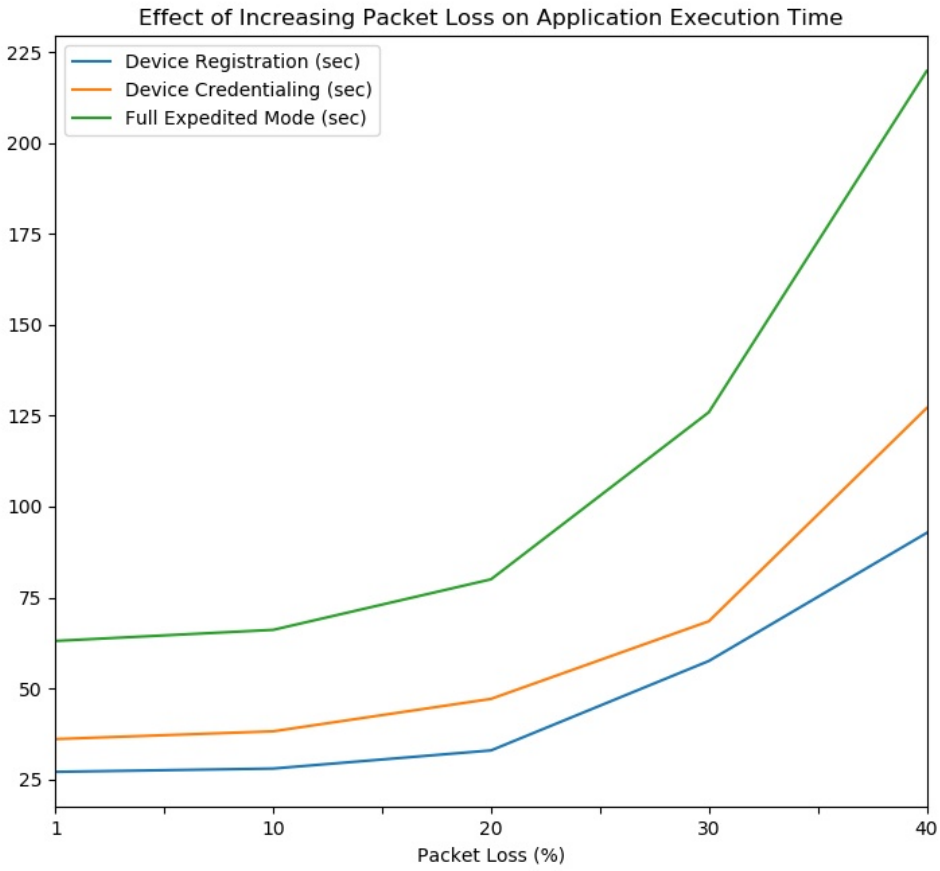
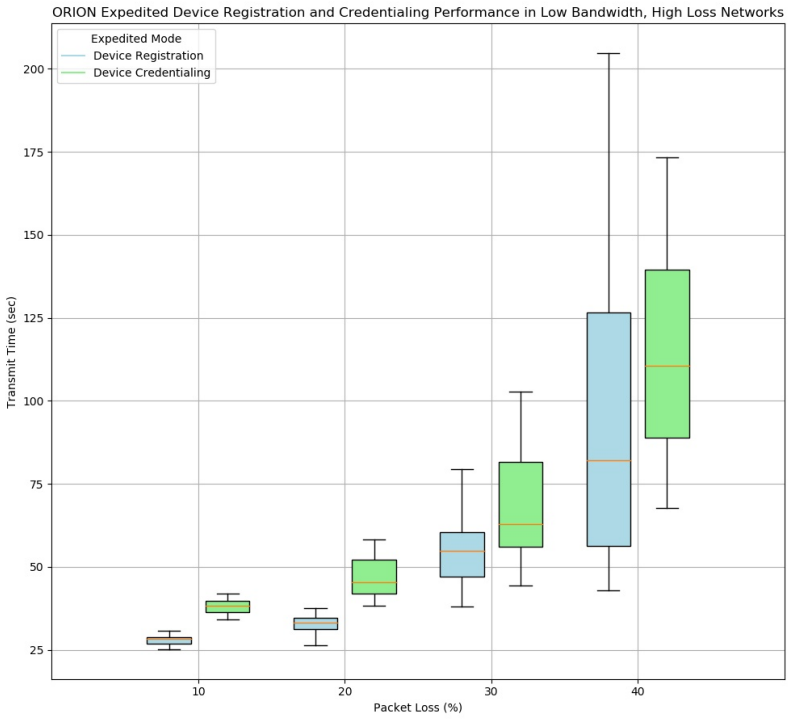
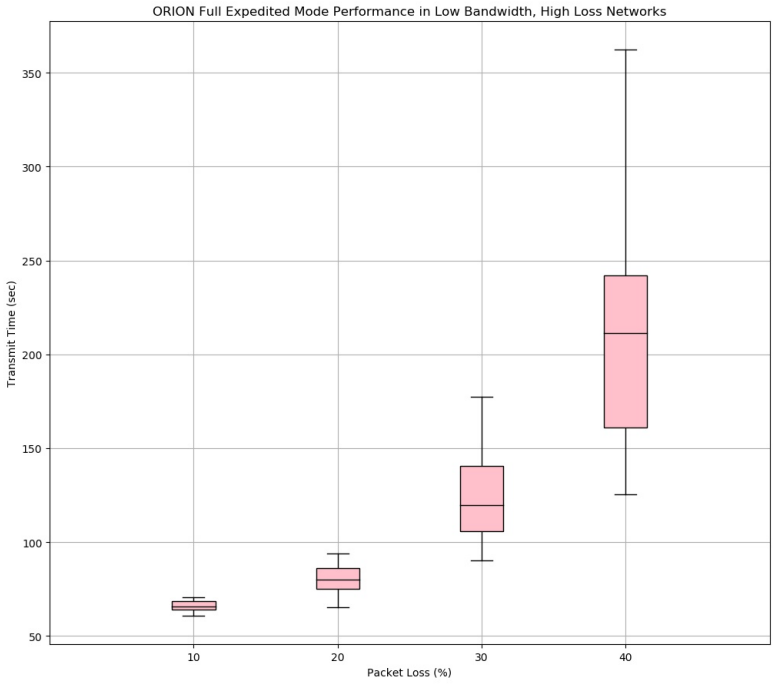


Figure 5.8. Application Timing Performance with Increasing Packet Loss



(a) Device Registration and Credentialing



(b) Full Expedited Mode

Figure 5.9. Performance in Low Bandwidth, High Loss Networks

CHAPTER 6:
Conclusion

I believe in my soul that Marines are different. Our identity is firmly rooted in our warrior ethos. This is the force that will always adapt and overcome no matter what the circumstances are. We fight and win in any clime and place.

—General David H. Berger, 38th Commandant of the Marine Corps [4]

6.1 Conclusion

Since the inception of this research, we have sought to address critical communication-related challenges facing forward deployed Marines in the future operating environment. The management complexity, hardware limitations, and lack of scalability in traditional networking infrastructure created an opportunity gap that could be filled by SDN. SDN solves many issues confronting traditional networks by separating the control plane from the data plane, allowing data plane network devices to serve as simple packet forwarders managed by the SDN controller residing within the control plane. The SDN controller serves as a centralized orchestrator, providing network administrators a global network view and the ability to enforce updates and deploy modifications across the entire network from any remote location.

Anticipating the proliferation of mobile, hand-held technology beyond garrison, we envisioned and developed ORION, a next generation PKI authentication framework for a network of ad-hoc mobile devices. Resembling a localized extension of Purebred (DISA's mobile security solution), ORION was designed specifically for tactical edge networks as it combines SDN's centralized management capability and globally, dynamic programmability with the decentralized, self-healing properties of MANET technology into one scalable, autonomous, interoperable system. Compared to traditional networking, SDN is unencumbered by vendor lock-in issues affecting hardware-based infrastructures. As a result, we believe this holds significant organizational cost saving potential over the system's life cycle. As the warfighting demand for PKE-devices continues to grow, SDN technology is well-positioned to address future challenges in distributed ad hoc mobility models.

In a D2E environment, access to back-end services is not guaranteed and a complete local replication of PKI services would be resource-prohibitive. Likewise, the covert nature of certain tactical operations may necessitate minimal electronic footprints that would be not possible with high-powered transmission systems. As such, ORION was designed to support PKI-dependent functionalities in the absence of remote PKI services. In designing ORION, we have sought to answer the following research questions:

1. In the event that a mobile device is lost, destroyed, or compromised, how can SDN facilitate timely revocation of the device's derived credentials?
2. How can SDN be exploited to securely automate the registration and credentialing process of new mobile devices?

With the understanding that a full device registration and credentialing process may not always be situationally possible, we designed an alternative process which reduced manual procedures and eliminated human oversight, CAC demonstration, and TOTP authentication. In lieu of a TA, the authentication process was automated through the adoption of biometrics. While the expedited mode is less secure, the combination of biometrics and device possession met NIST MFA requirements but offered a much faster enrollment process time. Furthermore, hosting ORION's CAs within the SDN controller offered the flexibility to locally manage the entire PKI process without the need for back-end CA services. Given the requirement to facilitate timely revocation of derived credentials, we proposed possible design considerations for the ORION Revocation App, the primary mechanism for managing on-demand ORION credential revocation.

ORION's framework and the ORION Registration App were developed using open source software and emulated within a virtual Linux environment. Mininet was used to prototype the network topology which included emulated hosts, switches, and links. The SDN's OS, ONOS, hosted the SDN controller and server-side services; ONOS programs were built using the Java programming language. The client-side ORION Registration App was coded in Python for rapid prototyping and testing. The commercial-grade cryptography software, OpenSSL, served as the cryptographic foundation for the PKI build and facilitated execution of tasks related to cryptographic operations.

The ORION Registration App was evaluated for correctness, than deployed onto a test network emulating link conditions characteristic of tactical edge networks to further

examine its performance in constrained and degraded networks. The stress testing demonstrated ORION's robustness and reliability in low bandwidth, high delay, high loss network environments. In the most adverse link environments simulated, e.g., with a low data rate of 128 kbps, network latency of 700ms, and a packet loss rate up to 40%, ORION consistently demonstrated its capability to support the full spectrum of mobile PKI credentialing. Additionally, we showed evidence through implementation that a SDN controller was capable of hosting and performing CA functionalities. As SDN technology continues to advance and gain traction within networking research communities and corporate institutions, we believe that its adaptability, scalability, and employment of software automation is unprecedented, and certainly worthy of consideration, as the Marine Corps continues to modernize the capabilities of tactical communications.

6.2 Future Work

Due to the technical limitations and the scope of this thesis, there exist potential research paths that require further exploration and study to fully realize the potential of ORION:

1. *Develop the ORION Revocation App*– While potential designs were proposed, the ORION Revocation App was not fully implemented and evaluated in this research. Future research should investigate the performance of CRL, OCSP, and OCSP Stapling and determine which method is best for obtaining certificate revocation statuses given the resource and network constraints of a tactical edge network.
2. *Examine distributed controller environments*– This research focused on the implementation of a strict hierarchical CA model using a single controller. Future research should experiment with multi-controller environments e.g., implementation of a federated hierarchical CA model, to examine the interaction between neighboring controllers. The end state of this research should investigate how controllers can work together to establish globally optimal decisions outside of their local area e.g., what happens when a client transitions from one controller network to a neighboring network.
3. *Implement biometric authentication*– Further research is required to design and implement a biometric verification system for ORION. The research should explore methods for biometric authentication from a security and performance perspective

while testing various biometric modalities.

4. *Optimize application performance*– The ORION Registration App's client/server source code requires further analysis to increase code efficiency. Future work should investigate performance bottlenecks in the registration and credentialing process in an effort to improve application execution times. Additionally, entry points and functions should be thoroughly examined to model potential threats and/or identify unexpected software bugs.

APPENDIX A:
Source Code

The ORION source code may be downloaded from the following GitHub repository:

https://github.com/usmc-orion/orion

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B:
Additional References

B.1 DoDIN Access Control Management

Policies, Directives, and Instructions
HSPD-12: Policy for a Common ID Standard for Federal Employees and Contractors
FIPS 201-2: Personal Identity Verification (PIV) of Federal Employees and Contractors
CNSSP-3: National Policy for Granting Access to Classified Cryptographic Information
CNSSP-16: National Policy for the Destruction of COMSEC Paper Material
CNSSD-506: National Directive to Implement PKI on Secret Networks
CNSSI-1300: Instructions for NSS PKI X.509
NSTISSI-3028: Operational Security Doctrine for the FORTEZZA User PCMCIA Card
CNSSI-4001: Controlled Cryptographic Items
CNSSI-4003: Reporting and Evaluating COMSEC Incidents
CNSSI-4005: Safeguarding COMSEC Facilities and Materials, amended by CNSS-008-14
CNSSI-4006: Controlling Authorities for COMSEC Material
DoDI 1000.25: DoD Personnel Identity Protection (PIP) Program
DoDI 5200.01: DoD Information Security Program and Protection of SCI
DoDI 5200.08: Security of DoD Installations and Resources and the DoD PSRB
DoDI 8520.02: Public Key Infrastructure (PKI) and Public Key Enabling (PKE)
DoDI 8520.03: Identity Authentication for Information Systems
DoDM 1000.13, Vol. 1: DoD ID Cards: ID Card Life-cycle

Table B.1. Updated Nov. 27, 2019. Source: [98].

THIS PAGE INTENTIONALLY LEFT BLANK

List of References

[1] D. Berger, "Together we must design the future force," *U.S. Naval Institute Proceedings*, Vol. 145/11/1401, nov. 2019. [Online]. Available: https://www.usni.org/magazines/proceedings/2019/november/together-we-must-design-future-force.

[2] Department of Defense, "Summary of the 2018 National Defense Strategy," Washington, DC, USA, 2018. [Online]. Available: https://dod.defense.gov/Portals/1/Documents/pubs/2018-National-Defense-Strategy-Summary.pdf.

[3] United States Navy, "Department of the Navy Information Superiority Vision," Washington, DC, USA, 2020. [Online]. Available: https://www.navy.mil/strategic/DON_Information_Superiority.pdf.

[4] United States Marine Corps, "38th Commandant's Planning Guidance (CPG)," Washington, DC, USA, 2019. [Online]. Available: https://www.marines.mil/News/Publications/MCPEL/Electronic-Library-Display/Article/1907265/38th-commandants-planning-guidance.

[5] C. Zouridaki, B. Mark, K. Gaj, and R. Thomas, "How many smartphones are in the world?" [Online]. Available: https://www.bankmycell.com/blog/how-many-phones-are-in-the-world. Accessed: 25-Feb-2020.

[6] A. Stone, "Are MIL-STD-810G smartphones truly combat ready?" Samsung Insights, august 2017. [Online]. Available: https://insights.samsung.com/2017/08/08/are-mil-std-810g-smartphones-truly-combat-ready-2/.

[7] K. McCaney, "The Army is automating digital certificate monitoring in the field," Defense Systems, august 10, 2016. [Online]. Available: https://defensesystems.com/articles/2016/08/10/army-cerdec-pkite-certificate-monitoring.aspx.

[8] C. D. D. DoD PKI PMO, "DoD PKI Purebred system and protocol review," DISA, 2018. [Online]. Available: https://dl.cyber.mil/pki-pke/pdf/unclass-purebred_security_whitepaper.pdf.

[9] L. Bacon, "Commandant looks to 'disruptive thinkers' to fix Corp's problems," Marine Corps Times, March 4, 2016. [Online]. Available: https://www.marinecorpstimes.com/news/your-marine-corps/2016/03/04/commandant-looks-to-disruptive-thinkers-to-fix-corps-problems/.

[10] SelfKey, "All data breaches in 2019 - an alarming timeline," November 2019. [Online]. Available: https://selfkey.org/data-breaches-in-2019/.

[11] J. Martin, "What is access control? A key component of data security," IDG Communications, Inc., 2019. [Online]. Available: https://www.csoonline.com/article/3251714/what-is-access-control-a-key-component-of-data-security.html.

[12] R. Sandhu and P. Samarati, "Access control: principles and practice," IEEE Communications Magazine, September 1994. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=312842.

[13] "DoD Information Assurance Certification and Accreditation Process (DIACAP) Handbook," Department of the Navy, Washington, DC, USA, July 2008. [Online]. Available: https://www.acqnotes.com/Attachments/NAVY%20DoD%20Information%20Assurance%20Certification%20and%20Accreditation%20Process%20Handbook.pdf.

[14] "Risk Management Framework (RMF) for DoD Information Technology (IT)," DoD Instruction 8510.01, Department of Defense, Washington, DC, USA, March 12, 2014. [Online]. Available: https://www.esd.whs.mil/Portals/54/Documents/DD/issuances/dodi/851001_2014.pdf.

[15] J. Cheng, "DoD switches to NIST security standards," Defense Systems, April 2014. [Online]. Available: https://defensesystems.com/articles/2014/04/03/dod-adopts-nist-security-standards.aspx.

[16] R. Kuhn, V. Hu, T. Polk, and S. Chang, "Sp 800-32 Introduction to Public Key Technology and the Federal PKI Infrastructure," NIST, 2001. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-32.pdf.

[17] T. Nidecki, "Common password vulnerabilities and how to avoid them," October 2019. [Online]. Available: https://www.acunetix.com/blog/web-security-zone/common-password-vulnerabilities/.

[18] "Intro to cryptography and symmetric cryptography," class notes for Introduction to Computer Security, Dept. of Computer Science, Naval Postgraduate School, Monterey, CA, USA, summer 2018.

[19] D. M'Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen, "RFC: 4226, HOTP: An HMAC-Based One-Time Password Algorithm," IETF, 1995. [Online]. Available: https://tools.ietf.org/html/rfc4226.

[20] N. Haller, C. Metz, P. Nesser, and M. Straw, "RFC: 2289, A One-Time Password System," IETF, 1998. [Online]. Available: https://tools.ietf.org/html/rfc2289.

[21] C. Woodford, "Two-factor authentication," October 2018. [Online]. Available: https://www.explainthatstuff.com/how-security-tokens-work.html.

[22] S. N. Corporation, “Simple key loader AN/PYQ-10,” March 2015. [Online]. Available: https://dokumen.tips/documents/skl.html.

[23] “Electronic Key Management System (EKMS) Policy and Procedures for Navy EKMS Tiers 2 and 3,” EKMS-1B, Naval Communications Security Material System, Andrews AFB, MD, USA, 2010. [Online]. Available: https://www.hqmc.marines.mil/Portals/137/EKMS-1B_AMD-9.pdf.

[24] Business Wire, “Sypris electronics awarded multi-year, $125 million IDIQ contract from the Department of Defense,” November 2007. [Online]. Available: https://archive.is/20131201104159/http://web.archive.org/web/20110715060125/http://www.pcb007.com/pages/zone.cgi?a=17519.

[25] H. Krawczyk, M. Bellare, and R. Canetti, “RFC: 2104, HMAC: Keyed-Hashing for Message Authentication,” IETF, 1997. [Online]. Available: https://www.ietf.org/rfc/rfc2104.txt.

[26] W. Simpson, “RFC: 1661, The Point-to-Point Protocol (PPP),” IETF, 1994. [Online]. Available: https://tools.ietf.org/html/rfc1661.

[27] W. Simpson, “RFC: 1334, PPP Authentication Protocols,” IETF, 1992. [Online]. Available: https://tools.ietf.org/html/rfc1334#section-2.

[28] W. Simpson, “RFC: 1994, PPP Challenge Handshake Authentication Protocol (CHAP),” IETF, 1996. [Online]. Available: https://tools.ietf.org/html/rfc1994.

[29] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz, “RFC: 3748, Extensible Authentication Protocol (EAP),” IETF, 2004. [Online]. Available: https://tools.ietf.org/html/rfc3748.

[30] K. Dooley, “An introduction to authentication protocols,” March 2015. [Online]. Available: https://www.auvik.com/franklymsp/blog/authentication-protocols/.

[31] C. Rigney, S. Willens, A. Rubens, and W. Simpson, “RFC: 2865, Remote Authentication Dial In User Service (RADIUS),” IETF, 2000. [Online]. Available: https://tools.ietf.org/html/rfc2865.

[32] V. Fajardo, J. Arkko, J. Loughney, and G. Zorn, “RFC: 6733, Diameter Base Protocol,” IETF, 2012. [Online]. Available: https://tools.ietf.org/html/rfc6733.

[33] C. Finseth, “RFC: 1492, An Access Control Protocol, Sometimes Called TACAS,” IETF, 1993. [Online]. Available: https://tools.ietf.org/html/rfc1492.

[34] Massachusetts Institute of Technology, “Kerberos: the network authentication protocol,” January 2019. [Online]. Available: https://web.mit.edu/kerberos/.

[35] C. Neuman, T. Yu, S. Hartman, and K. Raeburn, "RFC: 4120, The Kerberos Network Authentication Service (V5)," IETF, 2005. [Online]. Available: https://tools.ietf.org/html/rfc4120.

[36] MVPS Ltd., "The kerberos protocol," June 2019. [Online]. Available: https://www.mvps.net/docs/wp-content/uploads/2019/06/kerberos.png.

[37] IBM Knowledge Center, "Digital certificates and authentication," [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/SSB23S_1.1.0.13/gtps7/s7cert.html. Accessed: 09-Dec-2019.

[38] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "RFC: 5280, TOTP: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," IETF, 2008. [Online]. Available: https://tools.ietf.org/html/rfc5280.

[39] M. Cooper, "Best practices for securing private keys," 2008. [Online]. Available: https://revocent.com/best-practices-for-securing-private-keys/.

[40] S. Rani, P. Singh, and R. Preet, "Reviewing Manets & Configurations of Certification Authority (CA) for Node Authentication," International Journal of Computer Science and Information Technologies, Vol. 4 (6), 2013, 974-978, [Online]. Available: http://www.ijcsit.com/docs/Volume%204/Vol4Issue6/ijcsit2013040651.pdf. Accessed: 26-Aug-2019.

[41] A. Money, "Department of Defense (DoD) Public Key Infrastructure (PKI)," official memorandum, DoD Chief Information Officer, Washington, DC, USA, 2000. [Online]. Available: https://www.hsdl.org/?view&did=454235.

[42] J. Hamre, "Management Reform Memorandum #16 - Identifying Requirements for the Design, Development and Implementation of a DoD Public Key Infrastructure," official memorandum, Deputy Secretary of Defense, Washington, DC, USA, august 6, 1997. [Online]. Available: https://archive.defense.gov/dodreform/mrms/mrm16.htmlhttps://apps.dtic.mil/dtic/tr/fulltext/u2/a391663.pdf.

[43] United States Navy, "Public Key Infrastructure Implementation Plan for the Department of the Navy," DTIC, Ft. Belvoir, VA, USA, 2000. [Online]. Available: https://apps.dtic.mil/dtic/tr/fulltext/u2/a391663.pdf.

[44] H. Ferraiolo, D. Cooper, S. Francomacaro, A. Regenscheid, J. Mohler, S. Gupta, and W. Burr, "Special Publication 800-157 Guidelines for Derived PIV Credentials," NIST, 2014. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-157.pdf.

[45] J. Mauck, "DoN Current and Future PKI and PKE Activities," Chief Information Officer, Department of the Navy, Washington, DC, USA, May 18, 2010. [Online]. Available: https://www.doncio.navy.mil/ContentView.aspx?ID=1748.

[46] D. Deasy, "Modernizing the common access card - streamlining identity and improving operational interoperability," official memorandum, Office of the Secretary of Defense, Washington, DC, USA, February 1, 2019. [Online]. Available: https://dodcio.defense.gov/Portals/0/Documents/Cyber/modernizing_the_cac.pdf. [Accessed: 13-Jan-2020].

[47] P. Dasgupta, K. Chatha, and S. Gupta, "Viral attacks on the DoD common access card CAC," Department of Computer Sc. and Eng., Arizona State University, Tempe, AZ, USA, 2009. [Online]. Available: https://pdfs.semanticscholar.org/1a0a/895aa4abc182645f6e770b4a206ff65e6472.pdf.

[48] Defense Human Resource Activity (DHRA), "Common Access Card (CAC) Security," 2019. [Online]. Available: https://www.cac.mil/Common-Access-Card/CAC-Security/.

[49] A. Money, "Common Access Card (CAC)," official memorandum, DoD Chief Information Officer, Washington, DC, USA, January 16, 2001. [Online]. Available: https://www.dmdc.osd.mil/smartcard/images/01.16.01CACPolicyMemo-Final.pdf.

[50] V. Athanasopoulos, "Design and development of a web-based DOD PKI common access card (CAC) instruction tool," M.S. thesis, Dept. of Comp. Sci., NPS, Monterey, CA, USA, 2004. [Online]. Available: http://hdl.handle.net/10945/1714.

[51] The Linux Documentation Project, "Classication of Smart Cards," [Online]. Available: https://www.tldp.org/HOWTO/Smart-Card-HOWTO/classification.html. Accessed: 05-Feb-2020.

[52] S. Zanero, "Smart card content security," january 1997. [Online]. Available: https://home.deib.polimi.it/zanero/papers/scsecurity.pdf.

[53] "Department of Defense Commerical Mobile Device Implementation Plan," Chief Information Officer, Department of Defense, Washington, DC, USA, february 15, 2013. [Online]. Available: https://archive.defense.gov/news/DoDCMDImplementationPlan.pdf.

[54] T. Larkins, "BYOD in Defense Department? Not in this lifetime," 2014. [Online]. Available: https://www.informationweek.com/government/mobile-and-wireless/byod-in-defense-department-not-in-this-lifetime/d/d-id/1113418.

[55] Defense Information Systems Agency (DISA), "Mobile derived credentials purebred information brief," 2018. [Online]. Available: https://cloud.afcea.org/owncloud/s/SzGv4yUanag9y1X?path=%2FPresentations_Slides%2FCyber%20Security%20Theater#pdfviewer.

[56] P. Grassi, M. Garcia, and J. Fenton, "Special Publication 800-63-3 Digital Identity Guidelines," NIST, 2017. [Online]. Available: https://pages.nist.gov/800-63-3/sp800-63-3.html.

[57] C. Wallace, email, Feb. 2016.

[58] M. Pritikin, A. Nourse, and J. Vilhuber, "Simple Certificate Enrollment Protocol," IETF, 2011. [Online]. https://tools.ietf.org/html/draft-nourse-scep-23.

[59] K. Moriarty, M. Nystrom, S. Parkinson, A. Rusch, and M. Scott, "RFC: 7292, PKCS #12: Personal Information Exchange Syntax v1.1," IETF, 2014. [Online]. Available: https://tools.ietf.org/html/rfc7292.

[60] S. Willden, "Keystore Key Attestation," Android, september 2017. [Online]. Available: https://android-developers.googleblog.com/2017/09/keystore-key-attestation.html.

[61] J. Kurose and K. Ross, "*Computer Networking: A Top-Down Approach*, 7th Edition," Boston, MA, USA: Addison-Wesley, 2013.

[62] C. Cordeiro and D. Agrawal, "Mobile Ad hoc Networking," [Online]. Available: https://eecs.ceas.uc.edu/~cordeicm/course/survey_ad_hoc.pdf. Accessed: 23-Jan-2020.

[63] "INMARSAT BGAN RF-7800B-DUO24," Orbitica, [Online]. Available: https://www.francesatellite.com/inmarsat/rf7800bduo24.html. Accessed: 24-Jan-2020.

[64] M. Turner and K. Dingman, "Developing SCA Based Wideband Networking Waveforms," L3harris, 2011. [Online]. Available: https://www.wirelessinnovation.org/assets/Proceedings/2011/2011-7d-dingman-presentation.pdf.

[65] United States Army, "Army networking radios improve communications at tactical edge," ASA (ALT) Public Affairs, Washington, DC, USA, november 3, 2011. [Online]. Available: https://www.army.mil/article/68498/army_networking_radios_improve_communications_at_tactical_edge.

[66] L3Harris, "L3Harris Falcon iii AN/PRC-117G(V)1(C) Multiband Networking Manpack Radio," [Online]. Available: https://www.harris.com/sites/default/files/downloads/solutions/harris-falcon-iii-an-prc-152a-wideband-networking-handheld-radio.pdf. Accessed: 09-Feb-2020.

[67] L3Harris, "L3Harris Falcon III AN/PRC-152A," [Online]. Available: https://www.harris.com/solution/harris-falcon-iii-an-prc-117gv1c-multiband-networking-manpack-radio. Accessed: 09-Feb-2020.

[68] "FY14 Army programs AN/PRC-117G," DOT&E, 2014. [Online]. Available: https://www.dote.osd.mil/Portals/97/pub/reports/FY2014/army/2014anprc-117g.pdf?ver=2019-08-22-110519-110.

[69] Unadulterated Nerdery, "Designing ANW2 Networks," may 2014. [Online]. Available: https://www.unadulteratednerdery.com/2014/05/16/designing-anw2-networks/.

[70] M. Akin, "Secure infrastructure-less network (SINET)," M.S. thesis, Dept. of Comp. Sci., NPS, Monterey, CA, USA, 2017. [Online]. Available: http://hdl.handle.net/10945/55583.

[71] L3Harris, "AN/PRC-117G multiband manpack radio reference guide."

[72] "National Security Agency (NSA)," Crypto Museum, 2019. [Online]. Available: https://www.cryptomuseum.com/intel/nsa/index.htm.

[73] D. Kreutz, F. Ramos, P. Verissimo, C. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: a comprehensive survey," IEEE, 2014. [Online]. Available: https://arxiv.org/pdf/1406.0440.pdf.

[74] T. Keary, "What is software defined networking (SDN) and why is it important?" 2018. [Online]. Available: https://www.comparitech.com/net-admin/software-defined-networking/.

[75] V. Mishra, A. Dusia, and A. Sethi, "Routing in software-defined mobile ad hoc networks (SD-MANET)," US Army Research Laboratory, Adelphi, MD, USA, august 2018. [Online]. Available: https://www.arl.army.mil/arlreports/2018/ARL-TR-8469.pdf.

[76] K. Poularakis, G. Iosifidis, and L. Tassiulas, "SDN-enabled tactial ad hoc networks: extending programmable control to the edge," *IEEE Communications Magazine*, Volume 56, Issue 7, pp. 132-138, july 2018. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8419193.

[77] J. Nobre, D. Rosario, C. Both, E. Cerqueira, and M. Gerla, "Toward software-defined battlefield networking," *IEEE Communications Magazine*, Volume 54, Issue 10, pp. 152-157, october 2016. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7588285.

[78] K. Poularakis, Q. Qin, E. Nahum, M. Rio, and L. Tassiulas, "Bringing SDN to the mobile edge," SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI, august 2017. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8397407.

[79] C. Zouridaki, B. Mark, K. Gaj, and R. Thomas, "Distributed CA-based PKI for mobile ad hoc networks using elliptic curve cryptography," *First European PKI Workshop: Research and Applications*, Volume 3093, pp. 232-245, june 2004. [Online]. Available: https://ece.gmu.edu/~kgaj/publications/conferences/GMU_EuroPKI_2004.pdf.

[80] "Purebred Agent Guide," Department of Defense, Washington, DC, USA, december 2019. [Online]. Available: https://dl.cyber.mil/pki-pke/pdf/unclass_purebred_agent_guide.pdf.

[81] "Interim Digital Authentication Guidelines for Unclassified and Secret Classified DoD Networks and Information Systems," Department of Defense, Washington, DC, USA, august 2019. [Online]. Available: https://dl.cyber.mil/idam/pdf/unclass-dod_interim_digital_authentication_guidelines.pdf.

[82] P. Grassi, J. Fenton, R. Perlner, W. Burr, and J. Richer, "Special Publication 800-63b Digital Identity Guidelines- Authentication and Lifecycle Management," NIST, 2020. [Online]. Available: https://pages.nist.gov/800-63-3/sp800-63b.html.

[83] M. Wasserman, S. Hartman, and D. Zhang, "Security Analysis of the Open Networking Foundation (ONF) OpenFlow Switch Specification," IETF, 2012. [Online]. Available: https://tools.ietf.org/id/draft-mrw-sdnsec-openflow-analysis-00.html.

[84] "Department of Defense Public Key Infrastructure Registration Authority/Local Registration Authority Certification Practice Statement," Department of Defense, Washington, DC, USA, 2019. [Online]. Available: https://dl.cyber.mil/pki-pke/pdf/unclass-fouo-DoD-RA-LRA-CPS-v5-20190410.pdf.

[85] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," IETF, 2012. [Online]. Available: https://tools.ietf.org/html/rfc5280#section-5.

[86] Maikel, "The Current State of Certificate Revocation (CRLs, OCSP and OCSP Stapling," [Online]. Available: https://www.maikel.pro/blog/current-state-certificate-revocation-crls-ocsp/. Accessed: 04-June-2020.

[87] USMC, "Marine Corps Concept for Cyberspace Operations," U.S. Marine Corps Concepts & Programs, October 2015, [Online]. https://www.candp.marines.mil/

Portals/216/documents/Concepts/MCFC%206-1%20Cyberspace%20Operations_1.pdf?ver=2018-05-01-133728-813. [Accessed: 31-Mar-2020].

[88] "Mininet," Mininet Team, [Online]. Available: http://mininet.org/. Accessed: 23-Apr-2020.

[89] "Wireshark," Wireshark Foundation, [Online]. Available: https://www.wireshark.org/. Accessed: 23-Apr-2020.

[90] B. Lantz, N. Handigol, B. Heller, and V. Jeyakumar, "Introduction to Mininet," Github, [Online]. Available: https://github.com/mininet/mininet/wiki/Introduction-to-Mininet. Accessed: 23-Apr-2020.

[91] "OpenFlow Switch Specification Version 1.3.5," Open Networking Foundation, [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.3.5.pdf. Accessed: 25-Apr-2020.

[92] T. Vachuska, "Onos," Open Networking Foundation, [Online]. Available: https://wiki.onosproject.org/display/ONOS/ONOS. Accessed: 26-Apr-2020.

[93] "OpenSSL," OpenSSL Software Foundation, [Online]. Available: https://www.openssl.org/. Accessed: 2-June-2020.

[94] R. Housley, "RFC: 5652, Cryptographic Message Syntax (CMS)," IETF, 2009. [Online]. Available: https://tools.ietf.org/html/rfc5652#section-1.

[95] United States Marine Corps, "Marine Corps Operating Concept," Washington, DC, USA, 2016. [Online]. Available: https://www.candp.marines.mil/Portals/216/documents/Concepts/Marine%20Corps%20Operating%20Concept%20Sept%202016.pdf?ver=2018-05-01-133729-063.

[96] "Traffic Control Linux Manual Page," Linux, [Online]. Available: http://man7.org/linux/man-pages/man8/tc.8.html. Accessed: 11-June-2020.

[97] G. Hablinger and O. Hohlfeld, "The Gilbert-Elliot Model for packet loss in real time services on the internet," [Online]. Available: https://www.net.t-labs.tu-berlin.de/papers/HH-GEMPLRTSI-08.pdf. Accessed: 12-June-2020.

[98] Defense Technical Information Center, "The DoD cybersecurity policy chart," November 2019. [Online]. Available: https://www.csiac.org/resources/the-dod-cybersecurity-policy-chart/.

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California