SAND2021-11414

SANDIA REPORT
SAND20XX-XXXX
Printed June 2021

Sandia
National
Laboratories

Named Data Networking for DER Cybersecurity

Chavez, Adrian
Cordeiro, Patricia G
Huang, Gary
Kitsos, Panayioti Constantine
La Pay, Trevor
Onunkwo, Ifeoma
Short, Austin
Summers, Adam

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico
87185 and Livermore,
California 94550

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology & Engineering Solutions of Sandia, LLC.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods/

Special Thanks to Operant Networks

Operant Networks, Inc. is the industry leader in Named Data Networking. Founded in 2016 by veteran renewable energy executives, Operant in collaboration with the Department of Energy, has been spearheading the effort to commercialize Named Data Networking for the next generation electric grid. Building on the work done by their academic partners, Operant has made public their ndn.ind codebase with the purpose of gaining industry validation and widespread adoption.

OPERANT
NETWORKS

ABSTRACT

We present our research findings on the novel NDN protocol. In this work, we defined key attack scenarios for possible exploitation and detail software security testing procedures to evaluate the security of the NDN software. This work was done in the context of distributed energy resources (DER). The software security testing included an execution of unit tests and static code analyses to better understand the software rigor and the security that has been implemented. The results from the penetration testing are presented. Recommendations are discussed to provide additional defense for secure end-to-end NDN communications.

ACKNOWLEDGEMENTS

This work is supported by the Department of Energy Office of Solar Energy Technologies Office (SETO), Project # DE-FOA-0001976 CFDA Number: 81.049. This project is led by Operant Networks, Incorporated and partnered with Sandia National Laboratories.

CONTENTS

1. Propaedeutic 10
   1.1. Motivation 10
   1.2. Architecture 10
      1.2.1. Nodes 11
      1.2.2. Packets 12
   1.3. Protocol 12
      1.3.1. Design Principles 12
      1.3.2. Description 13
   1.4. Strengths and Weaknesses 13
   1.5. Testbed 13
   1.6. Security 14
2. Operant Innovation 16
   2.1. Application to DER 16
   2.2. Using NDN Simulator 16
3. Preliminary Cybersecurity Observations 17
4. Security Evaluation 18
   4.1. Test Objectives 18
   4.2. Scope of Testing 18
   4.3. Testing Strategy 18
   4.4. Toolsets 18
5. Attack Scenarios 19
   5.1. Distributed Denial of Service Attack 20
   5.2. Content Poisoning Attack 20
   5.3. Cache Pollution 21
   5.4. Name Hijacking 21
   5.5. Route Hijacking 22
   5.6. Application Hijacking 22
   5.7. Man-in-the-middle Attack 22
   5.8. Replay Attack 23
   5.9. Anonymity and Censorship Attack 23
   5.10. Privacy Attack of Interest Packets 24
   5.11. Reflection Attack 25
   5.12. Caveats 25
6. White Box Security Testing and unit testing 26
   6.1. SAST Test Cases 26
      6.1.1. Static Application Security Testing (SAST) Phase I 26
      6.1.2. SAST Findings 27
      6.1.3. SAST Finding Discussion 28
      6.1.4. Static Application Security Testing (SAST) using Code Dx 29
      6.1.5. SAST Findings using Code Dx 30
   6.2. Unit Tests 31
      6.2.1. ndn-ind 31
7. Penetration Testing 33
   7.1 Test Cases 33

7.2 Test Results....34
7.2.1 Reconnaissance....34
7.2.2 Interruption....43
7.2.3 Packet Replay....44
7.2.4 Using Components with Known Vulnerabilities....45
7.2.5 Content Poisoning....45
7.2.6 Injection....45
7.2.7 Sensitive Data Exposure....47
7.2.8 Broken Access Control....48
7.2.9 Security Misconfiguration....49
8 Conclusion....54

LIST OF FIGURES

Figure 1-1. Internet and Hourglass Architecture....11
Figure 1-2. Node Detail....11
Figure 1-3. Packet Types....12
Figure 2-1. Co-Simulation Architecture....16
5-1: Generic Adversary Workflow....19
Figure 7-1: NDN testbed for penetration testing....34
Figure 7-2: Extracted metadata from the ndn-ind and nfd-ind repositories....35
Figure 7-3: TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts....36
Figure 7-4: Full TCP Connect scan results indicating the likelihood of a firewall/IDS....37
Figure 7-5: TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts....37
Figure 7-6: TCP ACK scans show all ports as unfiltered, hinting that the firewall/IDS is likely stateless....37
Figure 7-7: TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts....39
Figure 7-8: TCP ACK scans show all ports as unfiltered, hinting that the firewall/IDS is likely stateless....39
Figure 7-9: Full TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts....40
Figure 7-10: Full TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts....41
Figure 7-11: Unsuccessful attempt at DDoS attack by flooding PIT with random content requests....43
Figure 7-12: Loop repeated replay attack....45
Figure 7-13: Generated data using /localnet/hello123456....46
Figure 7-14: Generated interest packet for /localhost/zeekall....46
Figure 7-15: Data packet returned for the generated interest packet for /localhost/zeekall....47
Figure 7-16: Interest packets with names encoded....48
Figure 7-17: Random encrypted data received in response to interest packets....48
Figure 7-17: Packets protected using name-based access control (NAC)....49
Figure 7-18: tlv-key-locator not found in packets....49

LIST OF TABLES

Table 5-1: Interest Packet Flooding ... 19
Table 5-2: Data Poisoning Attack using a malicious provider ... 20
Table 5-3: Data Poisoning Attack using a compromised router ... 20
Table 5-4: Cache Pollution ... 20
Table 5-5: Naming Hijacking ... 20
Table 5-6: Route Hijacking ... 21
Table 5-7: Application Hijacking ... 21
Table 5-8: MITM Data Exfiltration of Unencrypted Data ... 22
Table 5-9: Replay Attack ... 22
Table 5-10: Anonymity and Censorship Attack ... 22
Table 5-11: Man-in-the-Middle Attack ... 23
Table 5-12: Frequency-Watt points ... 23
Table 5-13: Ransomware Attack ... 24
Table 5-14: Privilege Attack ... 25
Table 5-15: Fixed PF ... 25
Table 5-16: Fixed Reactive Power ... 25
Table 5-17: Privacy Attack of interest packets ... 26
Table 5-18: Privacy Attack of interest tables ... 26
Table 5-19: Cache Timing Attack ... 26
Table 5-20: Object Discovery Attack ... 26
Table 6-1: SAST Test Cases ... 28
Table 6-2: SAST Profiling ... 29
Table 6-3: SAST Finding Detail ... 30
Table 7-1: Penetration Test Cases ... 32

This page left blank

ACRONYMS AND DEFINITIONS

Abbreviation	Definition
ACK	Acknowledge
CAIDA	Center for Applied Internet Data Analysis
CS	Content Store
CWE	Common Weakness Enumeration
DER	Distributed Energy Resource
DDoS	Distributed Denial of Service
DoS	Denial of Service
FIB	Forwarding Information Base
ICN	Information-Centric Networking
IDS	Intrusion Detection System
IoT	Internet of Things
IP	Internet Protocol
MITM	Man-in-the-Middle
NDN	Named Data Network
NFD	NDN Forwarding Daemon
NSF	National Science Foundation
NLSR	Named-data Link State Routing
OSINT	Open Source Intelligence
OT	Operational Technology
PIT	Pending Interest Table
UUID	Universally Unique Identifier

1. PROPAEDEUTIC

Named Data Networking (NDN) will be briefly described to orient the reader to the motivation, the design of the protocol and the security of the concept, contrasting it with the existing Internet communication structure. NDN's relevance and uniqueness in certain communication domains are examined to review the problems it attempts to solve, its inherent strengths, and the weaknesses yet to be addressed.

NDN, designed specifically from the start as a secured data-centric network architecture, is part of NSF's Future Internet Architecture Program.

1.1. Motivation

Internet communication, as advanced as it may seem, is modeled on the original idea of telephony: communicating from one known entity to another known entity. In response to the dramatic shift from point-to-point communication to multicast and requests for particular data, techniques for Information-Centric Networking (ICN), have in the past evolved for more efficient named content distribution. For example, in 2001, the content-addressable web was discussed in [13]. More recently, magnet links [14] are used for downloading files identified by a hash of their contents rather than by their location.

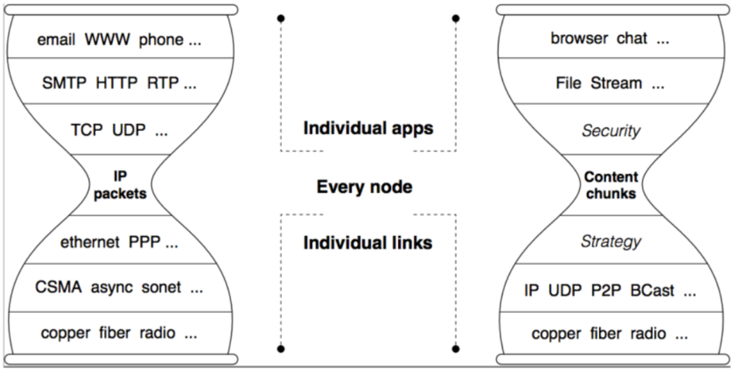
In [15], CISCO describes its motivation for embracing Information-Centric Networking. CISCO states, "All future networks will be inherently mobile ... and anchorless."

Named Data Networking Next Phase (NDN-NP) Project May 2016 - April 2017 Annual Report [16] argues, "an NDN packet can name anything – an endpoint, a data chunk in a movie or a book, a service, or a command to turn on some lights." Importantly, the report views NDN-based applications enabling IoT and multimedia working without reliance on the cloud.

As described in [1], an essential problem with a point-to-point model is that the desired address must be known, and the entity puts all its trust in this address. In contrast, the proposed NDN communication system consists of requests for and deliveries of named data, resulting purportedly in the efficient use of infrastructure, greater security and simpler applications. NDN is proposed as the next evolution of networking, as opposed to a replacement.

1.2. Architecture

NDN attempts to evolve from the notion of a communication network, as the Internet was designed, to a concept of a distribution network [10], and establishes an hourglass architecture differing from that of the existing Internet (Figure 1-1).



Internet and NDN Hourglass Architectures

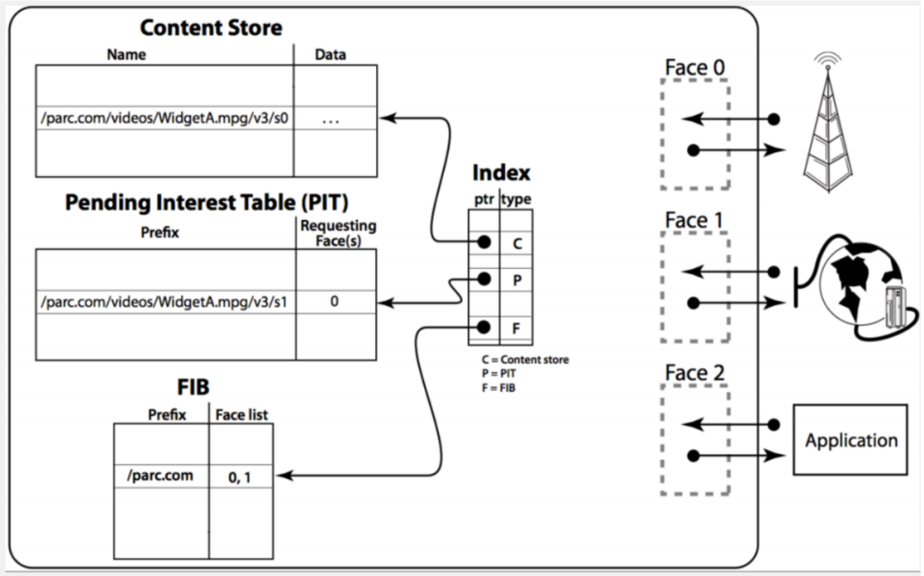
Figure 1-1. Internet and Hourglass Architecture

The components of the NDN architecture for routing data between data producers and data consumers include packets and nodes described below.

1.2.1. Nodes

An NDN node through which data is routed consists of a Content Store (CS) which caches data, a Pending Interest Table (PIT) for caching data requests, and a Forwarding Information Base (FIB) as a type of routing table.

The functionality of nodes of the NDN architecture is detailed in Figure 1-2, from **Error! Reference source not found.**.

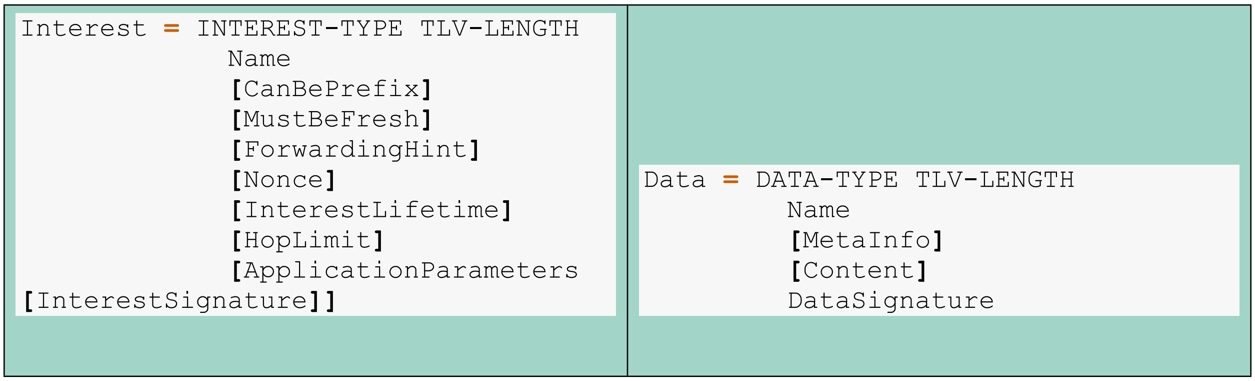


NDN Node

Figure 1-2. Node Detail

1.2.2. Packets

The NDN communication architecture consists of two main types of packets: interest packets published by a requestor and data packets returned from a producer with a signature to verify the authenticity and integrity of the message. The size of the NDN interest packets in some use cases are generally larger than the TCP/IP or UDP/IP packets. Larger packets have less overhead in the header; thus, they are preferable. In general, larger packets will result in higher throughput and less computational overhead for the header information. As shown in Figure 1-3, also from **Error! Reference source not found.**, the Interest and Data Packets carry no host or interface IP addresses.



```
Interest = INTEREST-TYPE TLV-LENGTH
             Name
             [CanBePrefix]
             [MustBeFresh]
             [ForwardingHint]
             [Nonce]
             [InterestLifetime]
             [HopLimit]
             [ApplicationParameters
[InterestSignature]]
```

```
Data = DATA-TYPE TLV-LENGTH
         Name
         [MetaInfo]
         [Content]
         DataSignature
```

Figure 1-3. Packet Types

1.3. Protocol

Data-centricity is the most fundamental divergence of NDN from the Internet protocol. The NDN design principles also feature scalability and security.

1.3.1. Design Principles

The following list from [12] illustrates the design principles of NDN:

1. **Universality**: NDN should be a common network protocol for all applications and network environments.
2. **Data-Centricity and Data Immutability**: NDN should fetch uniquely named, immutable "data packets" requested using "interest packets".
3. **Securing Data Directly**: Security should be the property of data packets, staying the same whether the packets are in motion or at rest.
4. **Hierarchical Naming**: Packets should carry hierarchical names to enable demultiplexing and provide structured context.
5. **In-Network Name Discovery**: Interests should be able to use incomplete names to retrieve data packets.
6. **Hop-by-Hop Flow Balance**: Over each link, one interest packet should bring back no more than one data packet.

1.3.2. Description

The NDN concept, like other ICNs, is a protocol in which data is requested and delivered by name. In NDN the network seeks data specified by Interest Packets, rather than seeking hosts specified by addresses.

Forwarding of the Interest Packet in NDN is controlled by an NDN Forwarding Daemon (NFD) located in every node. Each node participating in NDN interest forwarding determines:

- Is this data stored 'here'?
- Is this a duplicate request?
- Which next hop forward?

The NFD forwarding strategy determines a request's next hop, as well as when to drop a request by reason of timeout or congestion. Data returned from a producer traverses the exact reverse node path of the request forwarded from the consumer.

1.4. Strengths and Weaknesses

The strength of the NDN design, according to [1] is its efficiency in supporting host to host patterns and user-specific data and service, serving not just bulk transmission. Importantly for digital commerce, it is capable of counting advertising clicks and impressions. Where it was still lagging at the time of the cited tutorial, was in efficiently routing names, scaling the data plane, and providing provable security.

A focus on the efficiency benefit centers on the usage of requests for named data, without concern for the location of the data store [1]. The tangential question arises with the aspect of uses such as browsing, when no specific named data is being sought. In the cases where a user's interest is in a data feed, the source is indeed more important than a known, named data. Unclear is whether copies of the data feed distributed in network stores and frequently updated, are a more efficient use of the balance of storage and bandwidth. Also, when researching a new topic, a user will clearly seek new, unknown data. A search engine crawling the web won't likely index what is on distributed routers. Efficiency gains would appear to be uneven for common uses of communication networks. Indeed, active research into NDN is found in specialized realms such as tactical and military networking (see for example, the YouTube 'Named Data Networking' channel [8]).

NSF-funded research in NDN applications continues to address the areas of synchronization, security and live video streaming.

1.5. Testbed

There currently exists a globally shared resource [3] consisting of NDN gateways, routers and nodes, significant in its value for providing some scale for research and testing. Policies dictate that the participating organization allow the NDN testbed management institution administrative control of a machine dedicated as the participating organization's gateway, behind which may be any number of locally controlled NDN nodes. Sandia National Laboratories' (SNL) participation in the NDN testbed may be beneficial for gaining insights into the general usage and structure of the NDN protocol. SNL may be able to leverage an existing university collaboration to find a partner already participating in the NDN testbed. See [3] for a list of participating organizations.

1.6. Security

Ongoing research in network security and privacy describes and proposes those mitigations that apply to the risks inherent in NDN [4]. In particular, the following security risks and mitigations are described:

- Denial of Service
    - Software Defined Networking for Per-flow Rate Limiting
    - Bloom filter for self-routing interests
    - Suspicious name prefixes aggregation/suppression in PIT
    - Proof of work puzzle solving
- Content Poisoning
    - Attack origin identification and cache prevention
    - Content digest comparison between interest and data packet
- Cache Pollution
    - Collaborative caching
- Secure Naming
    - Metadata
- Secure Routing and forwarding
    - Self-routing, validation, name resolvers, signatures
    - Namespace mapping, buffering and reassembly, augmenting NDN forwarding plane with NACK
- Application Security
    - Filtering, Anomaly Detection, attention to Cyber Physical architecture, application layer, trust and integrity

Similarly, the following privacy risks and mitigations are described below [4]:

- Timing attack on cache hit/miss time
    - Collaborative caching plus network coding
- Monitoring attack (traffic analysis and profiling censorship)
    - Efficient Secure Tunneling, Name Randomization
- Anonymity
    - Steganography, Encryption, Coding
- Protocol attack (discover cached content): NDN may have vulnerabilities
    - Object discovery attack using NDN prefix matching and exclusion feature
    - Prefix-based matching and scoping attacks
- Name and Signature
    - Name Obfuscation
    - Overlay Network

The NDN project overview [9] describes its design choice of securing content rather than container, as decoupling trust in the data from trust in the hosts. The overview lists trust models, network security, and content protection and privacy as technical challenges.

In [2] a security overview of NDN concludes that by naming and securing data directly, named data carrying its own certificates can be fetched from anywhere securely. Through careful naming conventions, the trust policies and access control can be systematically defined; security is not left to application developers. Noting the importance of privacy, [2] also reports the positive impact that the data requestor's information is not disclosed by the interest packet, and that a remaining privacy challenge is that the data producer's identity may be disclosed by the packet name and signature. Finally, [2] notes that NDN may increase the attack surface of routers, which store content and interest packet details while conducting NDN communications. However, the NDN Tech Report NDN-0065[1] describes the Content Store and Pending Interest Tables as effective measures in mitigating common sources of flooding attacks.

[1] https://named-data.net/publications/techreports/ndn-0065-1-pap/

2. OPERANT NETWORKS INNOVATION

NDN researchers, since 2010, have collaborated on an open-source code base for experimentation and development. Operant has innovated the existing technology toward industry requirements and commercialization, developing a specialized code base toward these ends.

2.1. Application to DER

Regarding a smart grid operating distributed energy resources, Operant seeks to demonstrate an information-centric smart grid architecture using named data networking to create scalability, security, and trust in the power grid.

According to [4], the smart grid is "a 21st century critical infrastructure with 20th century security." Operant is studying the implementation of NDN for the smart grid as an exemplar of the solutions that are possible for a system presenting complexity, a broad scope of evolving communicating entities, diverse requirements for communication reliability, bandwidth, and latency, plus requirements for scalable networking and security.

2.2. Using NDN Simulator

Satyajayant Misra's work [7] includes a co-simulation experiment of NDN and the power system depicted in Figure 2-1, examining a wide area monitoring protection and control use case with two-way dataflow between the physical and cyber systems. The cyber system is represented with the network simulator NS-3 [5], and implements communications with both IP and NDN modules. The NDN structure is achieved using ndnSIM [6], an NS-3 based NDN simulator.

The experiment showed zero percent packet loss with NDN, as well as better latency values for NDN as congestion increased, as compared to UDP and TCP.

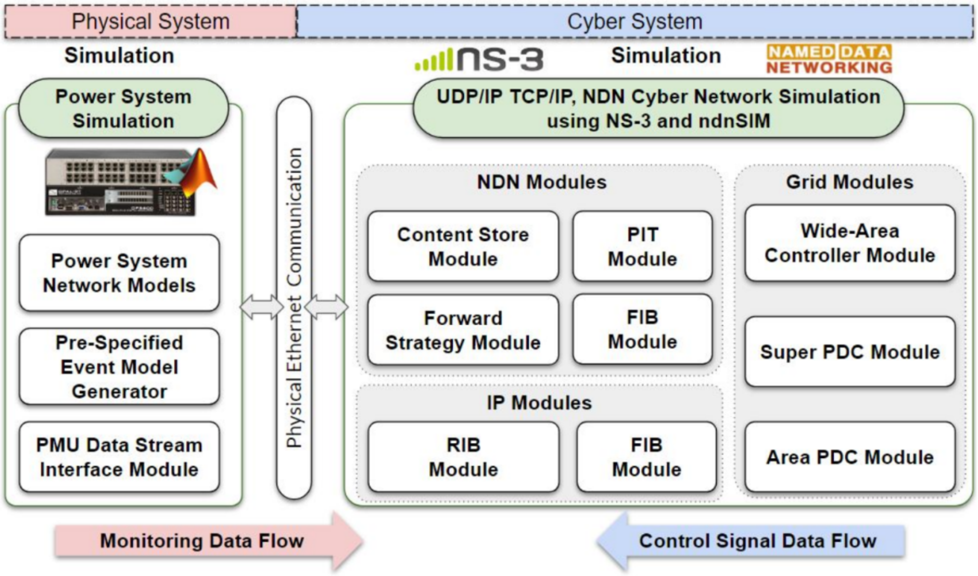


Figure 2-1. Co-Simulation Architecture

3. PRELIMINARY CYBERSECURITY OBSERVATIONS

From the initial context of the propaedeutic, the cybersecurity review begins with several observations regarding the Named Data Network concept.

The NDN project[2] overview lists trust models, network security, content protection and privacy as technical challenges. However, we did not find any status update that explicitly addresses these challenges. It is also noted that security at the packet level[3] is contingent on packets carrying signatures and names of the key locators. However, upon investigation of how this is implemented or who implements it, we had the following response from Satyajayant Misra:

"In the NDNfit example provided in [17], a data generator/consumer controls the entire system of policies, certificates and trust anchors that she relies on for confidentiality, authenticity and integrity. This is of course, not the only scenario required for networked data, as in many cases a data consumer will not be the same entity as the data generator."

This response does appear to be ambiguous regarding handling of scenarios where the consumer is not the only data generator.

The observation that NDN imposes the requirement of securing on a per-data basis as opposed to per-address appears to theoretically suggest larger demand on certificate management. When we asked about this possibility, Satyajayant Misra disagreed with that viewpoint. In particular, his response was that:

"In fact, the authors of [17] argue the opposite. That is, securing communication channels between every two communicating parties in a TCP/IP network "will dramatically increase the communication overheads."

In the security notes[4], it is implied that NDN may increase the attack surface of routers because they store content and interest packet details. If routers are storing pending interest requests, then they may indeed be prone to NDN interest request flood attacks. Satyajayant Misra suggested in [4] that flood attacks on routers are indeed of concern and the course to mitigation is currently in active development. Such mitigation plans may include filtering, rate-limitation, and proof of work. It is not clear to us how router's forwarding tables will be secured as well. On this concern, Satyajayant Misra did agree with the following note:

"Available research described in [4] reports that secure routing and forwarding are of concern and mitigations are in active development, such as self-routing and namespace mapping."

There is also concerns about privacy in the NDN communication. According to the security notes, data producer's identity may be disclosed by packet name and signature. We recommend a review of the construction for packet names and signatures. Satyajayant Misra, in response, noted that available research described in his work reports that the privacy of a data producer's identity is also of concern. Mitigation plans are in active development and may include features such as name obfuscation and overlay networks.

[2] https://named-data.net/project/
[3] https://named-data.net/wp-content/uploads/2019/02/SecurityOverview.pdf
[4] https://named-data.net/wp-content/uploads/2019/02/SecurityOverview.pdf

4. SECURITY EVALUATION

The purpose of this section is to provide guidance in conducting a secure code review and penetration testing for the NDN application. Secure code reviews are just one of a handful of activities that should be integrated into any software development life cycle in order to improve the security of the product. An authorized adversary-based assessment using the attack scenarios defined will be conducted to strengthen defenses through awareness of the system's potential vulnerabilities.

4.1. Test Objectives

The testing objectives are to determine how robust the software implementation of the NDN protocol is and to make suggestions for potential improvements based on the evaluation.

4.2. Scope of Testing

The security evaluation will focus on the NDN protocol and its software implementation, which includes NDN and NFD layers.

4.3. Testing Strategy

The security evaluation plan is to

- Describe the approach used to perform the high-level analysis of the NDN protocol and software. This will involve defining key attack scenarios to identify the security threats for possible exploitation. The scenarios are informed by experience from other network protocols and our research findings around NDN.
- Perform software testing to evaluate the source codes and check for security loopholes and vulnerabilities. This will involve static analysis or white box method to search for inconsistencies and errors in the source code. We will also apply black box method to evaluate potential issues with running the software.
- Provide guidance for a checklist of issues to review when identifying possible security defects.

4.4. Toolsets

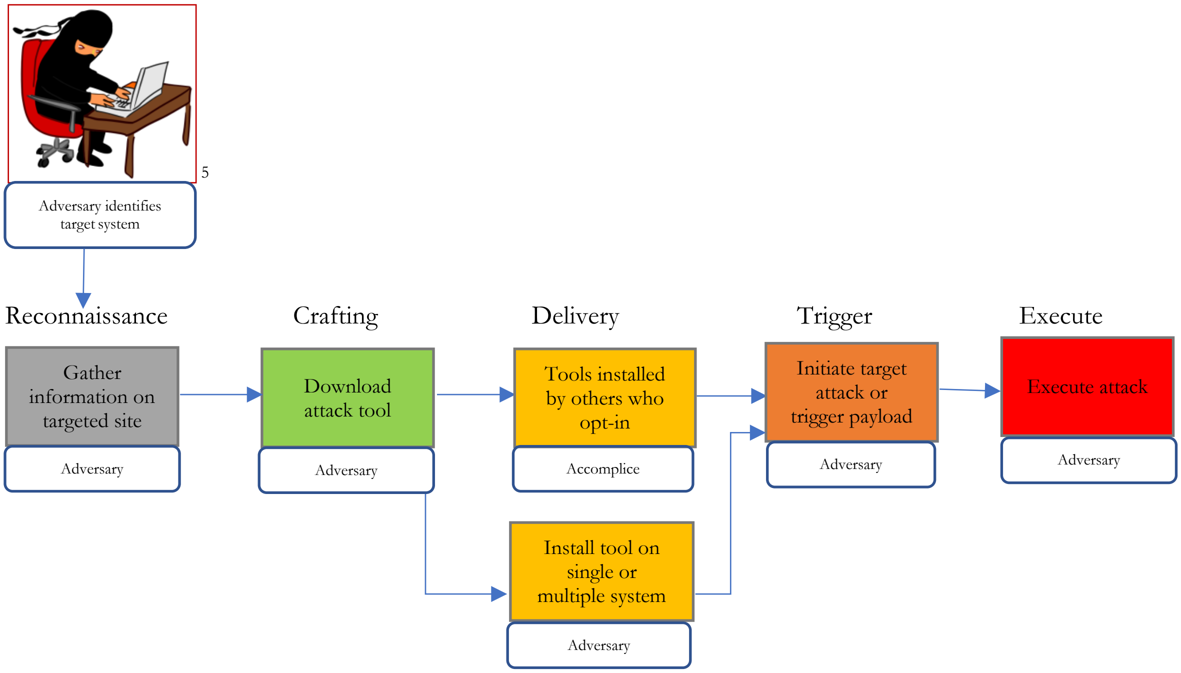
The list of potential toolsets that will be used are enumerated in section 7.1.

5. ATTACK SCENARIOS

Cybersecurity for DERs is an essential component to assuring the safety, reliability and resilience of the smart grid. As development investments and technologies are made to advance security capabilities for the future energy sector, it is important that their security capabilities are assessed. NDN is viewed as a next generation technology. Thus, understanding the effectiveness of the security going from design principles to implementation is paramount. This section explores the possible loopholes in the protocol specifications and the assumptions that were made for possible exploitation. We use the observations gleaned from the cybersecurity review to design test scenarios to explore the effectiveness of the NDN security. Also, NDN should address the following security requirements to protect, preserve, and promote trust in identity and data transactions.

- *Availability* to ensure that access to data is provided when needed.
- *Confidentiality* to protect information from unauthorized or unintended disclosure.
- *Integrity* to provide mechanisms to detect unauthorized (intentional or unintentional) data modifications, dropped or repeated messages.
- *Authentication* to provide assurance that the protected data came from an authenticated entity.
- *Authorization* to establish the access requirements, namely which users, systems or applications may read, write, create, delete, etc. specific types of information.
- *Non-repudiation* to provide the assurance of the origins of data in authenticated transactions.

The use of simulated and real-world assets with induced attack methods will be test platforms for developing and securing NDN. The attack scenarios that will be described will be modeled after the threat methodology described in **Error! Reference source not found.** below.


[5]
Adversary identifies target system
Reconnaissance
Crafting
Delivery
Trigger
Execute
Gather information on targeted site
Adversary
Download attack tool
Adversary
Tools installed by others who opt-in
Accomplice
Initiate target attack or trigger payload
Adversary
Execute attack
Adversary
Install tool on single or multiple system
Adversary

5-1: Generic Adversary Workflow

[5] Image from Wikimedia Commons

5.1. Distributed Denial of Service Attack

Distributed Denial of Service (DDoS) attacks are common attacks used to disrupt services: devices, applications, websites, etc. to make them unavailable to authorized users. Protecting against DoS or DDoS attacks in Operational Technology (OT) environments is critical to maintain high availability and low latency communications. DoS and DDoS attacks exist in all communication networks, including NDN. Within NDN, the adversary could launch an interest flooding attack by generating a large volume of interest packets intentionally requesting data that does not exist. The growing number of interest packets will cause the Pending Interest Tables of the routers to fill and consequently, congest the network. However, the NDN Tech Report NDN-0065[6] states that NDN employs effective mitigation to interest packet flooding. To validate this mitigation, the attack can be coordinated by several systems on the network to launch a DDoS attack at the same time and explore the availability of the NDN routers and network under a DDoS attack. A successful attack depends on the number of adversaries participating in the DDoS and the volume of traffic overcoming the threshold of interest packets that the routers can handle.

Step	Description
1	Generate a large volume of interest packets with different data names across multiple systems.
2	Coordinate between all systems to transmit the interest packets at the same time to fill the Pending Interest Tables in the routers.

Table 5-1: Interest Packet Flooding

5.2. Content Poisoning Attack

The adversary could respond to interest packets and provide false interest messages to the requester or consumer. Alternatively, the adversary could compromise a router and respond to interest packets with false data. The forged content is modified with valid content names to avoid detection. This is a data poisoning attack within NDN. Poisoning attacks are of two types, either with or without valid signing information [19]. Those without valid signatures are more common but will be detected by signature verification and therefore should fail as corrupted. This attack scenario explores the integrity of data from either a malicious provider or a collaboration of bad producers and consumers or a compromised router. If a router is compromised and the adversary attempts to modify the data, then checking the signature will expose the attacker. However, if a malicious provider is in place, then the attack is more challenging to detect since the producer will have legitimate credentials to correctly sign the false data. This attack is equivalent to a false-data injection attack within traditional computer networks but modified to include components of NDN. The goal of the adversary may be to disperse fake information thus forcing consumers to keep requesting for valid data. This may also result in a DoS scenario.

Step	Description

[6] https://named-data.net/publications/techreports/ndn-0065-1-pap/

1	Adversary introduces their own producer system into the network.
2	Adversary responds to interest packets with false data to poison the data.

Table 5-2: Data Poisoning Attack using a malicious provider

Step	Description
1	Adversary compromises a router in the network.
2	Adversary intercepts and responds to interest packets with false data that is not signed to poison the data.

Table 5-3: Data Poisoning Attack using a compromised router

5.3. Cache Pollution

The goal of a cache pollution attack is to increase network traffic by forcing unpopular content to be cached more prevalently than popular content, requiring more request forwarding than necessary [19]. This attack is drawn from proxy cache pollution attacks in traditional networks but tailored to the NDN infrastructure. This attack scenario will test whether any security mitigations are in place and if network performance is degraded.

Step	Description
1	Generate a large volume of unpopular interest packets with different data names across multiple systems to ensure frequent caching.

Table 5-4: Cache Pollution

5.4. Name Hijacking

Authentication of data in NDN does not rely on the data hosts. Instead, this functionality is attached directly to the naming scheme of the data [21], and applications are responsible for naming their data [20]. In order to uniquely name data, cryptographic hashes can be applied [4]. This attack scenario will verify whether name hijacking is prevented. The name hijacking attack specific to NDN is equivalent to DNS hijacking attacks in traditional networks. In addition to checking authentication, data integrity is also explored.

Step	Description
1	Adversary captures content names.
2	Explore the feasibility of manipulating naming schemas.

Table 5-5: Naming Hijacking

5.5. Route Hijacking

A route hijacking attack is proposed to test NDN's secure routing. The Named Data Networking FAQ [20] claims that because all NDN packets are signed, updates to NDN routing which are exchanged by NDN packets are automatically secure. This attack scenario will verify whether the trust policies and anchors are in place for signing and verifying the routing updates. This attack is like false route advertisements in traditional networks.

Step	Description
1	Adversary compromises a router in the network.
2	Adversary advertises bogus routes.

Table 5-6: Route Hijacking

5.6. Application Hijacking

Not only do applications name their own data, but they may also optionally encrypt their data and distribute the applicable keys [22]. This attack will verify whether an application with encrypted data securely distributes the applicable keys. This attack will analyze the key exchange algorithms and protocols which is critical in any key management system.

Step	Description
1	Adversary captures encrypted data.
2	Explore the designed key distribution schema for possible loopholes.

Table 5-7: Application Hijacking

5.7. Man-in-the-middle Attack

The adversary could perform a Man-in-the-Middle (MITM) attack by directly connecting, inline, to a communication channel. MITM is a network attack in which data transmissions are maliciously dropped, delayed or altered while in transit from producer to consumer or vice versa. The adversary can exfiltrate unencrypted data and eavesdrop on the communication channel. The adversary can also modify data in the communication stream. However, if the adversary modifies data, they can be detected since the signature will not be valid when checked against the modified data. This attack can be applied to any network but will focus on validating the signature validation of data that is built into NDN. To accomplish this attack, the adversary would require physical access to a communication link so that they can intercept and forward packets between a victim and the network. This scenario explores the confidentiality and integrity of data transfers.

Step	Description
1	Adversary can directly connect, inline, to a communication link.
2	Adversary will intercept data and forward/drop or modify packets as desired.

Table 5-8: MITM Data Exfiltration of Unencrypted Data

5.8. Replay Attack

Packet replay is a network attack in which data is captured and then maliciously replayed or repeated later. To begin the attack, network communication will be compromised, and data will be captured. The communications can be compromised by an inline adversary, a compromised router, or a compromised consumer. This attack is applicable to traditional networks and NDN specific components will be analyzed. The messages intercepted by the adversary are saved so that they can be replayed later. The adversary can effectively inject previous requests as desired. The goal of the adversary is to cause the victim to believe the replayed packets are new and legitimate packets. Replay attacks can be used to trick the victim into performing actions based on a sequence of packets.

Step	Description
1	Adversary compromises a communication link and captures communications.
2	Adversary organizes the packets so that a set of actions will be performed to replay traffic as inclined.

Table 5-9: Replay Attack

5.9. Anonymity and Censorship Attack

The adversary can eavesdrop and intercept messages to reveal information about the consumers and contents requested (interest packets). This attack depends on the MITM attack previously described but can be orchestrated by the administrator of the network. This attack can be applied to traditional networks but will focus on NDN components. The adversary can match the interest packets to the data packets which will then be dropped or redirected to censor requests. The goal of the adversary in this scenario may be to maintain stealth so that the victim is unaware of the ongoing attack.

Step	Description

1	Adversary compromises a communication link and captures requested packet data.
2	Adversary uses the information to drop or censor requested data packets.

Table 5-10: Anonymity and Censorship Attack

5.10. Privacy Attack of Interest Packets

The adversary could monitor interest packets or interest tables to learn about sensitive information being queried or cached. The adversary can also determine if content has been cached by timing the amount of time it takes for data to be returned to the adversary. Additionally, since the routers cache interests in NDN, the adversary can learn what others are querying. Depending on the timing of the packets returned, the adversary can determine if data has been cached. The adversary can learn about timings by sending the same interest packets multiple times and comparing the first interest packet to the next interest packet. If the second interest packet is returned faster, then that can inform the adversary that the router is caching. If the router is caching requests, then the adversary can send additional interest packets on different topics to learn if other consumers have requested that same data. Additionally, an adversary can discover the structure of the data by sending interest packets starting from root ('/') and then probing again for new data based on what is received from the root request. The process can be repeated until the entire structure of the data is received. This attack is like web crawling, directory traversal, or SNMP walking in traditional networks.

Step	Description
1	Adversary compromises a communication link and captures communications.
2	Adversary observes interest packets and monitors for sensitive queries.

Table 5-11: Privacy Attack of Interest Packets

Step	Description
1	Adversary inserts their own malicious router or compromises a router.
2	Adversary observes cached interest table and monitors for sensitive queries.

Table 5-12: Privacy Attack of Interest Tables

Step	Description

1	Adversary measures baseline of round-trip time for cached interests and interests that are not cached.
2	Adversary measures future interest requests to determine if the timing is consistent with cached data or data that has not been cached to infer what other users are querying.

Table 5-13: Cache Timing Attack

Step	Description
1	Adversary first sends an interest packet with a request of "/" to obtain the root of the data.
2	The adversary will receive data back that has been cached under the root namespace.
3	The adversary repeats step 2 using the namespace data that was received most recently. This process is repeated until the entire namespace is received.

Table 5-14: Object Discovery Attack

5.11. Reflection Attack

The adversary could coerce other secondary victims or reflectors to attack a victim host by generating a large volume of interest packets with the victim's IP address. The fake packets with the victim's IP address are sent to the secondary victims to force the victim host to respond to the forged packets. This attack scenario will verify the resiliency of NDN since data can only flow back through the path established by the preceding interest path.

Step	Description
1	Adversary compromises a communication link of an NDN router and generates a large amount of interest packets
2	Adversary sends forged packets to the reflectors to attempt to overwhelm the victim host with responses

Table 5-21: Reflection Attack

5.12. Caveats

Implementation of the attack scenarios described herein will depend upon the architecture and implementation of the NDN DER test network provided. In emulating the DER protocols or devices, there may necessarily be many levels of abstraction pertaining to the device and protocol models themselves, the communication backbone, or any given hardware in the loop for example.

The given attack scenarios will be further developed as necessary under discussion with the test network providers.

6. WHITE BOX SECURITY TESTING AND UNIT TESTING

Static application security testing scans on Operant's `ndn-ind` and `nfd-ind` code repositories were performed, and the highest severity vulnerabilities enumerated and examined. Additionally, verification of the unit tests in both repositories was performed against the code in the master branch as of June 3, 2020.

6.1. SAST Test Cases

The SAST tests described in the table below, reviews the source codes in the `ndn-ind` and `nfd-ind` repositories to detect and report high risk security vulnerabilities.

Category	Checklist	Tools
NDN-ind Static Code Analysis	Insecure code	CxSAST, CppCheck
NFD-ind Static Code Analysis	Insecure code	CxSAST, CppCheck

Table 6-1: SAST Test Cases

6.1.1. Static Application Security Testing (SAST) Phase I

To uncover security vulnerabilities in the NDN codebase, the security team performed white box analysis, by running static application security testing (SAST) tools to find language specific vulnerabilities. For both the `ndn-ind` project (primarily using the C programming language) and the `nfd-ind` project (primarily using the C++ programming language) we compiled the code, and performed SAST scans of the master branches of the two repositories (as of June 4, 2020) using Checkmarx CxSAST v. 8.8.0 and Cppcheck v. 1.88. We used Code Dx v. 5.0.3 Enterprise to correlate and de-duplicate findings. The team then reviewed each vulnerability by examining the code in a white box manner to determine exploitability. Due to time constraints, we limited our analysis to findings classified as "high" severity by each of these tools, which resulted in 20 findings.

The table below shows profiles for the `ndn-ind` and `nfd-ind` projects and the number of SAST scan findings.

	`ndn-ind`	`nfd-ind`
Lines of Code	406,198	47,568
Source Files	583	332
Cyclomatic Complexity	2.21	2.39
Finding Density	1.79/KLOC	2.08/KLOC
High-Severity Findings	19	1
Medium-Severity Findings	382	54

Low-Severity Findings	327	42

Table 6-2: SAST Profiling I

Given the method in which these tests are performed, it is difficult to determine whether these findings identified by the SAST tools lead to actual exploitable attacks given the budget. Without proper dynamic testing and a proof-of-concept attack, the security team was unable make definitive claims about the exploitability of these vulnerabilities. As a result, we performed our analysis of these findings based solely on reviewing the source code and severity levels assigned by the SAST tools.

Most of these findings are related to third party libraries. `Gtest` and `sqlite` comprised many of the vulnerabilities discovered by SAST tools; we recommend updating these libraries to the most recent version to minimize available attack surface. While `gtest` is ostensibly only used for unit testing, `sqlite` presents a specific set of vulnerabilities that may be present in these libraries' production environment.

6.1.2. SAST Findings

Repo	Finding Type	Location	Tool	Tool Rating	Confirmed?	Evaluated Rating	Notes
ndn-ind	CWE-119	ndn-ind/contrib/gtest-1.7.0/fused-src/gtest/gtest.h:2869	CxSAST	High	Yes	Low, testing library	Finding occurs in third-party code Use of an unsafe function strncpy This function is dangerous because it can cause buffer overflows https://devblogs.microsoft.com/oldnewthing/20050107-00/?p=36773 Recommend updating, latest version on google doesn't use this function
ndn-ind	CWE-119	ndn-ind/contrib/openssl/internal/md32_common.h:339	CxSAST	High	N/A	N/A but recommend upgrading	Finding occurs in third-party code Unclear what version is used here, but would recommend updating this library as it is regularly updated to patch CVE findings.
ndn-ind	CWE-119	ndn-ind/contrib/sqlite3/sqlite3.c:44867	Cppcheck	High	N/A	N/A but recommend upgrading	Finding occurs in third-party code CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-119	ndn-ind/src/interest-filter.cpp:118	Cppcheck	High	No	FP	While this line of code doesn't make functional sense to me, in C++ it is allowable to check s[0] of an empty string s https://stackoverflow.com/questions/26310772/why-an-empty-string-can-output-index-0-element-in-c
ndn-ind	CWE-562	ndn-ind/contrib/sqlite3/sqlite3.c:18726	Cppcheck	High	N/A	N/A but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-562	ndn-ind/contrib/sqlite3/sqlite3.c:18768	Cppcheck	High	N/A	N/A but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-562	ndn-ind/contrib/sqlite3/sqlite3.c:122619	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-562	ndn-ind/contrib/sqlite3/sqlite3.c:125531	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html

Repo	Finding Type	Location	Tool	Tool Rating	Confirmed?	Evaluated Rating	Notes
ndn-ind	CWE-399	ndn-ind/contrib/sqlite3/sqlite3.c:104996	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-399	ndn-ind/contrib/sqlite3/sqlite3.c:127347	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-399	ndn-ind/contrib/gtest-1.7.0/fused-src/gtest/gtest-all.cc:7485	Cppcheck	High	False Positive	N/A, but recommend upgrading	Finding occurs in third-party code that is used for unit testing only The issue is a FP because it occurs in a context for which it is a non-issue
ndn-ind	CWE-758	ndn-ind/contrib/sqlite3/sqlite3.c:107274	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-758	ndn-ind/contrib/sqlite3/sqlite3.c:107275	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-401	ndn-ind/contrib/sqlite3/sqlite3.c:20798	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-401	ndn-ind/contrib/sqlite3/sqlite3.c:20863	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-77	ndn-ind/src/security/tpm/tpm-back-end-file.cpp:75	CxSAST	High	Possible True Positive	High	Possible user input sent to system call. While the line in question attempts to provide safety check by hard coding double quotes, it should be investigated whether the keyStorePath_ variable can be set by the user to include terminating quotes and malicious command injection payloads.
ndn-ind	CWE-562	ndn-ind/src/util/boost-info-parser.hpp:104	Cppcheck	High	False Positive	N/A	Finding identifies possible problem returning an invalid list, but code clearly checks size of list before returning.
ndn-ind	Null Pointer	ndn-ind/contrib/sqlite3/sqlite3.c:89174	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
ndn-ind	CWE-189	ndn-ind/contrib/sqlite3/sqlite3.c:24499	Cppcheck	High	N/A	N/A, but recommend upgrading	Finding occurs in third-party code: sqlite3 version 3.19.3 CVEs exist https://www.cvedetails.com/vulnerability-list/vendor_id-9237/Sqlite.html
nfd-ind	CWE-189	nfd-ind/tools/ndn-autoconfig/dns-srv.cpp:97	CxSAST	High	No	Need more information	A memory address is indexed using a sizeof(var), which appears to be of a dynamic nature. More information about how the memory is organized at this part of the code is needed to make any sort of definitive statement. Will require dynamic testing review to determine applicability of the vulnerability.

Table 6-3: SAST Finding Detail

6.1.3. SAST Finding Discussion

Upon re-scanning the code after security patching, only three possible findings remained that required manual verification:

Repo	Finding Type	Location	Tool	Tool Rating	Confirmed?	Evaluated Rating	Notes
ndn-ind	CWE-119	ndn-ind/contrib/gtest-1.7.0/fused-src/gtest/gtest.h:2869	CxSAST	High	Yes	Low, testing library	Finding occurs in third-party code Use of an unsafe function strncpy This function is dangerous because it can cause buffer overflows https://devblogs.microsoft.com/oldnewthing/20050107-00/?p=36773 Recommend updating, latest version on google doesn't have use function
ndn-ind	CWE-77	ndn-ind/src/security/tpm/tpm-back-end-file.cpp:75	CxSAST	High	Possible True Positive	Low	Possible user input sent to system call. While the line in question attempts to provide safety check by hard coding double quotes, it should be investigated whether the keyStorePath_ variable can be set by the user to include terminating quotes and malicious command injection payloads.
nfd-ind	CWE-189	nfd-ind/tools/ndn-autoconfig/dns-srv.cpp:97	CxSAST	High	No	N/A	A memory address is indexed using a sizeof(var), which appears to be of a dynamic nature. More information about how the memory is organized at this part of the code is needed to make any sort of definitive statement. Will require dynamic testing review to determine applicability of the vulnerability.

The first finding in the ndn-ind library is specific to googletest v1.7.0, a third-party unit test library. As it is unlikely that a unit test would pose a risk to NDN functionality, we do not recommend any action on Operant's part to address this, although an update to the latest stable release (1.10) may prevent future SAST findings related to strcopy.

The second finding in the ndn-ind library involves a possible command injection vulnerability related to a keyStorePath variable used in a system() call in tpm-back-end-file.cpp. On further analysis, the only input that is modifiable here to create the system() callis getenv("HOME"). This is a standard library call that pulls the current HOME directory for the given user. To exploit this, it must be possible for a user to modify the HOME directory; if a user has access to the HOME directory and can rewrite it, they already have access to the host, which means that this particular exploit would be irrelevant to an attacker.

The lone finding in the nfd-ind library involves a possible memory address issue in the dns-srv.cpp file. Deeper analysis shows that the sizeof() call that triggers the finding is related to a struct with fixed size (rechdr). As such, an overflow issue is not possible because the size of the object evaluated by the sizeof() method can never exceed the size of the struct. We recommend no changes to the nfd-ind library at this time.

6.1.4. Static Application Security Testing (SAST) using Code Dx

Static application security testing scans on Operant's `ndn-ind` and `nfd-ind` code repositories were performed using Code Dx, and the highest severity vulnerabilities enumerated and examined. Additionally, verification of the unit tests in the `ndn-ind` repository was performed against the code in the master branch as of January 2021.

The table below shows the overall project profiles for the `ndn-ind` and `nfd-ind` projects and the number of SAST scan findings.

	ndn-ind	**nfd-ind**
Lines of Code	177,669	131,224

Source Files	608	737
Cyclomatic Complexity	1.91	1.98
Finding Density	0.48/KLOC	1.37/KLOC
High-Severity Findings	1	88
Medium-Severity Findings	15	26
Low-Severity Findings	69	33
Info-Severity Findings	0	33

Table 6-4: SAST Profiling using Code Dx

For the ndn-ind repo, files in the following sub-directories were ignored and does not include any severity issues from these excluded directories.

- ndn-ind-master/contrib/gtest-1.7.0
- ndn-ind-master/tests
- ndn-ind-master/examples

For the nfd-ind repo, files in the following sub-directories were ignored and does not include any severity issues from these excluded directories.

- nfd-ind-patched/tests
- nfd-ind-patched/.waf-tools
- nfd-ind-patched/docs

Below are counts from Code Dx scans on the latest code in the repos/branches below that excludes the directories listed above.

	Repo à	ndn-ind	nfd-ind
Severity	High	0	6
	Medium	9	26
	Low	51	29
	Info	0	2

Table 6-5: Counts from Code Dx

6.1.5. SAST Findings using Code Dx

Code Dx Enterprise v. 5.2.10 was used, and the high-severity findings below come from these specific tools bundled with Code Dx:

- Cppcheck v. 1.88

- ESLint v. 7.4.0
- Pylint v. 2.4.4

Following are the high-severity findings, not including any issues from the excluded directories. All the findings are false positives and appear in test code in a third-party dependency.

Repo	Finding Type	Location	Tool	Tool Rating	Confirmed?	Evaluated Rating	Notes
nfd-ind	Unknown macro	websocketpp/test/utility/sha1.cpp:37	Cppcheck	High	No	N/A	False positive finding. Issue is in test case code in a third-party dependency.
nfd-ind	Use of unsafe function	websocketpp/examples/telemetry_server/index.html:57	ESLint	High	No	N/A	False positive finding. Issue is in example code in a third-party dependency.
nfd-ind	CWE-79	websocketpp/examples/telemetry_server/index.html:24	ESLint	High	No	N/A	False positive finding. Issue is in example code in a third-party dependency.
nfd-ind	CWE-79	websocketpp/examples/telemetry_server/index.html:41	ESLint	High	No	N/A	False positive finding. Issue is in example code in a third-party dependency.
nfd-ind	CWE-79	websocketpp/examples/telemetry_server/index.html:36	ESLint	High	No	N/A	False positive finding. Issue is in example code in a third-party dependency.
nfd-ind	Unknown macro	websocketpp/test/utility/utilities.cpp:36	Cppcheck	High	No	N/A	False positive finding. Issue is in test case code in a third-party dependency.

6.2. Unit Tests

The unit tests in the `ndn-ind` repository were executed to verify the workings of the codes and to understand the level of rigor taken into consideration during code writing.

6.2.1. ndn-ind

The `ndn-ind` library unit tests were performed on the master branch using both Ubuntu v18.04 and macOS Mojave v10.14.6, and revealed the following:

```
============================================================================
     Testsuite summary for ndn-ind 0.17
============================================================================
     # TOTAL: 30
     # PASS:  28
     # SKIP:  0
     # XFAIL: 0
     # FAIL:  2
     # XPASS: 0
     # ERROR: 0
```

The failed tests were `test-face-methods` and `test-registration-callbacks`. These failures resulted from socket connection failures and "No default identity" exceptions thrown from the `TestFaceRegisterMethod's` test fixture, and a "No default identity" exception thrown from the `TestRegistrationCallback` test fixture.

These tests were rerun on January 19, 2021 and the same 28 unit tests passed out of a total of 30. The two failing tests are the same ones that failed as described above:

```
bin/unit-tests/test-face-methods
bin/unit-tests/test-registration-callbacks
```

In addition to the README-dev.md file and the information in the updated repository, these were the additional steps taken to get the codes to compile.

1. Install additional library

```
apt install libsystemd-dev
```

2. Provide additional parameters when building nfd-ind with unit test support (to get further before getting the compile errors above):

```
./waf configure --with-tests --boost-includes=/usr/local/include --
boost-libs=/usr/local/lib
```

7. PENETRATION TESTING

The black box penetration testing will focus on security and implementation problems specific to NDN that will bypass or break security such as described in the attack scenarios.

7.1 Test Cases

Category	Checklist	Potential Tools
Reconnaissance	Involves both active and passive information gathering about the target system	Nmap, OpenVAS, Wireshark, Nessus, Metasploit
Interruption	Involves obstruction to communication and rendering the system unavailable to legitimate users	Hping, Metasploit
Interception	Involves altering communication between two or more users or entities	Ettercap, Metasploit
Packet Replay	Involves maliciously replaying or repeating data transmissions	Tcpdump, Tcpreplay
Using Components with Known Vulnerabilities	Involves exploiting vulnerable code & components to undermine defenses OWASP Top 10 Risks (https://owasp.org/www-project-top-ten/)	CxSAST, CppCheck
Content Poisoning	Involves remote providers disseminating bad data	Ettercap, Routersploit,
Injection	Involves sending malicious data to disclose or corrupt data OWASP Top 10 Risks (https://owasp.org/www-project-top-ten/)	Scanning & Fuzzing tools
Sensitive Data Exposure	The act of compromising and exfiltrating unencrypted sensitive data OWASP Top 10 Risks (https://owasp.org/www-project-top-ten/)	Wireshark, Ettercap, Metasploit.
Broken Access Control	Involves exploiting access not properly enforced OWASP Top 10 Risks (https://owasp.org/www-project-top-ten/)	Metasploit
Security Misconfiguration	Involves exploiting security controls that are not securely implemented OWASP Top 10 Risks (https://owasp.org/www-project-top-ten/)	Nmap, Metasploit, OpenSCAP

Table 7-1: Penetration Test Cases

7.2 Test Results

Vulnerability assessment and penetration testing are focused on finding and exploiting flaws to determine the security of systems. The tests for the assessment are designed to:

1) Find vulnerabilities in the software and hardware
2) Perform network service tests to exploit information from the operating system and network services
3) Manipulate available metadata from documents in the repositories
4) Find common types of attacks against the NDN network and nodes on the network
5) Determine if data is modified without adequate validation such that the modified data causes the nodes to malfunction or allow access to unauthorized users

7.2.1 Reconnaissance

The penetration testing was conducted using a VPN connection into Operant's NDN environment. The testbed was created with four NDN nodes. A computer with Kali Linux software was added to the network to evaluate the security of the systems and is shown in Figure 7-1 below.

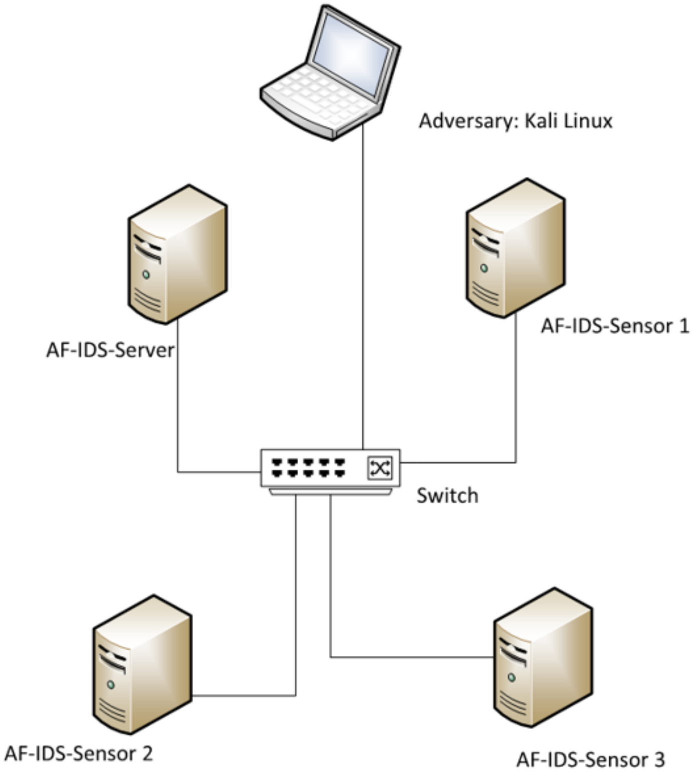
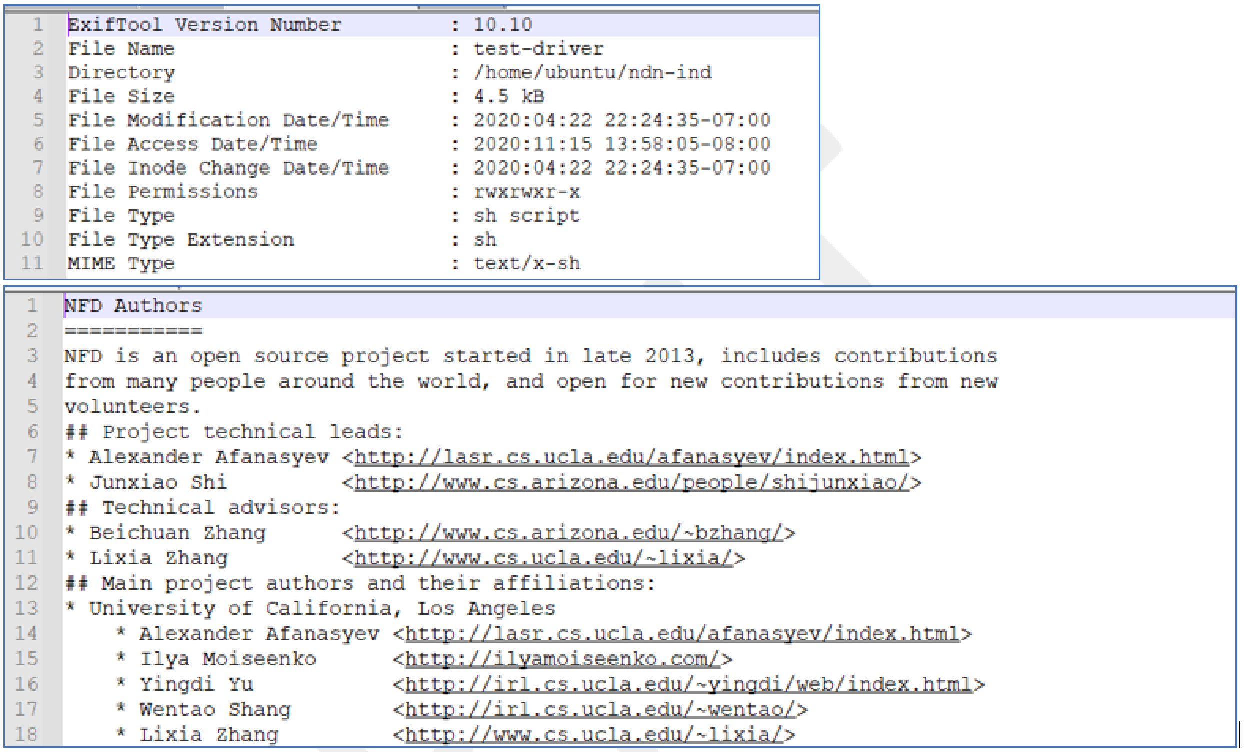


Figure 7-1: NDN testbed for penetration testing

A.1.1.1. Meta-Data Analysis

Exiftool [7] and the Linux strings commands are well-known tools for analyzing document metadata. They were used as the meta-data extractor, revealing the information about the documentation in the repositories. This information is shown below.



```
ExifTool Version Number         : 10.10
File Name                       : test-driver
Directory                       : /home/ubuntu/ndn-ind
File Size                       : 4.5 kB
File Modification Date/Time     : 2020:04:22 22:24:35-07:00
File Access Date/Time           : 2020:11:15 13:58:05-08:00
File Inode Change Date/Time     : 2020:04:22 22:24:35-07:00
File Permissions                : rwxrwxr-x
File Type                       : sh script
File Type Extension             : sh
MIME Type                       : text/x-sh
```

```
NFD Authors
===========
NFD is an open source project started in late 2013, includes contributions
from many people around the world, and open for new contributions from new
volunteers.
## Project technical leads:
* Alexander Afanasyev <http://lasr.cs.ucla.edu/afanasyev/index.html>
* Junxiao Shi         <http://www.cs.arizona.edu/people/shijunxiao/>
## Technical advisors:
* Beichuan Zhang      <http://www.cs.arizona.edu/~bzhang/>
* Lixia Zhang         <http://www.cs.ucla.edu/~lixia/>
## Main project authors and their affiliations:
* University of California, Los Angeles
    * Alexander Afanasyev <http://lasr.cs.ucla.edu/afanasyev/index.html>
    * Ilya Moiseenko      <http://ilyamoiseenko.com/>
    * Yingdi Yu           <http://irl.cs.ucla.edu/~yingdi/web/index.html>
    * Wentao Shang        <http://irl.cs.ucla.edu/~wentao/>
    * Lixia Zhang         <http://www.cs.ucla.edu/~lixia/>
```

Figure 7-2: Extracted metadata from the ndn-ind and nfd-ind repositories

Typically, software used to create documents embeds a significant amount of information in the document files. The Exiftool could not determine the file type and did not yield much useful information. The Strings metadata did not yield additional information that was not already contained in the file. However, information in the AUTHORS.md file revealed names, email addresses and URLs. This information can be used for social engineering by targeting valid users in those organizations and attempting to trick them into revealing sensitive information.

A.1.1.2. Scanning Phase

Nmap was used again to scan ports, fingerprint the OS, and enumerate services on each endpoint to help the pen-test team understand the target attack surface. These tests included running TCP Connect, TCP SYN, TCP ACK, UDP, and NSE scripting scans. The results of the network reconnaissance did not provide much useful information to the pen-test team and it was evident that the Intrusion Detection (IDS)was performing its job since most scans indicated that all ports were either filtered or open|filtered. To better understand the firewall rules and its state, the pen-test team performed TCP ACK scans. Despite using evasion techniques like packet fragmenting to bypass the

[7] https://exiftool.org/

firewall and IDS measures, the scans returned the same results and were often inconclusive due to long scan times and potential rate limiting.

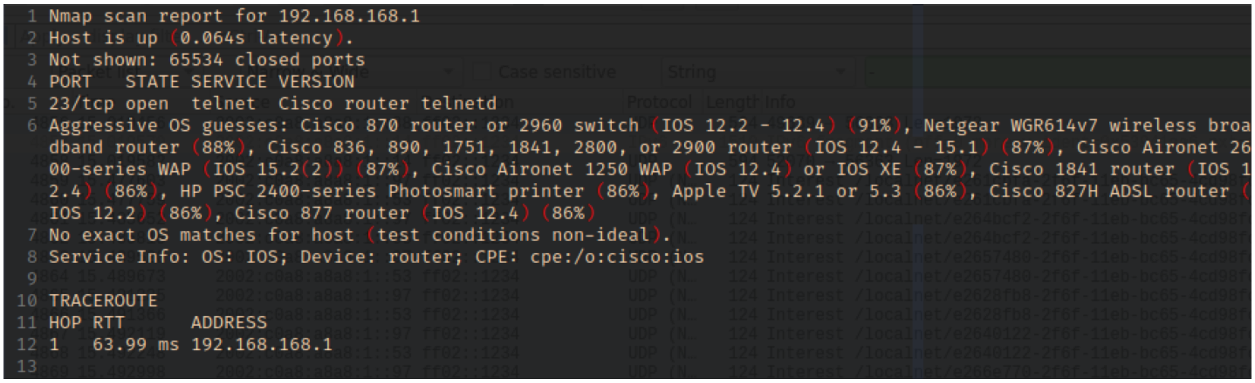
For the sensors, The TCP SYN scans provided the most information out of all the scan types. The results of these particular scans indicate that the endpoint sensors are running GNU Linux on an x86_64 architecture with a fairly high likelihood of being an Open DD-WRT device, although that firmware description ended up inaccurate. Nmap was able to identify port 22 was open and running OpenSSH Server v7.4 on all endpoints. The results also depicted NDN and some unknown services as closed on higher number ports. No major concerns were revealed by the NSE scripts apart from some public SSH keys displayed. These were helpful for identifying potentially weak encryption algorithms used for SSH.

After performing the active network scans, the pen-test team attempted to find known vulnerabilities and existing exploits for newly acquired OS and service information. Searchsploit[8] and ExploitDB[9] revealed possible username enumeration for the OpenSSH server. The auxiliary Metasploit module, ssh_enumusers, was used to confirm that the current configuration was not susceptible to a username enumeration attack using timed and malformed packets. Part of this verification used OSINT to gather employee information from the Operant webpage to create a wordlist using the common <firstname><lastname> format. The tests using control cases demonstrated that a pen-test team could not identify valid usernames, thus making it much more difficult and time consuming to brute force SSH credentials.

The data provided by Nmap on the four NDN nodes is summarized below.

operant-ndn-ids-router

- IP Address: 192.168.168.1
- Operating System Fingerprint: Cisco IOS Router (R6503)
- Open Ports: 23/TCP (telnet)



```
Nmap scan report for 192.168.168.1
Host is up (0.064s latency).
Not shown: 65534 closed ports
PORT   STATE SERVICE VERSION
23/tcp open  telnet  Cisco router telnetd
Aggressive OS guesses: Cisco 870 router or 2960 switch (IOS 12.2 - 12.4) (91%), Netgear WGR614v7 wireless broa
dband router (88%), Cisco 836, 890, 1751, 1841, 2800, or 2900 router (IOS 12.4 - 15.1) (87%), Cisco Aironet 26
00-series WAP (IOS 15.2(2)) (87%), Cisco Aironet 1250 WAP (IOS 12.4) or IOS XE (87%), Cisco 1841 router (IOS 1
2.4) (86%), HP PSC 2400-series Photosmart printer (86%), Apple TV 5.2.1 or 5.3 (86%), Cisco 827H ADSL router (
IOS 12.2) (86%), Cisco 877 router (IOS 12.4) (86%)
No exact OS matches for host (test conditions non-ideal).
Service Info: OS: IOS; Device: router; CPE: cpe:/o:cisco:ios

TRACEROUTE
HOP RTT      ADDRESS
1   63.99 ms 192.168.168.1

```

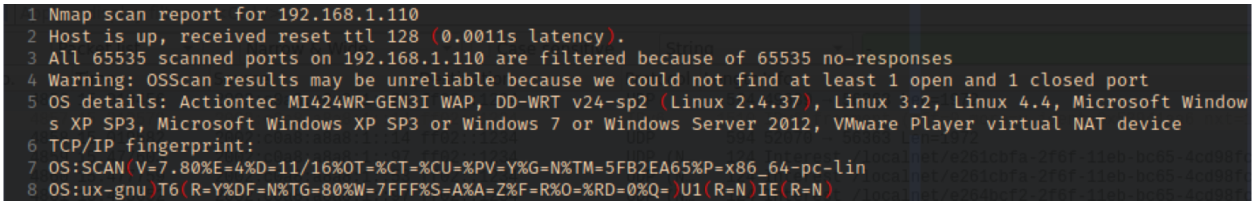
Figure 7-3: TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts

[8] https://www.exploit-db.com/searchsploit

[9] https://www.exploit-db.com/

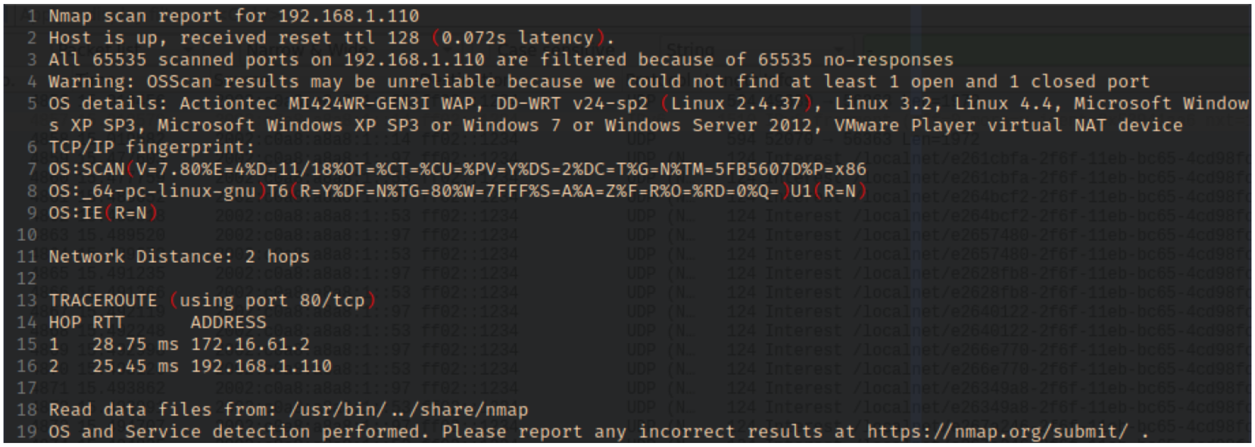
operant-ndn-ids-server

- IP Address: 192.168.168.110
- Operating System Fingerprint: GNU Linux (Likely Ationtec MI424WR-GEN3I WAP, DD-WRT v24-sp2, Linux 3.2, or 4.4)
- Architecture: Intel x86_64
- Ports: All ports filtered. This indicates the likelihood of the presence of a firewall/IDS



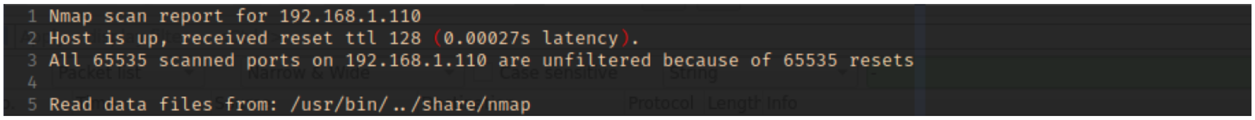
```
Nmap scan report for 192.168.1.110
Host is up, received reset ttl 128 (0.0011s latency).
All 65535 scanned ports on 192.168.1.110 are filtered because of 65535 no-responses
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
OS details: Actiontec MI424WR-GEN3I WAP, DD-WRT v24-sp2 (Linux 2.4.37), Linux 3.2, Linux 4.4, Microsoft Window
s XP SP3, Microsoft Windows XP SP3 or Windows 7 or Windows Server 2012, VMware Player virtual NAT device
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=11/16%OT=%CT=%CU=%PV=Y%G=N%TM=5FB2EA65%P=x86_64-pc-lin
OS:ux-gnu)T6(R=Y%DF=N%TG=80%W=7FFF%S=A%A=Z%F=R%O=%RD=0%Q=)U1(R=N)IE(R=N)
```

Figure 7-4: Full TCP Connect scan results indicating the likelihood of a firewall/IDS



```
Nmap scan report for 192.168.1.110
Host is up, received reset ttl 128 (0.072s latency).
All 65535 scanned ports on 192.168.1.110 are filtered because of 65535 no-responses
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
OS details: Actiontec MI424WR-GEN3I WAP, DD-WRT v24-sp2 (Linux 2.4.37), Linux 3.2, Linux 4.4, Microsoft Window
s XP SP3, Microsoft Windows XP SP3 or Windows 7 or Windows Server 2012, VMware Player virtual NAT device
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=11/18%OT=%CT=%CU=%PV=Y%DS=2%DC=T%G=N%TM=5FB5607D%P=x86
OS:_64-pc-linux-gnu)T6(R=Y%DF=N%TG=80%W=7FFF%S=A%A=Z%F=R%O=%RD=0%Q=)U1(R=N)
OS:IE(R=N)

Network Distance: 2 hops

TRACEROUTE (using port 80/tcp)
HOP RTT      ADDRESS
1   28.75 ms 172.16.61.2
2   25.45 ms 192.168.1.110

Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
```

Figure 7-5: TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts

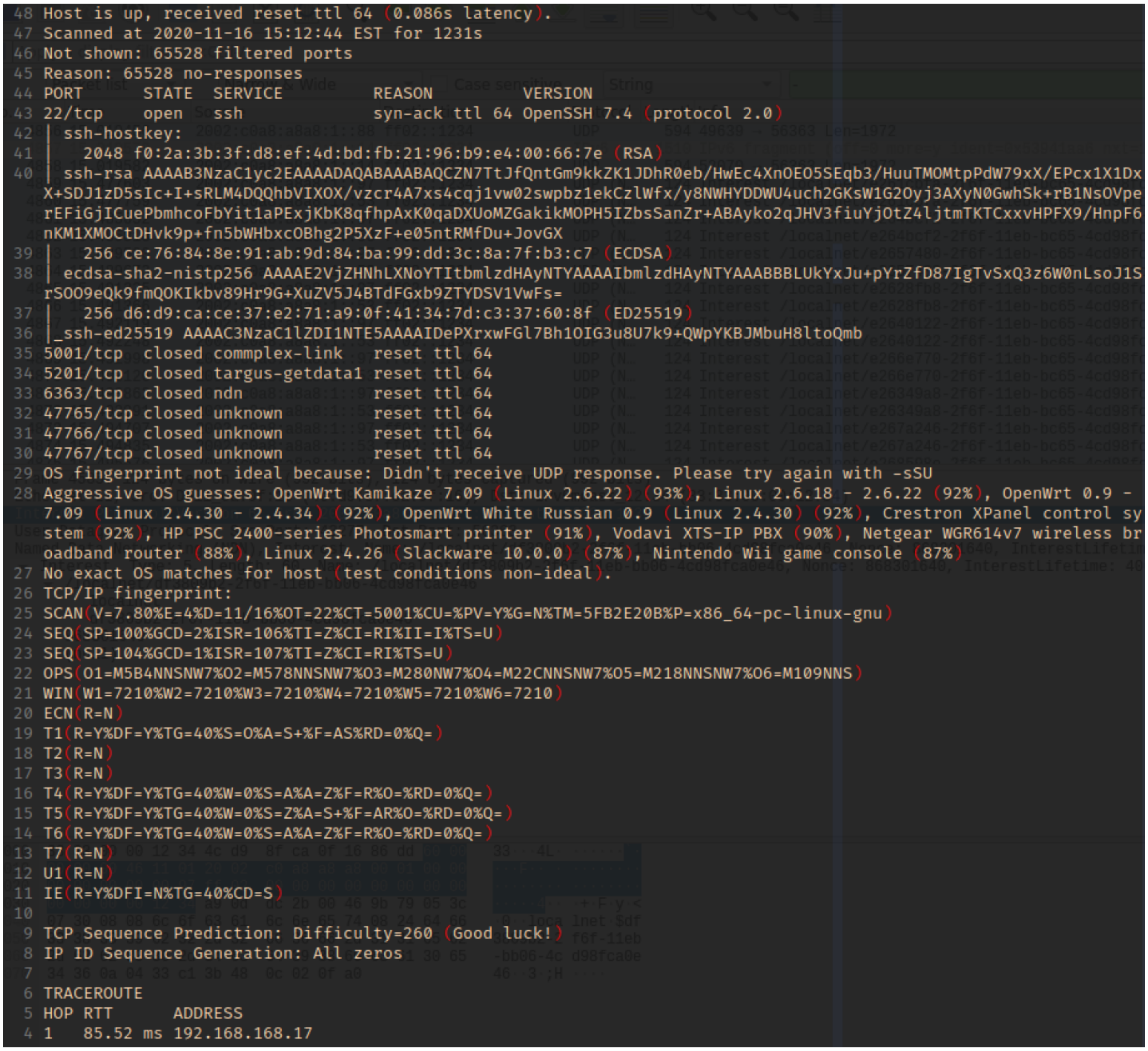


```
Nmap scan report for 192.168.1.110
Host is up, received reset ttl 128 (0.00027s latency).
All 65535 scanned ports on 192.168.1.110 are unfiltered because of 65535 resets

Read data files from: /usr/bin/../share/nmap
```

Figure 7-6: TCP ACK scans show all ports as unfiltered, hinting that the firewall/IDS is likely stateless

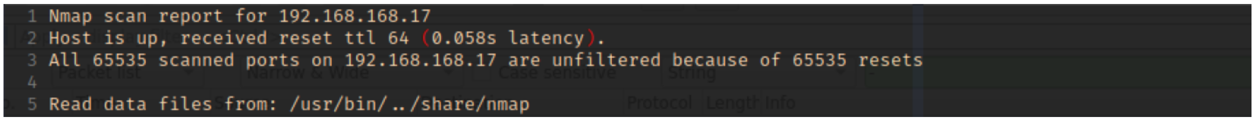
operant-ndn-ids-sensor1

- IP Address: 192.168.168.17
- Operating System Fingerprint: GNU Linux (Likely OpenWrt White Russian 0.9, OpenWrt Kamikaze)
- Architecture: Intel x86_64
- Open Ports: 22/TCP (OpenSSH Server v7.4)



```
Host is up, received reset ttl 64 (0.086s latency).
Scanned at 2020-11-16 15:12:44 EST for 1231s
Not shown: 65528 filtered ports
Reason: 65528 no-responses
PORT      STATE  SERVICE         REASON         VERSION
22/tcp    open   ssh             syn-ack ttl 64 OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 f0:2a:3b:3f:d8:ef:4d:bd:fb:21:96:b9:e4:00:66:7e (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCZN7TtJfQntGm9kkZK1JDhR0eb/HwEc4XnOEO5SEqb3/HuuTMOMtpPdW79xX/EPcx1X1Dx
X+SDJ1zD5sIc+I+sRLM4DQQhbVIXOX/vzcr4A7xs4cqj1vw02swpbz1ekCzlWfx/y8NWHYDDWU4uch2GKsW1G2Oyj3AXyN0GwhSk+rB1NsOVpe
rEFiGjICuePbmhcoFbYit1aPExjKbK8qfhpAxK0qaDXUoMZGakikMOPH5IZbsSanZr+ABAyko2qJHV3fiuYjOtZ4ljtmTKTCxxvHPFX9/HnpF6
nKM1XMOCtDHvk9p+fn5bWHbxcOBhg2P5XzF+e05ntRMfDu+JovGX
|   256 ce:76:84:8e:91:ab:9d:84:ba:99:dd:3c:8a:7f:b3:c7 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBLUkYxJu+pYrZfD87IgTvSxQ3z6W0nLsoJ1S
rSOO9eQk9fmQOKIkbv89Hr9G+XuZV5J4BTdFGxb9ZTYDSV1VwFs=
|   256 d6:d9:ca:ce:37:e2:71:a9:0f:41:34:7d:c3:37:60:8f (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIDePXrxwFGl7Bh1OIG3u8U7k9+QWpYKBJMbuH8ltoQmb
5001/tcp  closed commplex-link   reset ttl 64
5201/tcp  closed targus-getdata1 reset ttl 64
6363/tcp  closed ndn             reset ttl 64
47765/tcp closed unknown         reset ttl 64
47766/tcp closed unknown         reset ttl 64
47767/tcp closed unknown         reset ttl 64
OS fingerprint not ideal because: Didn't receive UDP response. Please try again with -sSU
Aggressive OS guesses: OpenWrt Kamikaze 7.09 (Linux 2.6.22) (93%), Linux 2.6.18 - 2.6.22 (92%), OpenWrt 0.9 -
7.09 (Linux 2.4.30 - 2.4.34) (92%), OpenWrt White Russian 0.9 (Linux 2.4.30) (92%), Crestron XPanel control sy
stem (92%), HP PSC 2400-series Photosmart printer (91%), Vodavi XTS-IP PBX (90%), Netgear WGR614v7 wireless br
oadband router (88%), Linux 2.4.26 (Slackware 10.0.0) (87%), Nintendo Wii game console (87%)
No exact OS matches for host (test conditions non-ideal).
TCP/IP fingerprint:
SCAN(V=7.80%E=4%D=11/16%OT=22%CT=5001%CU=%PV=Y%G=N%TM=5FB2E20B%P=x86_64-pc-linux-gnu)
SEQ(SP=100%GCD=2%ISR=106%TI=Z%CI=RI%II=I%TS=U)
SEQ(SP=104%GCD=1%ISR=107%TI=Z%CI=RI%TS=U)
OPS(O1=M5B4NNSNW7%O2=M578NNSNW7%O3=M280NW7%O4=M22CNNSNW7%O5=M218NNSNW7%O6=M109NNS)
WIN(W1=7210%W2=7210%W3=7210%W4=7210%W5=7210%W6=7210)
ECN(R=N)
T1(R=Y%DF=Y%TG=40%S=O%A=S+%F=AS%RD=0%Q=)
T2(R=N)
T3(R=N)
T4(R=Y%DF=Y%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T5(R=Y%DF=Y%TG=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
T6(R=Y%DF=Y%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T7(R=N)
U1(R=N)
IE(R=Y%DFI=N%TG=40%CD=S)

TCP Sequence Prediction: Difficulty=260 (Good luck!)
IP ID Sequence Generation: All zeros

TRACEROUTE
HOP RTT      ADDRESS
1   85.52 ms 192.168.168.17
```

Figure 7-7: TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts



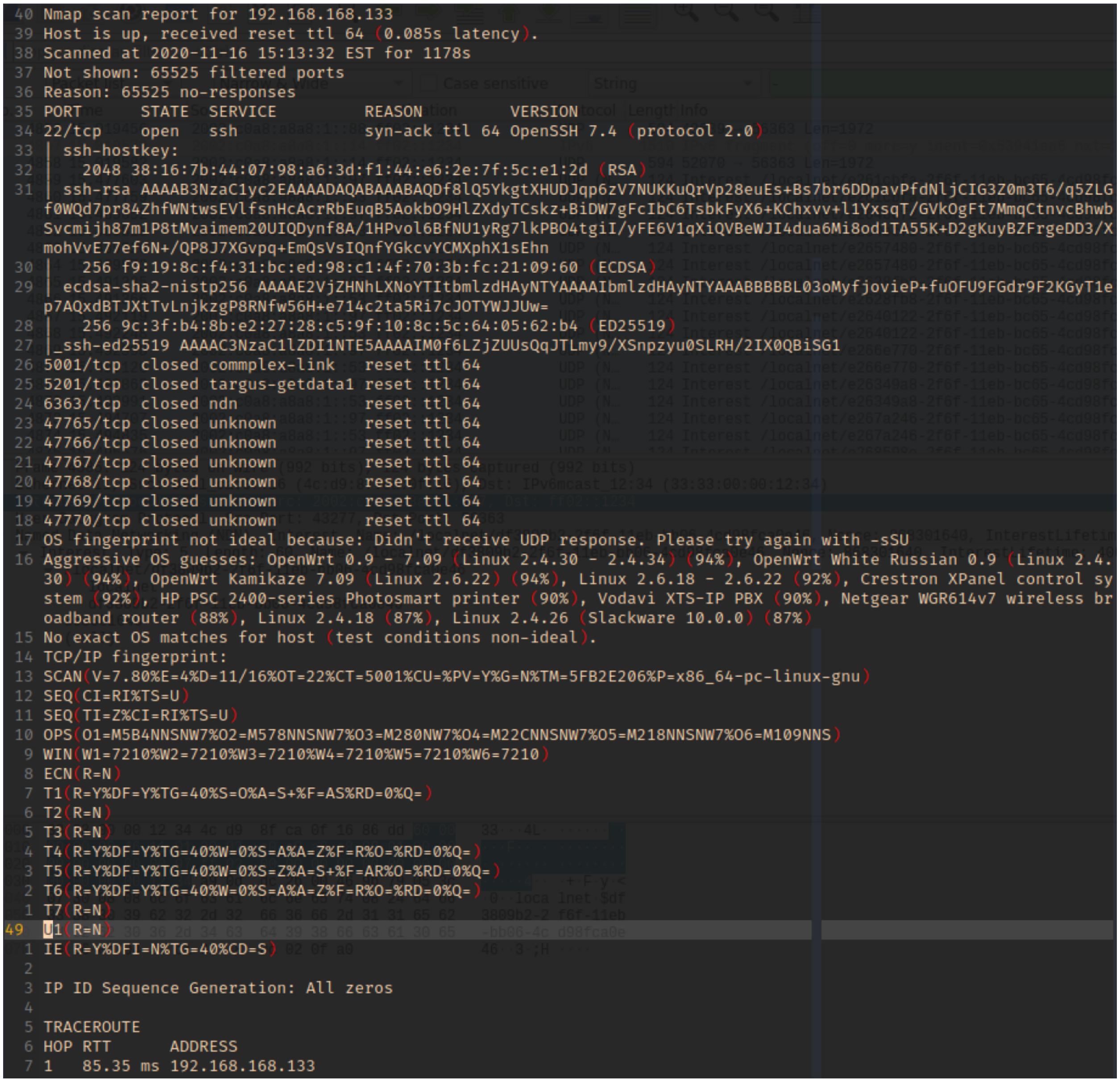
```
Nmap scan report for 192.168.168.17
Host is up, received reset ttl 64 (0.058s latency).
All 65535 scanned ports on 192.168.168.17 are unfiltered because of 65535 resets

Read data files from: /usr/bin/../share/nmap
```

Figure 7-8: TCP ACK scans show all ports as unfiltered, hinting that the firewall/IDS is likely stateless

operant-ndn-ids-sensor2

- IP Address: 192.168.168.133
- Operating System: GNU Linux (Likely OpenWrt White Russian 0.9, OpenWrt Kamikaze)
- Architecture: Intel x86_64

- Open Ports: 22/TCP (OpenSSH Server v7.4)

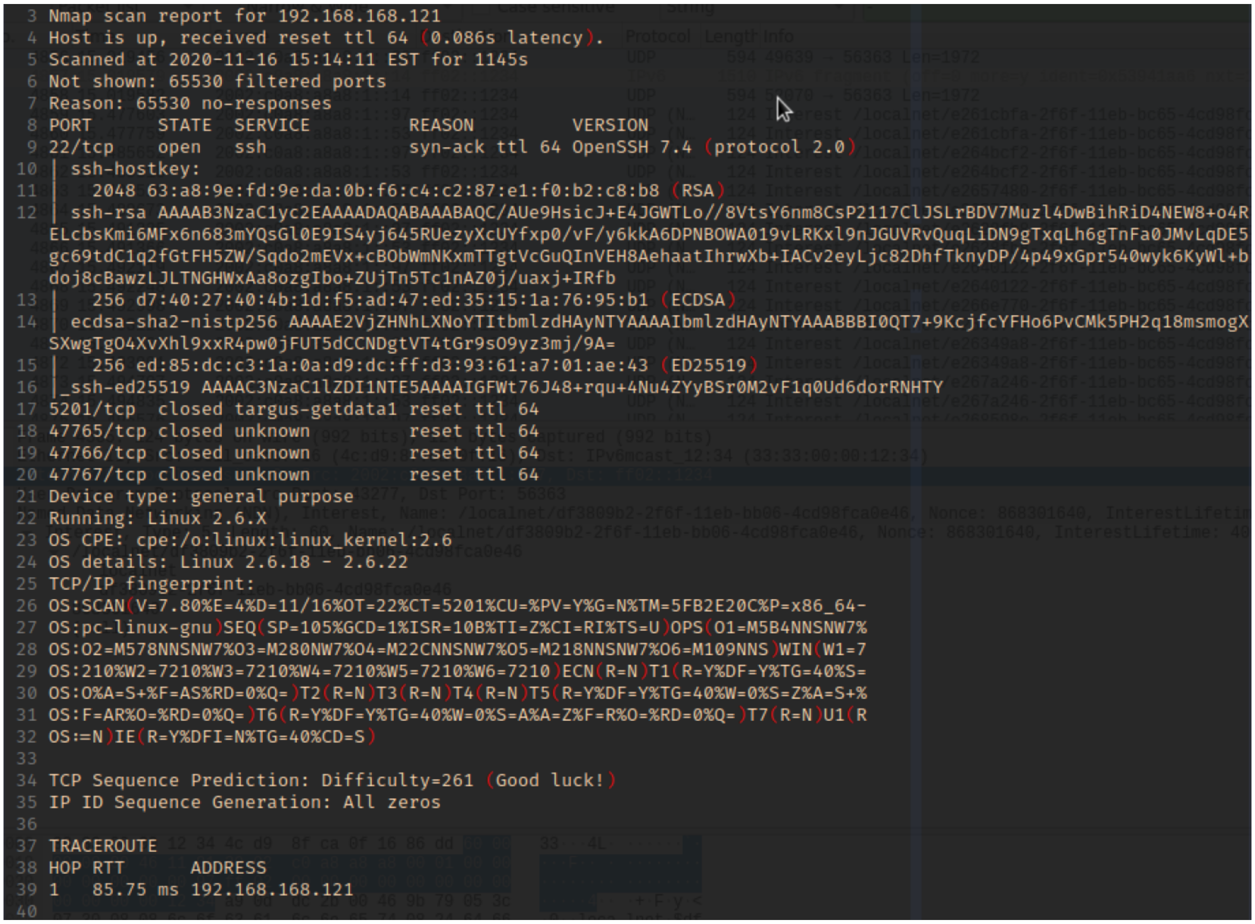


```
Nmap scan report for 192.168.168.133
Host is up, received reset ttl 64 (0.085s latency).
Scanned at 2020-11-16 15:13:32 EST for 1178s
Not shown: 65525 filtered ports
Reason: 65525 no-responses
PORT      STATE  SERVICE          REASON       VERSION
22/tcp    open   ssh              syn-ack ttl 64 OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 38:16:7b:44:07:98:a6:3d:f1:44:e6:2e:7f:5c:e1:2d (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDf8lQ5YkgtXHUDJqp6zV7NUKKuQrVp28euEs+Bs7br6DDpavPfdNljCIG3Z0m3T6/q5ZLG
f0WQd7pr84ZhfWNtwsEVlfsnHcAcrRbEuqB5AokbO9HlZXdyTCskz+BiDW7gFcIbC6TsbkFyX6+KChxvv+l1YxsqT/GVkOgFt7MmqCtnvcBhwb
Svcmijh87m1P8tMvaimem20UIQDynf8A/1HPvol6BfNU1yRg7lkPBO4tgiI/yFE6V1qXiQVBeWJI4dua6Mi8od1TA55K+D2gKuyBZFrgeDD3/X
mohVvE77ef6N+/QP8J7XGvpq+EmQsVsIQnfYGkcvYCMXphX1sEhn
|   256 f9:19:8c:f4:31:bc:ed:98:c1:4f:70:3b:fc:21:09:60 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBBBL03oMyfjovieP+fuOFU9FGdr9F2KGyT1e
p7AQxdBTJXtTvLnjkzgP8RNfw56H+e714c2ta5Ri7cJOTYWJJUw=
|   256 9c:3f:b4:8b:e2:27:28:c5:9f:10:8c:5c:64:05:62:b4 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIM0f6LZjZUUsQqJTLmy9/XSnpzyu0SLRH/2IX0QBiSG1
5001/tcp  closed commplex-link    reset ttl 64
5201/tcp  closed targus-getdata1  reset ttl 64
6363/tcp  closed ndn              reset ttl 64
47765/tcp closed unknown          reset ttl 64
47766/tcp closed unknown          reset ttl 64
47767/tcp closed unknown          reset ttl 64
47768/tcp closed unknown          reset ttl 64
47769/tcp closed unknown          reset ttl 64
47770/tcp closed unknown          reset ttl 64
OS fingerprint not ideal because: Didn't receive UDP response. Please try again with -sSU
Aggressive OS guesses: OpenWrt 0.9 - 7.09 (Linux 2.4.30 - 2.4.34) (94%), OpenWrt White Russian 0.9 (Linux 2.4.
30) (94%), OpenWrt Kamikaze 7.09 (Linux 2.6.22) (94%), Linux 2.6.18 - 2.6.22 (92%), Crestron XPanel control sy
stem (92%), HP PSC 2400-series Photosmart printer (90%), Vodavi XTS-IP PBX (90%), Netgear WGR614v7 wireless br
oadband router (88%), Linux 2.4.18 (87%), Linux 2.4.26 (Slackware 10.0.0) (87%)
No exact OS matches for host (test conditions non-ideal).
TCP/IP fingerprint:
SCAN(V=7.80%E=4%D=11/16%OT=22%CT=5001%CU=%PV=Y%G=N%TM=5FB2E206%P=x86_64-pc-linux-gnu)
SEQ(CI=RI%TS=U)
SEQ(TI=Z%CI=RI%TS=U)
OPS(O1=M5B4NNSNW7%O2=M578NNSNW7%O3=M280NW7%O4=M22CNNSNW7%O5=M218NNSNW7%O6=M109NNS)
WIN(W1=7210%W2=7210%W3=7210%W4=7210%W5=7210%W6=7210)
ECN(R=N)
T1(R=Y%DF=Y%TG=40%S=O%A=S+%F=AS%RD=0%Q=)
T2(R=N)
T3(R=N)
T4(R=Y%DF=Y%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T5(R=Y%DF=Y%TG=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
T6(R=Y%DF=Y%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T7(R=N)
U1(R=N)
IE(R=Y%DFI=N%TG=40%CD=S)

IP ID Sequence Generation: All zeros

TRACEROUTE
HOP RTT      ADDRESS
1   85.35 ms 192.168.168.133
```

Figure 7-9: Full TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts

operant-ndn-ids-sensor3

- IP Address: 192.168.168.121
- Operating System: GNU Linux (Likely OpenWrt White Russian 0.9, OpenWrt Kamikaze)
- Architecture: Intel x86_64
- Open Ports: 22/TCP (OpenSSH Server v7.4)



```
Nmap scan report for 192.168.168.121
Host is up, received reset ttl 64 (0.086s latency).
Scanned at 2020-11-16 15:14:11 EST for 1145s
Not shown: 65530 filtered ports
Reason: 65530 no-responses
PORT      STATE  SERVICE          REASON         VERSION
22/tcp    open   ssh              syn-ack ttl 64 OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 63:a8:9e:fd:9e:da:0b:f6:c4:c2:87:e1:f0:b2:c8:b8 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC/AUe9HsicJ+E4JGWTLo//8VtsY6nm8CsP2117ClJSLrBDV7Muzl4DwBihRiD4NEW8+o4R
ELcbsKmi6MFx6n683mYQsGl0E9IS4yj645RUezyXcUYfxp0/vF/y6kkA6DPNBOWA019vLRKxl9nJGUVRvQuqLiDN9gTxqLh6gTnFa0JMvLqDE5
gc69tdC1q2fGtFH5ZW/Sqdo2mEVx+cBObWmNKxmTTgtVcGuQInVEH8AehaatIhrwXb+IACv2eyLjc82DhfTknyDP/4p49xGpr540wyk6KyWl+b
doj8cBawzLJLTNGHWxAa8ozgIbo+ZUjTn5Tc1nAZ0j/uaxj+IRfb
|   256 d7:40:27:40:4b:1d:f5:ad:47:ed:35:15:1a:76:95:b1 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBI0QT7+9KcjfcYFHo6PvCMk5PH2q18msmogX
SXwgTgO4XvXhl9xxR4pw0jFUT5dCCNDgtVT4tGr9sO9yz3mj/9A=
|   256 ed:85:dc:c3:1a:0a:d9:dc:ff:d3:93:61:a7:01:ae:43 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIGFWt76J48+rqu+4Nu4ZYyBSr0M2vF1q0Ud6dorRNHTY
5201/tcp  closed targus-getdata1 reset ttl 64
47765/tcp closed unknown          reset ttl 64
47766/tcp closed unknown          reset ttl 64
47767/tcp closed unknown          reset ttl 64
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.18 - 2.6.22
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=11/16%OT=22%CT=5201%CU=%PV=Y%G=N%TM=5FB2E20C%P=x86_64-
OS:pc-linux-gnu)SEQ(SP=105%GCD=1%ISR=10B%TI=Z%CI=RI%TS=U)OPS(O1=M5B4NNSNW7%
OS:O2=M578NNSNW7%O3=M280NW7%O4=M22CNNSNW7%O5=M218NNSNW7%O6=M109NNS)WIN(W1=7
OS:210%W2=7210%W3=7210%W4=7210%W5=7210%W6=7210)ECN(R=N)T1(R=Y%DF=Y%TG=40%S=
OS:O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=N)T5(R=Y%DF=Y%TG=40%W=0%S=Z%A=S+%
OS:F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=N)U1(R
OS:=N)IE(R=Y%DFI=N%TG=40%CD=S)

TCP Sequence Prediction: Difficulty=261 (Good luck!)
IP ID Sequence Generation: All zeros

TRACEROUTE
HOP RTT      ADDRESS
1   85.75 ms 192.168.168.121
```

Figure 7-10: Full TCP Stealth scan of all ports with service enumeration, OS fingerprinting, and NSE scripts

The pen-test team also utilized OpenVAS to scan the targets for potential vulnerabilities. The results of these scans revealed a few medium risk vulnerabilities. Recommended mitigations are also provided.

Table 7-2: OpenVAS Vulnerabilities

Vulnerability	Severity	Hosts	Location	Impact	Mitigation
Telnet Unencrypted Telnet Login	Medium (4.8)	192.168.168.1	23/tcp	An attacker can uncover login names and passwords by sniffing plaintext telnet traffic.	Replace Telnet with SSH for encrypted communications.
Cleartext Transmission of Sensitive Information via HTTP	Medium (4.8)	192.168.168.110	9200/tcp	An attacker can eavesdrop on plaintext communications to gain access to sensitive information via a MITM attack.	Enforce transmission of sensitive data via encrypted SSL/TLS connection.
Default Cisco Credentials	Medium (4.4)	192.168.168.1	23/tcp	Remote Cisco router has default credentials set - allowing an attacker to gather a lot of information about the network and possibly shut it down.	Change the default **cisco:cisco** credentials.

Vulnerability	Severity	Hosts	Location	Impact	Mitigation
SSH Weak Encryption Algorithm Supported	Medium (4.4)	192.168.168.110, 192.168.168.17, 192.168.168.121, 192.168.168.133	22/tcp	An attacker may recover sensitive plaintext information due to weak encryption algorithms.	Disable weak encryption ciphers including RC4 and CBC cipher variants.

A.1.1.3. NDN Architecture and Sniffing

Operant has implemented an industrialized version of NDN protocol for securing communication of smart grid devices. To do this, they have introduced an overlay network using an IDS and a copy of the architecture was received. The concern for us here is that some of the attack scenarios may not hold with the architecture that has been implemented. To better understand the traffic in the NDN network, Wireshark[10] – a network sniffer – was used to capture data on the NDN nodes. The initial packets captured October 2020 showed interest and data packets. Subsequent packets captured in November 2020 no longer show these packets.

Analysis of the pcaps captured on October show

: interest and data packets as IPv6 UDP to port 56363 (default NFD port is 6363) (Recommendation: It is a good practice to not use default ports as it scales down opportunistic attacks)

: Packet name generic name values all include localnet <ids-prefix>
: Gateway is 192.168.168.168
: Noisy multicast listener reports, and more
: TLSv1.2 encrypted http-over-tls application data
: 192-network data and Sonicwall ARPs protocol are observed

NDN Architecture diagrams received in November show

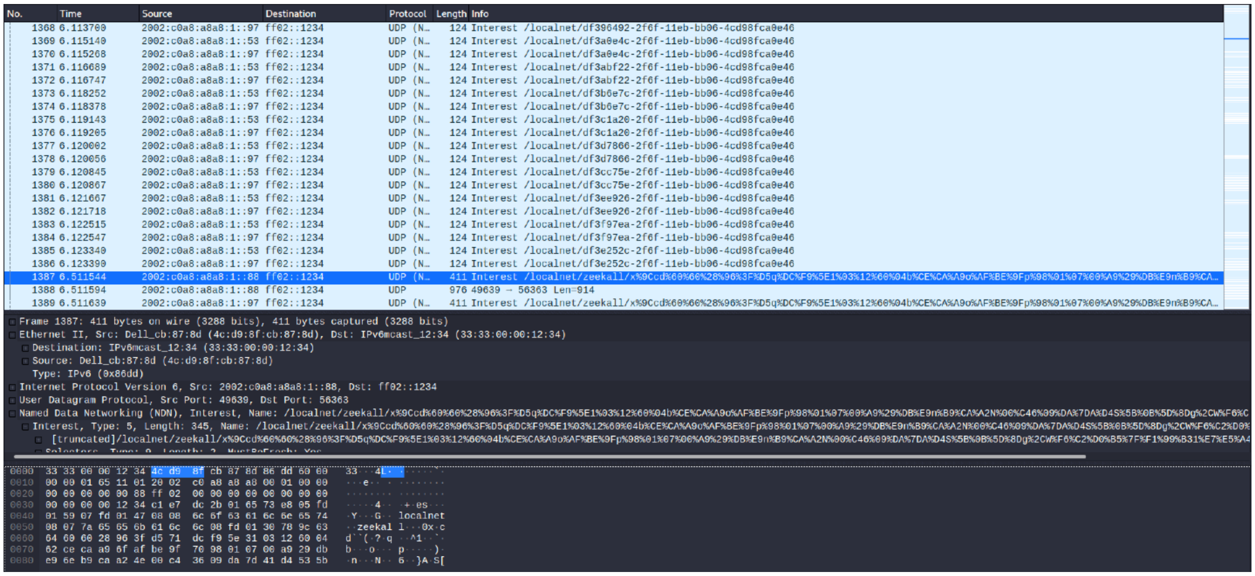
- Network taps for IDS sensor nodes attached to analysis nodes, all using Zeek[11] which is recommended
- IDS sensor and analysis nodes are an overlay to the physical nodes which poses a new/different attack surface
- Server node traffic includes multicast listener reports, HTTP GETs, Sonicwall ARPs, TLS1.2 encrypted http-over-TLS-application data were seen
- Ids-sensor1 includes membership reports, Sonicwall ARPs, other noisy traffic was observed
- Ids-sensor1a includes multicast listener, membership reports, Sonicwall ARPs, other noisy traffic were observed
- Sensor2 traffic includes complete TLS1.2 Handshake (Recommendation: Were TLS1.3 protocol used, TLS1.3 would pass most of this information encrypted)
- Multicast Listener Report Messages are from the (previously-)UDP listeners?
- TLS appears to be using chained, Red Hat certificates. It was not immediately clear if there was an internal or external certificate authority which needs further verification

[10] https://www.wireshark.org/
[11] https://zeek.org/

7.2.2 Interruption

Traditional interruption attacks were performed by the pen-test team in the form of ICMP and SYN flood DoS/DDoS attacks. The pen-test team used hping3 and the auxiliary synflood module from Metasploit to perform the attacks. Both attacks generated fake data from randomized sources and attempted to flood the targets with traffic to render the system unusable by legitimate users and services. The endpoints were monitored using tcpdump and ping probes to determine status and latency of the target. These tests revealed that none of the nodes were susceptible to these forms of attack on a small scale (1-2 attack machines running 64-bit processors). There wasn't a noticeable or significant introduction of latency to the environment and the endpoints never completely crashed, however; the pen-test team was unable to generate a larger scale attack because the VPN connection kept terminating.

A DDoS attack was also performed within the environment by generating a large number of interest packets for non-existent data. This attack was performed to attempt to flood the pending interest table of the router with forwarding requests that would cause NDN communications to be overwhelmed and deny legitimate requests. The pen-test team created a python script that would generate a large number of interest packets requesting data for random UUID's that do not exist. The program was executed on sensors 1 and 3 while another node requested legitimate content for <ids-prefix> in use. The outcome of this attack demonstrated that the legitimate content was served to the endpoint with no impact from the bogus requests. It is believed that there are a couple of factors that led to this result. There is the possibility that there was not a high enough volume of interest requests and not enough nodes participating in the attack. Similarly, a short interest lifetime and request timeouts could impact the outcome of the attack.



```
No.  Time     Source                Destination   Protocol Length Info
1368 6.113700 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df396492-2f6f-11eb-bb06-4cd98fca0e46
1369 6.115140 2002:c0a8:a8a8:1::53 ff02::1234    UDP (N…   124 Interest /localnet/df3a0e4c-2f6f-11eb-bb06-4cd98fca0e46
1370 6.115268 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df3a0e4c-2f6f-11eb-bb06-4cd98fca0e46
1371 6.116689 2002:c0a8:a8a8:1::53 ff02::1234    UDP (N…   124 Interest /localnet/df3abf22-2f6f-11eb-bb06-4cd98fca0e46
1372 6.116747 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df3abf22-2f6f-11eb-bb06-4cd98fca0e46
1373 6.118252 2002:c0a8:a8a8:1::53 ff02::1234    UDP (N…   124 Interest /localnet/df3b6e7c-2f6f-11eb-bb06-4cd98fca0e46
1374 6.118378 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df3b6e7c-2f6f-11eb-bb06-4cd98fca0e46
1375 6.119143 2002:c0a8:a8a8:1::53 ff02::1234    UDP (N…   124 Interest /localnet/df3c1a20-2f6f-11eb-bb06-4cd98fca0e46
1376 6.119205 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df3c1a20-2f6f-11eb-bb06-4cd98fca0e46
1377 6.120002 2002:c0a8:a8a8:1::53 ff02::1234    UDP (N…   124 Interest /localnet/df3d7866-2f6f-11eb-bb06-4cd98fca0e46
1378 6.120056 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df3d7866-2f6f-11eb-bb06-4cd98fca0e46
1379 6.120845 2002:c0a8:a8a8:1::53 ff02::1234    UDP (N…   124 Interest /localnet/df3cc75e-2f6f-11eb-bb06-4cd98fca0e46
1380 6.120867 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df3cc75e-2f6f-11eb-bb06-4cd98fca0e46
1381 6.121667 2002:c0a8:a8a8:1::53 ff02::1234    UDP (N…   124 Interest /localnet/df3ee926-2f6f-11eb-bb06-4cd98fca0e46
1382 6.121718 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df3ee926-2f6f-11eb-bb06-4cd98fca0e46
1383 6.122515 2002:c0a8:a8a8:1::53 ff02::1234    UDP (N…   124 Interest /localnet/df3f97ea-2f6f-11eb-bb06-4cd98fca0e46
1384 6.122547 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df3f97ea-2f6f-11eb-bb06-4cd98fca0e46
1385 6.123340 2002:c0a8:a8a8:1::53 ff02::1234    UDP (N…   124 Interest /localnet/df3e252c-2f6f-11eb-bb06-4cd98fca0e46
1386 6.123390 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   124 Interest /localnet/df3e252c-2f6f-11eb-bb06-4cd98fca0e46
1387 6.511544 2002:c0a8:a8a8:1::88 ff02::1234    UDP (N…   411 Interest /localnet/zeekall/x%9Ccd%60%60%28%96%3F%D5q%DC%F9%5E1%03%12%60%04b%CE%CA%A9o%AF%BE%9Fp%98%01%07%00%A9%29%DB%E9n%B9%CA…
1388 6.511594 2002:c0a8:a8a8:1::88 ff02::1234    UDP       976 49639 → 56363 Len=914
1389 6.511639 2002:c0a8:a8a8:1::97 ff02::1234    UDP (N…   411 Interest /localnet/zeekall/x%9Ccd%60%60%28%96%3F%D5q%DC%F9%5E1%03%12%60%04b%CE%CA%A9o%AF%BE%9Fp%98%01%07%00%A9%29%DB%E9n%B9%CA…

Frame 1387: 411 bytes on wire (3288 bits), 411 bytes captured (3288 bits)
Ethernet II, Src: Dell_cb:87:8d (4c:d9:8f:cb:87:8d), Dst: IPv6mcast_12:34 (33:33:00:00:12:34)
  Destination: IPv6mcast_12:34 (33:33:00:00:12:34)
  Source: Dell_cb:87:8d (4c:d9:8f:cb:87:8d)
  Type: IPv6 (0x86dd)
Internet Protocol Version 6, Src: 2002:c0a8:a8a8:1::88, Dst: ff02::1234
User Datagram Protocol, Src Port: 49639, Dst Port: 56363
Named Data Networking (NDN), Interest, Name: /localnet/zeekall/x%9Ccd%60%60%28%96%3F%D5q%DC%F9%5E1%03%12%60%04b%CE%CA%A9o%AF%BE%9Fp%98%01%07%00%A9%29%DB%E9n%B9%CA%A2N%00%C46%09%DA%7DA%D4S%5B%0B%5D%8Dg%2CW%F6%C
  Interest, Type: 5, Length: 345, Name: /localnet/zeekall/x%9Ccd%60%60%28%96%3F%D5q%DC%F9%5E1%03%12%60%04b%CE%CA%A9o%AF%BE%9Fp%98%01%07%00%A9%29%DB%E9n%B9%CA%A2N%00%C46%09%DA%7DA%D4S%5B%0B%5D%8Dg%2CW%F6%C2%D0%
    [truncated]/localnet/zeekall/x%9Ccd%60%60%28%96%3F%D5q%DC%F9%5E1%03%12%60%04b%CE%CA%A9o%AF%BE%9Fp%98%01%07%00%A9%29%DB%E9n%B9%CA%A2N%00%C46%09%DA%7DA%D4S%5B%0B%5D%8Dg%2CW%F6%C2%D0%B5%7F%F1%99%B31%E7%E5%A4
    Selectors, Type: 9, Length: 2, MustBeFresh: Yes

0000  33 33 00 00 12 34 4c d9  8f cb 87 8d 86 dd 60 00   33···4L· ······`·
0010  00 00 01 65 11 01 20 02  c0 a8 a8 a8 00 01 00 00   ···e·· · ········
0020  00 00 00 00 00 88 ff 02  00 00 00 00 00 00 00 00   ········ ········
0030  00 00 00 00 12 34 c1 e7  dc 2b 01 65 73 e8 05 fd   ·····4·· ·+·es···
0040  01 59 07 fd 01 47 08 08  6c 6f 63 61 6c 6e 65 74   ·Y···G·· localnet
0050  08 07 7a 65 65 6b 61 6c  6c 08 fd 01 30 78 9c 63   ··zeekal l···0x·c
0060  64 60 60 28 96 3f d5 71  dc f9 5e 31 03 12 60 04   d``(·?·q ··^1··`·
0070  62 ce ca a9 6f af be 9f  70 98 01 07 00 a9 29 db   b···o··· p·····)·
0080  e9 6e b9 ca a2 4e 00 c4  36 09 da 7d 41 d4 53 5b   ·n···N·· 6··}A·S[
```

Figure 7-11: Unsuccessful attempt at DDoS attack by flooding PIT with random content requests

Another DDoS attack was performed by the pen-test team in the form of a reflection attack. This attack was similar to the previous attack except that source IP addresses for all the nodes were spoofed to cause packets to be redirected to a victim. The intention of this attack it to flood the victim with bogus requests to interrupt communications and deny legitimate requests. To perform the reflection attack, the pen-test team used arpspoof on participating machines to redirect the source IP of the traffic to the victim's IP. It was unclear if this attack was successful since the victim was still able to receive legitimate traffic for <ids-prefix> in use and there wasn't a good way to validate results.

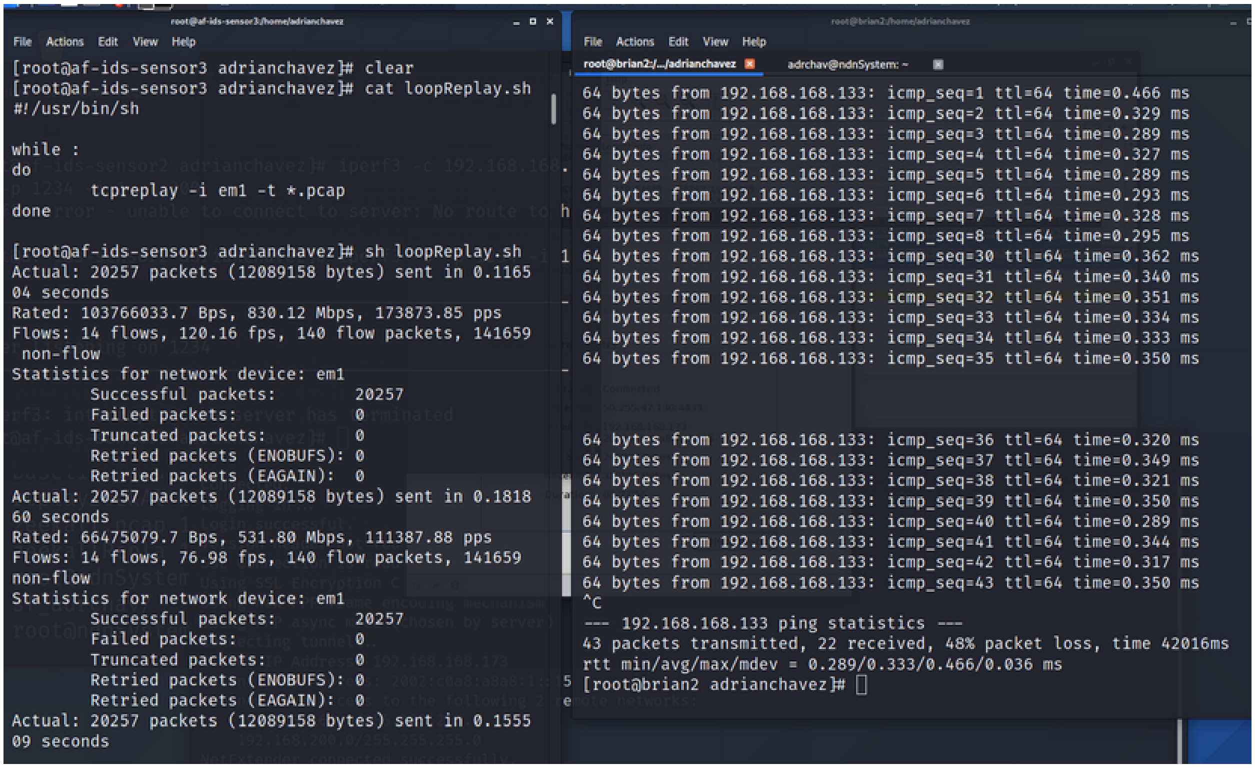
7.2.3 Packet Replay

A replay attack seeks to fraudulently replay previously eavesdropped data to, for instance, trick the recipient(s) into believing that the packets are legitimate or cause some interference or congestion in network traffic. To better understand the success of this attack (were an pen-test team able to compromise a node (brian3) and capture traffic on the network interface) a node was setup and access was given with the intention of playing back the captured data on the sensors. This attack was performed in two stages. First the interest packet using the "test-echo-consumer' code was generated on the node. The '/localnet/<ids-prefix>' topic is the only topic that returns a data packet (to the team's best knowledge from testing out other topics like '/test/hello123'). Tcpdump was used to capture this traffic which was replayed on **operant-ndn-ids**-sensor3 using Tcpreplay. However, on **operant-ndn-ids**-sensor3, data packets for the '/localnet/<ids-prefix>' topic is continuously being generated with no unique identifying information to determine the validity of the responses from the replayed interest packet.

When asked what success will look like, Operant had responded that a hex encoded ID that appears in both the interest and data packets was the unique identifier. However, the team continued to see the same hex encoded identifier in interest and data packets even before carrying out the replay attack.

To further determine the impact of this attack, the team decided to continuously replay the packets in a loop while measuring the network connectivity using pings between the node and sensor3 before, during, and after the replay. This is also known as the loop repeated replay attack.

The pings are running normally before the replay attack loop is started on **operant-ndn-ids**-sensor3. As soon as the replay attack starts on sensor3, icmp ping packets are dropped between brian2 and **operant-ndn-ids**-sensor3. As soon as the tcpreplay loop is stopped, the pings resume. Below is a screenshot of the tests. The left terminal shows the sensor3 system running the loop repeated replay attack where several previously captured pcap's are replayed. The right terminal shows the icmp pings that are interrupted by the tcp replay attack between icmp_seq=8 and icmp_seq=30. So, it appears that the tcpreplay has a negative impact on network performance.


root@af-ids-sensor3:/home/adrianchavez
File Actions Edit View Help
root@brian2:/home/adrianchavez
File Actions Edit View Help
root@brian2:/.../adrianchavez
adrchav@ndnSystem: ~

```
[root@af-ids-sensor3 adrianchavez]# clear
[root@af-ids-sensor3 adrianchavez]# cat loopReplay.sh
#!/usr/bin/sh

while :
do
        tcpreplay -i em1 -t *.pcap
done

[root@af-ids-sensor3 adrianchavez]# sh loopReplay.sh
Actual: 20257 packets (12089158 bytes) sent in 0.1165
04 seconds
Rated: 103766033.7 Bps, 830.12 Mbps, 173873.85 pps
Flows: 14 flows, 120.16 fps, 140 flow packets, 141659
 non-flow
Statistics for network device: em1
        Successful packets:        20257
        Failed packets:            0
        Truncated packets:         0
        Retried packets (ENOBUFS): 0
        Retried packets (EAGAIN):  0
Actual: 20257 packets (12089158 bytes) sent in 0.1818
60 seconds
Rated: 66475079.7 Bps, 531.80 Mbps, 111387.88 pps
Flows: 14 flows, 76.98 fps, 140 flow packets, 141659
non-flow
Statistics for network device: em1
        Successful packets:        20257
        Failed packets:            0
        Truncated packets:         0
        Retried packets (ENOBUFS): 0
        Retried packets (EAGAIN):  0
Actual: 20257 packets (12089158 bytes) sent in 0.1555
09 seconds
```

```
64 bytes from 192.168.168.133: icmp_seq=1 ttl=64 time=0.466 ms
64 bytes from 192.168.168.133: icmp_seq=2 ttl=64 time=0.329 ms
64 bytes from 192.168.168.133: icmp_seq=3 ttl=64 time=0.289 ms
64 bytes from 192.168.168.133: icmp_seq=4 ttl=64 time=0.327 ms
64 bytes from 192.168.168.133: icmp_seq=5 ttl=64 time=0.289 ms
64 bytes from 192.168.168.133: icmp_seq=6 ttl=64 time=0.293 ms
64 bytes from 192.168.168.133: icmp_seq=7 ttl=64 time=0.328 ms
64 bytes from 192.168.168.133: icmp_seq=8 ttl=64 time=0.295 ms
64 bytes from 192.168.168.133: icmp_seq=30 ttl=64 time=0.362 ms
64 bytes from 192.168.168.133: icmp_seq=31 ttl=64 time=0.340 ms
64 bytes from 192.168.168.133: icmp_seq=32 ttl=64 time=0.351 ms
64 bytes from 192.168.168.133: icmp_seq=33 ttl=64 time=0.334 ms
64 bytes from 192.168.168.133: icmp_seq=34 ttl=64 time=0.333 ms
64 bytes from 192.168.168.133: icmp_seq=35 ttl=64 time=0.350 ms



64 bytes from 192.168.168.133: icmp_seq=36 ttl=64 time=0.320 ms
64 bytes from 192.168.168.133: icmp_seq=37 ttl=64 time=0.349 ms
64 bytes from 192.168.168.133: icmp_seq=38 ttl=64 time=0.321 ms
64 bytes from 192.168.168.133: icmp_seq=39 ttl=64 time=0.350 ms
64 bytes from 192.168.168.133: icmp_seq=40 ttl=64 time=0.289 ms
64 bytes from 192.168.168.133: icmp_seq=41 ttl=64 time=0.344 ms
64 bytes from 192.168.168.133: icmp_seq=42 ttl=64 time=0.317 ms
64 bytes from 192.168.168.133: icmp_seq=43 ttl=64 time=0.350 ms
^C
--- 192.168.168.133 ping statistics ---
43 packets transmitted, 22 received, 48% packet loss, time 42016ms
rtt min/avg/max/mdev = 0.289/0.333/0.466/0.036 ms
[root@brian2 adrianchavez]# 
```

Figure 7-12: Loop repeated replay attack

7.2.4 Using Components with Known Vulnerabilities

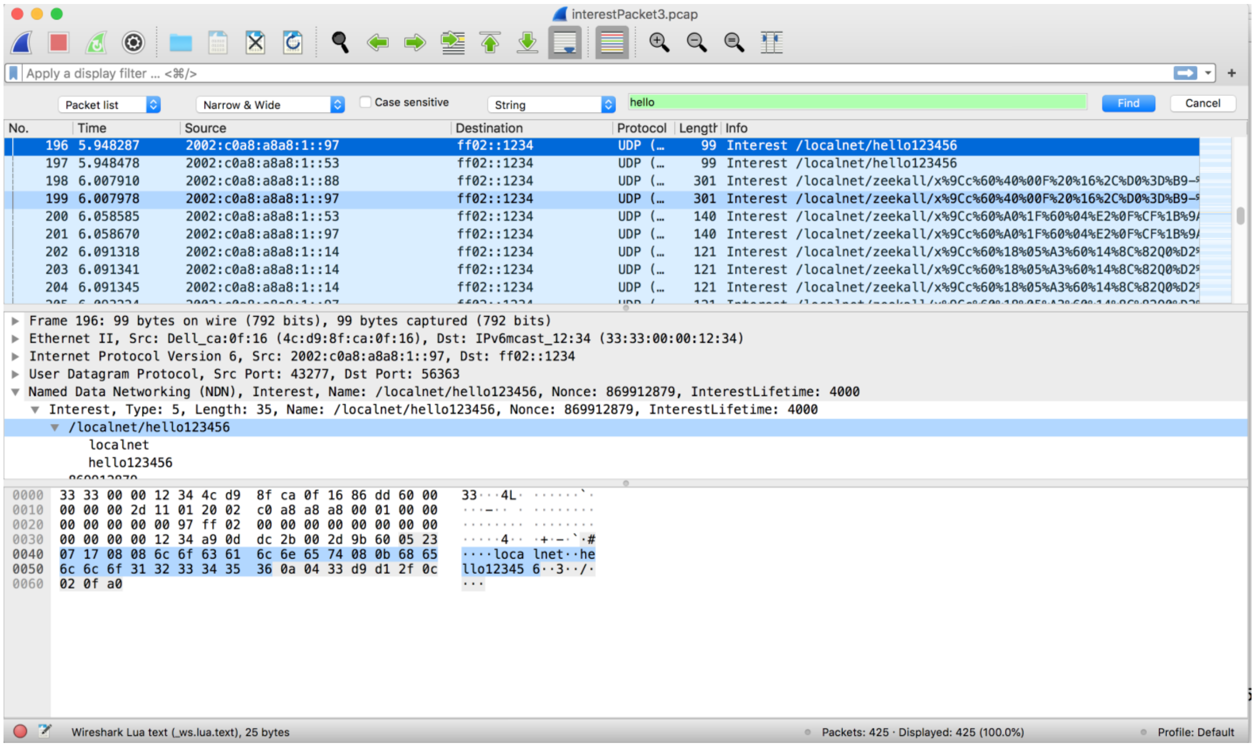
See Section 6.1.3

7.2.5 Content Poisoning

Content Poisoning was launched but a conclusive result could not be determined given the system downtime experienced while running the MiTM attack.

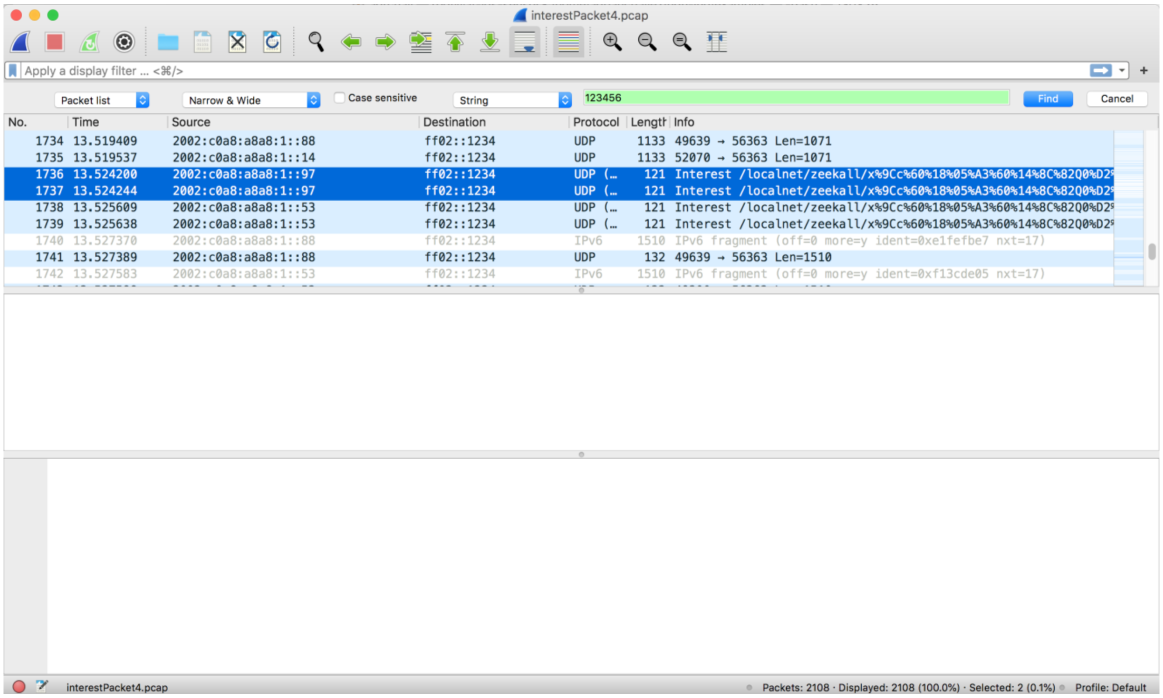
7.2.6 Injection

Modification of information by both untrusted and trusted parties are threats that organizations should be cognizant of. The team attempted to use Kali Linux as an untrusted outsider to inject data into the network failed because Ettercap was not able to intercept the packets from any of the nodes. An insider threat is difficult to effectively combat, and to test this threat, the team decided to install Ettercap on one of the sensor nodes. Given the novel NDN architecture, an NDN node (**operant-ndn-ids**-sensor3) was **assumed** to be compromised, acting as a malicious insider pen-test team, to circumvent information from another NDN node (**operant-ndn-ids**-sensor1) in the network. The interest packets were generated as shown below.



```
No.  Time      Source               Destination  Protocol Length Info
196  5.948287  2002:c0a8:a8a8:1::97  ff02::1234  UDP (…    99  Interest /localnet/hello123456
197  5.948478  2002:c0a8:a8a8:1::53  ff02::1234  UDP (…    99  Interest /localnet/hello123456
198  6.007910  2002:c0a8:a8a8:1::88  ff02::1234  UDP (…   301  Interest /localnet/zeekall/x%9Cc%60%40%00F%20%16%2C%D0%3D%B9–
199  6.007978  2002:c0a8:a8a8:1::97  ff02::1234  UDP (…   301  Interest /localnet/zeekall/x%9Cc%60%40%00F%20%16%2C%D0%3D%B9–
200  6.058585  2002:c0a8:a8a8:1::53  ff02::1234  UDP (…   140  Interest /localnet/zeekall/x%9Cc%60%A0%1F%60%04%E2%0F%CF%1B%9/
201  6.058670  2002:c0a8:a8a8:1::97  ff02::1234  UDP (…   140  Interest /localnet/zeekall/x%9Cc%60%A0%1F%60%04%E2%0F%CF%1B%9/
202  6.091318  2002:c0a8:a8a8:1::14  ff02::1234  UDP (…   121  Interest /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%820%D2
203  6.091341  2002:c0a8:a8a8:1::14  ff02::1234  UDP (…   121  Interest /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%820%D2
204  6.091345  2002:c0a8:a8a8:1::14  ff02::1234  UDP (…   121  Interest /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%820%D2

▶ Frame 196: 99 bytes on wire (792 bits), 99 bytes captured (792 bits)
▶ Ethernet II, Src: Dell_ca:0f:16 (4c:d9:8f:ca:0f:16), Dst: IPv6mcast_12:34 (33:33:00:00:12:34)
▶ Internet Protocol Version 6, Src: 2002:c0a8:a8a8:1::97, Dst: ff02::1234
▶ User Datagram Protocol, Src Port: 43277, Dst Port: 56363
▼ Named Data Networking (NDN), Interest, Name: /localnet/hello123456, Nonce: 869912879, InterestLifetime: 4000
  ▼ Interest, Type: 5, Length: 35, Name: /localnet/hello123456, Nonce: 869912879, InterestLifetime: 4000
    ▼ /localnet/hello123456
        localnet
        hello123456

0000  33 33 00 00 12 34 4c d9  8f ca 0f 16 86 dd 60 00   33···4L· ······`·
0010  00 00 00 2d 11 01 20 02  c0 a8 a8 a8 00 01 00 00   ···–·· · ········
0020  00 00 00 00 00 97 ff 02  00 00 00 00 00 00 00 00   ········ ········
0030  00 00 00 00 12 34 a9 0d  dc 2b 00 2d 9b 60 05 23   ·····4·· ·+·–·`·#
0040  07 17 08 08 6c 6f 63 61  6c 6e 65 74 08 0b 68 65   ····loca lnet··he
0050  6c 6c 6f 31 32 33 34 35  36 0a 04 33 d9 d1 2f 0c   llo12345 6··3··/·
0060  02 0f a0                                           ···

Wireshark Lua text (_ws.lua.text), 25 bytes     Packets: 425 · Displayed: 425 (100.0%)     Profile: Default
```

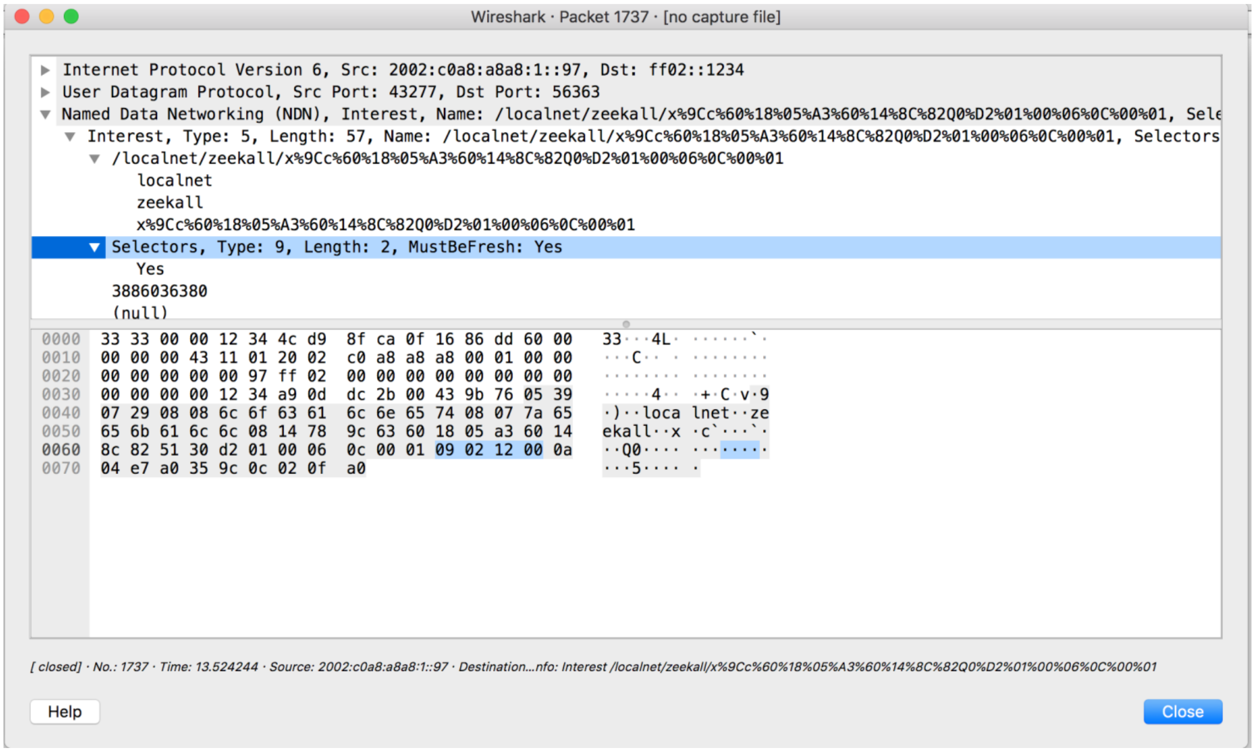
Figure 7-13: Generated data using /localnet/hello123456



```
No.   Time       Source                Destination  Protocol Length Info
1734  13.519409  2002:c0a8:a8a8:1::88  ff02::1234   UDP      1133  49639 → 56363 Len=1071
1735  13.519537  2002:c0a8:a8a8:1::14  ff02::1234   UDP      1133  52070 → 56363 Len=1071
1736  13.524200  2002:c0a8:a8a8:1::97  ff02::1234   UDP (…    121  Interest /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%820%D2
1737  13.524244  2002:c0a8:a8a8:1::97  ff02::1234   UDP (…    121  Interest /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%820%D2
1738  13.525609  2002:c0a8:a8a8:1::53  ff02::1234   UDP (…    121  Interest /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%820%D2
1739  13.525638  2002:c0a8:a8a8:1::53  ff02::1234   UDP (…    121  Interest /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%820%D2
1740  13.527370  2002:c0a8:a8a8:1::88  ff02::1234   IPv6     1510  IPv6 fragment (off=0 more=y ident=0xe1fefbe7 nxt=17)
1741  13.527389  2002:c0a8:a8a8:1::88  ff02::1234   UDP       132  49639 → 56363 Len=1510
1742  13.527583  2002:c0a8:a8a8:1::53  ff02::1234   IPv6     1510  IPv6 fragment (off=0 more=y ident=0xf13cde05 nxt=17)

interestPacket4.pcap     Packets: 2108 · Displayed: 2108 (100.0%) · Selected: 2 (0.1%)     Profile: Default
```

Figure 7-14: Generated interest packet Operant's NDN pub/sub

The data packet received is shown below.



```
▶ Internet Protocol Version 6, Src: 2002:c0a8:a8a8:1::97, Dst: ff02::1234
▶ User Datagram Protocol, Src Port: 43277, Dst Port: 56363
▼ Named Data Networking (NDN), Interest, Name: /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%82Q0%D2%01%00%06%0C%00%01, Sele
  ▼ Interest, Type: 5, Length: 57, Name: /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%82Q0%D2%01%00%06%0C%00%01, Selectors
    ▼ /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%82Q0%D2%01%00%06%0C%00%01
        localnet
        zeekall
        x%9Cc%60%18%05%A3%60%14%8C%82Q0%D2%01%00%06%0C%00%01
    ▼ Selectors, Type: 9, Length: 2, MustBeFresh: Yes
        Yes
      3886036380
      (null)
```

```
0000  33 33 00 00 12 34 4c d9  8f ca 0f 16 86 dd 60 00   33···4L· ······`·
0010  00 00 00 43 11 01 20 02  c0 a8 a8 a8 00 01 00 00   ···C·· · ········
0020  00 00 00 00 00 97 ff 02  00 00 00 00 00 00 00 00   ········ ········
0030  00 00 00 00 12 34 a9 0d  dc 2b 00 43 9b 76 05 39   ·····4·· ·+·C·v·9
0040  07 29 08 08 6c 6f 63 61  6c 6e 65 74 08 07 7a 65   ·)··loca lnet··ze
0050  65 6b 61 6c 6c 08 14 78  9c 63 60 18 05 a3 60 14   ekall··x ·c`···`·
0060  8c 82 51 30 d2 01 00 06  0c 00 01 09 02 12 00 0a   ··Q0···· ········
0070  04 e7 a0 35 9c 0c 02 0f  a0                        ···5···· ·
```

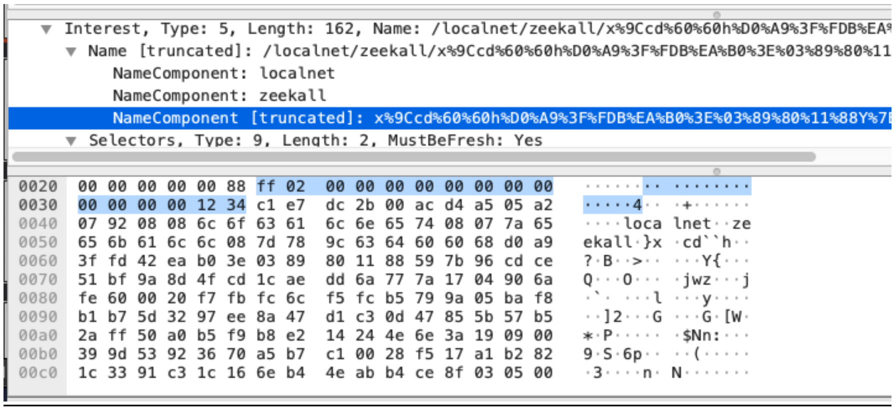
[closed] · No.: 1737 · Time: 13.524244 · Source: 2002:c0a8:a8a8:1::97 · Destination...nfo: Interest /localnet/zeekall/x%9Cc%60%18%05%A3%60%14%8C%82Q0%D2%01%00%06%0C%00%01

Figure 7-15: Data packet returned for the generated interest packet for NDN pub/sub

Ettercap's filter script was used to modify "hello12" to "in use <ids-prefix>". After modifying the interest packet that was intercepted using a MiTM attack, it was undetermined if the attack failed or succeeded. The "hello123456" interest packet normally should not return anything and the "in use <ids-prefix>" interest packet should return a data packet. When running the attack to modify the interest packet the first time, the attack appeared to fail. However, repeating this attack after the first unsuccessful attempt, there was no network connectivity between the **operant-ndn-ids**-sensor1 and **operant-ndn-ids**-sensor3 nodes and this connectivity had to be restored by the Operant team. It is unclear if this attack was successful, and we were not able to validate the results at the time of testing.

7.2.7 Sensitive Data Exposure

A privacy attack of interest packets attempts to learn of sensitive information being requested by data consumers. The interest packets captured all show names encoded in some manner, and further investigation would be necessary to determine the nature and sensitivity of any information included in the interest packet names.



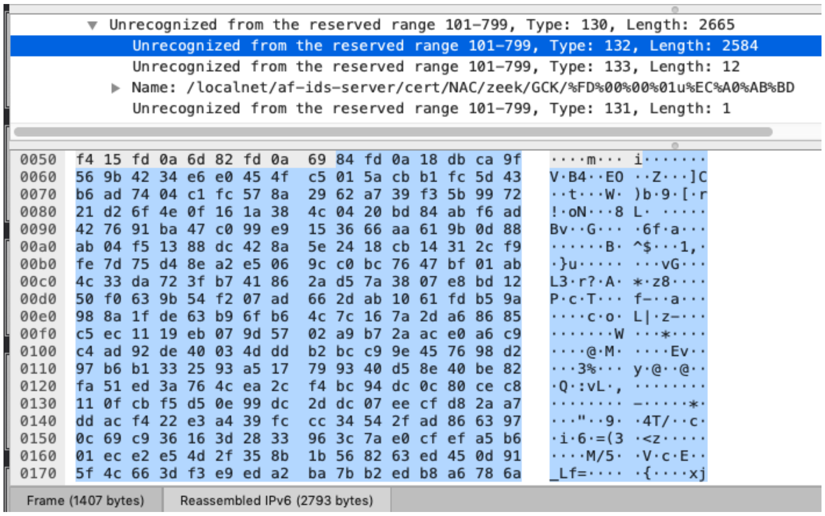
```
▼ Interest, Type: 5, Length: 162, Name: /localnet/zeekall/x%9Ccd%60%60h%D0%A9%3F%FDB%EA%
  ▼ Name [truncated]: /localnet/zeekall/x%9Ccd%60%60h%D0%A9%3F%FDB%EA%B0%3E%03%89%80%11
      NameComponent: localnet
      NameComponent: zeekall
      NameComponent [truncated]: x%9Ccd%60%60h%D0%A9%3F%FDB%EA%B0%3E%03%89%80%11%88Y%7
  ▼ Selectors, Type: 9, Length: 2, MustBeFresh: Yes

0020  00 00 00 00 00 88 ff 02  00 00 00 00 00 00 00 00   ········ ········
0030  00 00 00 00 12 34 c1 e7  dc 2b 00 ac d4 a5 05 a2   ·····4·· ·+······
0040  07 92 08 08 6c 6f 63 61  6c 6e 65 74 08 07 7a 65   ····loca lnet··ze
0050  65 6b 61 6c 6c 08 7d 78  9c 63 64 60 60 68 d0 a9   ekall·}x ·cd``h··
0060  3f fd 42 ea b0 3e 03 89  80 11 88 59 7b 96 cd ce   ?·B··>·· ···Y{···
0070  51 bf 9a 8d 4f cd 1c ae  dd 6a 77 7a 17 04 90 6a   Q···O··· ·jwz···j
0080  fe 60 00 20 f7 fb fc 6c  f5 fc b5 79 9a 05 ba f8   ·`· ···l ···y····
0090  b1 b7 5d 32 97 ee 8a 47  d1 c3 0d 47 85 5b 57 b5   ··]2···G ···G·[W·
00a0  2a ff 50 a0 b5 f9 b8 e2  14 24 4e 6e 3a 19 09 00   *·P····· ·$Nn:···
00b0  39 9d 53 92 36 70 a5 b7  c1 00 28 f5 17 a1 b2 82   9·S·6p·· ··(·····
00c0  1c 33 91 c3 1c 16 6e b4  4e ab b4 ce 8f 03 05 00   ·3····n· N·······
```

Figure 7-16: Interest packets with names encoded

The success or failure of the **Name Hijacking** attack is still unknown as of the time of testing.

7.2.8 Broken Access Control

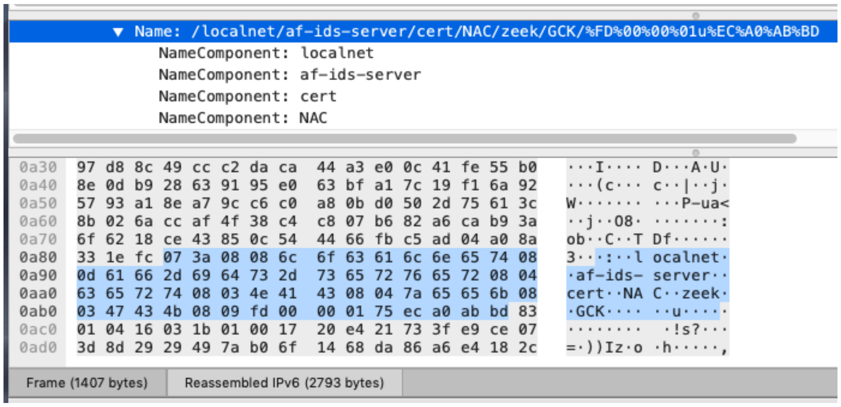
Requests for contents are made by the requestor and a producer transmits the requested content using a data packet. Application hijacking entails capturing data encryption keys during key distribution in order to produce bogus application data. The NDN packets were reviewed for encryption and authentication information. The produced data sent in response to interest requests appears to be random, that is, it is assumed to be encrypted application data as shown below.



```
▼ Unrecognized from the reserved range 101–799, Type: 130, Length: 2665
    Unrecognized from the reserved range 101–799, Type: 132, Length: 2584
    Unrecognized from the reserved range 101–799, Type: 133, Length: 12
  ▶ Name: /localnet/af-ids-server/cert/NAC/zeek/GCK/%FD%00%00%01u%EC%A0%AB%BD
    Unrecognized from the reserved range 101–799, Type: 131, Length: 1

0050  f4 15 fd 0a 6d 82 fd 0a  69 84 fd 0a 18 db ca 9f   ····m··· i·······
0060  56 9b 42 34 e6 e0 45 4f  c5 01 5a cb b1 fc 5d 43   V·B4··EO ··Z···]C
0070  b6 ad 74 04 c1 fc 57 8a  29 62 a7 39 f3 5b 99 72   ··t···W· )b·9·[·r
0080  21 d2 6f 4e 0f 16 1a 38  4c 04 20 bd 84 ab f6 ad   !·oN···8 L· ·····
0090  42 76 91 ba 47 c0 99 e9  15 36 66 aa 61 9b 0d 88   Bv··G··· ·6f·a···
00a0  ab 04 f5 13 88 dc 42 8a  5e 24 18 cb 14 31 2c f9   ······B· ^$···1,·
00b0  fe 7d 75 d4 8e a2 e5 06  9c c0 bc 76 47 bf 01 ab   ·}u····· ···vG···
00c0  4c 33 da 72 3f b7 41 86  2a d5 7a 38 07 e8 bd 12   L3·r?·A· *·z8····
00d0  50 f0 63 9b 54 f2 07 ad  66 2d ab 10 61 fd b5 9a   P·c·T··· f-··a···
00e0  98 8a 1f de 63 b9 6f b6  4c 7c 16 7a 2d a6 86 85   ····c·o· L|·z-···
00f0  c5 ec 11 19 eb 07 9d 57  02 a9 b7 2a ac e0 a6 c9   ·······W ···*····
0100  c4 ad 92 de 40 03 4d dd  b2 bc c9 9e 45 76 98 d2   ····@·M· ····Ev··
0110  97 b6 b1 33 25 93 a5 17  79 93 40 d5 8e 40 be 82   ···3%··· y·@··@··
0120  fa 51 ed 3a 76 4c ea 2c  f4 bc 94 dc 0c 80 ce c8   ·Q·:vL·, ········
0130  11 0f cb f5 d5 0e 99 dc  2d dc 07 ee cf d8 2a a7   ········ -·····*·
0140  dd ac f4 22 e3 a4 39 fc  cc 34 54 2f ad 86 63 97   ···"··9· ·4T/··c·
0150  0c 69 c9 36 16 3d 28 33  96 3c 7a e0 cf ef a5 b6   ·i·6·=(3 ·<z·····
0160  01 ec e2 e5 4d 2f 35 8b  1b 56 82 63 ed 45 0d 91   ····M/5· ·V·c·E··
0170  5f 4c 66 3d f3 e9 ed a2  ba 7b b2 ed b8 a6 78 6a   _Lf=···· ·{····xj
```

Figure 7-17: Random encrypted data received in response to interest packets

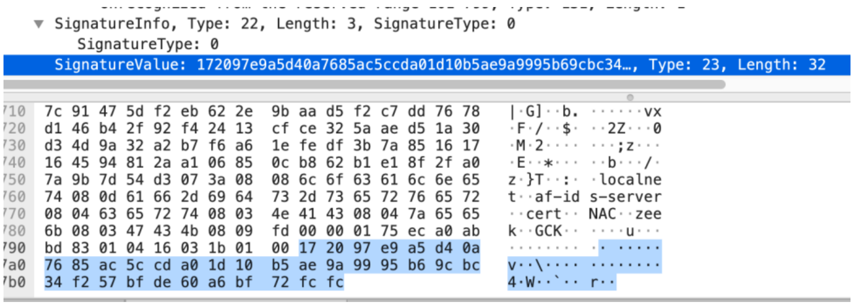
The observed captures do not appear to contain key distribution exchanges - likely that the producer and consumer nodes have been provisioned with keys beforehand. It is possible that an additional application process is encrypting the NDN data independent of the producer application. The packets are apparently protected using name-based access control (NAC). The packet content includes a certificate name, with a 12-byte field possibly used to identify decryption key certificate information for previously stored keys.



```
▼ Name: /localnet/af-ids-server/cert/NAC/zeek/GCK/%FD%00%00%01u%EC%A0%AB%BD
      NameComponent: localnet
      NameComponent: af-ids-server
      NameComponent: cert
      NameComponent: NAC

0a30  97 d8 8c 49 cc c2 da ca  44 a3 e0 0c 41 fe 55 b0   ···I···· D···A·U·
0a40  8e 0d b9 28 63 91 95 e0  63 bf a1 7c 19 f1 6a 92   ···(c··· c··|··j·
0a50  57 93 a1 8e a7 9c c6 c0  a8 0b d0 50 2d 75 61 3c   W······· ···P-ua<
0a60  8b 02 6a cc af 4f 38 c4  c8 07 b6 82 a6 ca b9 3a   ··j··O8· ·······:
0a70  6f 62 18 ce 43 85 0c 54  44 66 fb c5 ad 04 a0 8a   ob··C··T Df······
0a80  33 1e fc 07 3a 08 08 6c  6f 63 61 6c 6e 65 74 08   3···:··l ocalnet·
0a90  0d 61 66 2d 69 64 73 2d  73 65 72 76 65 72 08 04   ·af-ids- server··
0aa0  63 65 72 74 08 03 4e 41  43 08 04 7a 65 65 6b 08   cert··NA C··zeek·
0ab0  03 47 43 4b 08 09 fd 00  00 01 75 ec a0 ab bd 83   ·GCK···· ··u·····
0ac0  01 04 16 03 1b 01 00 17  20 e4 21 73 3f e9 ce 07   ········  ·!s?···
0ad0  3d 8d 29 29 49 7a b0 6f  14 68 da 86 a6 e4 18 2c   =·))Iz·o ·h·····,

Frame (1407 bytes)    Reassembled IPv6 (2793 bytes)
```

Figure 7-18: Packets protected using name-based access control (NAC)

Packets are signed, but signatures do not include a KeyLocator TLV. The nodes perhaps have previously stored certificates for all applicable 'producer' nodes. This was not confirmed at this time.



```
▼ SignatureInfo, Type: 22, Length: 3, SignatureType: 0
     SignatureType: 0
  SignatureValue: 172097e9a5d40a7685ac5ccda01d10b5ae9a9995b69cbc34…, Type: 23, Length: 32

710  7c 91 47 5d f2 eb 62 2e  9b aa d5 f2 c7 dd 76 78   |·G]··b. ······vx
720  d1 46 b4 2f 92 f4 24 13  cf ce 32 5a ae d5 1a 30   ·F·/··$· ··2Z···0
730  d3 4d 9a 32 a2 b7 f6 a6  1e fe df 3b 7a 85 16 17   ·M·2···· ···;z···
740  16 45 94 81 2a a1 06 85  0c b8 62 b1 e1 8f 2f a0   ·E··*··· ··b···/·
750  7a 9b 7d 54 d3 07 3a 08  08 6c 6f 63 61 6c 6e 65   z·}T··:· ·localne
760  74 08 0d 61 66 2d 69 64  73 2d 73 65 72 76 65 72   t··af-id s-server
770  08 04 63 65 72 74 08 03  4e 41 43 08 04 7a 65 65   ··cert·· NAC··zee
780  6b 08 03 47 43 4b 08 09  fd 00 00 01 75 ec a0 ab   k··GCK·· ····u···
790  bd 83 01 04 16 03 1b 01  00 17 20 97 e9 a5 d4 0a   ········ ·· ·····
7a0  76 85 ac 5c cd a0 1d 10  b5 ae 9a 99 95 b6 9c bc   v··\···· ········
7b0  34 f2 57 bf de 60 a6 bf  72 fc fc                  4·W··`·· r··
```

Figure 7-19: tlv-key-locator not found in packets

7.2.9 *Security Misconfiguration*

This involves exploiting security controls that are not severely implemented using OpenSCAP tool.

Each node was scanned using 2 profiles:

1. "Standard System Security Profile for Red Hat Enterprise Linux 7"
    1. Less strict with 51 rules
    2. Profile ID = xccdf_org.ssgproject.content_profile_standard
2. "Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171)"
    1. More strict with 104 rules
    2. Profile ID = xccdf_org.ssgproject.content_profile_cui

Shown below, is a summary of the number and severity of failed rules.

Profile -->	xccdf_org.ssgproject.content_profile_standard				xccdf_org.ssgproject.content_profile_cui			
Severity --> **Node**	**High**	**Medium**	**Low**	**Other**	**High**	**Medium**	**Low**	**Other**
operant-ndn-ids-server	3	31	2	0	6	51	2	18
operant-ndn-ids-sensor1	1	31	2	0	5	52	2	18
operant-ndn-ids-sensor2	1	31	2	0	5	53	2	18
operant-ndn-ids-sensor3	1	30	2	0	5	53	2	18

OpenSCAP High-Severity Findings

Title	**Rationale**	**Description**	**Affected Node(s)**	**Scan Profile(s)**	**Assessed Security Level**
Prevent Login to Accounts With Empty Password	If an account has an empty password, anyone could log in and run commands with the privileges of that account. Accounts with empty passwords should never be used in operational environments.	If an account is configured for password authentication but does not have an assigned password, it may be possible to log into the account without authentication. Remove any instances of the nullok option in /etc/pam.d/system-auth to prevent logins with empty passwords.	• operant-ndn-ids-sensor1 • operant-ndn-ids-sensor2 • operant-ndn-ids-sensor3 • operant-ndn-ids-server	• xccdf_org.ssgproject.content_profile_standard • xccdf_org.ssgproject.content_profile_cui	Medium. Should be easy to fix.
Disable Ctrl-Alt-Del Burst Action	A locally logged-in user who presses Ctrl-Alt-Del, when at the console, can reboot the system. If	By default, SystemD will reboot the system if the Ctrl-Alt-Del key sequence is pressed Ctrl-Alt-Delete more than 7 times in 2 seconds. To configure the system to ignore the CtrlAltDelBurstAction setting, add or	• operant-ndn-ids-sensor1 • operant-ndn-	• xccdf_org.ssgproject.content_profile_cui	Low. Medium if testbed availability is critical or if operational environments will be set

	accidentally pressed, as could happen in the case of mixed OS environment, this can create the risk of short-term loss of availability of systems due to unintentional reboot.	modify the following to /etc/systemd/system.conf: CtrlAltDelBurstAction=none	ids-sensor 2 • operant-ndn-ids-sensor 3 • operant-ndn-ids-server		up using the same steps as the testbed.
Disable Ctrl-Alt-Del Reboot Activation	A locally logged-in user who presses Ctrl-Alt-Del, when at the console, can reboot the system. If accidentally pressed, as could happen in the case of mixed OS environment, this can create the risk of short-term loss of availability of systems due to unintentional reboot.	By default, SystemD will reboot the system if the Ctrl-Alt-Del key sequence is pressed. To configure the system to ignore the Ctrl-Alt-Del key sequence from the command line instead of rebooting the system, do either of the following: ln -sf /dev/null /etc/systemd/system/ctrl-alt-del.target or systemctl mask ctrl-alt-del.target Do not simply delete the /usr/lib/systemd/system/ctrl-alt-del.service file, as this file may be restored during future system updates.	• operant-ndn-ids-sensor 1 • operant-ndn-ids-sensor 2 • operant-ndn-ids-sensor 3 • operant-ndn-ids-server	• xccdf_org.ssgproject.content_profile_cui	Low. Medium if testbed availability is critical or if operational environments will be set up using the same steps as the testbed.
Enable FIPS Mode in GRUB2	Use of weak or untested encryption algorithms undermines the purposes of utilizing encryption to protect data. The operating system must implement cryptographic modules adhering to the higher standards approved by the federal government since this provides assurance they have been tested and validated.	To ensure FIPS mode is enabled, install package dracut-fips, and rebuild initramfs by running the following commands: $ sudo yum install dracut-fips dracut -f After the dracut command has been run, add the argument fips=1 to the default GRUB 2 command line for the Linux operating system in /etc/default/grub, in the manner below: GRUB_CMDLINE_LINUX="crashkernel=auto rd.lvm.lv=VolGroup/LogVol06 rd.lvm.lv=VolGroup/lv_swap rhgb quiet rd.shell=0 fips=1" Finally, rebuild the grub.cfg file by using the grub2-mkconfig -o command as follows: • On BIOS-based machines, issue the following command as root: ~]# grub2-mkconfig -o /boot/grub2/grub.cfg • On UEFI-based machines, issue the following command as root: ~]# grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg	• operant-ndn-ids-sensor 1 • operant-ndn-ids-sensor 2 • operant-ndn-ids-sensor 3 • operant-ndn-ids-server	• xccdf_org.ssgproject.content_profile_cui	Low. Medium if testbed availability is critical or if operational environments will be set up using the same steps as the testbed.

Ensure gpgcheck Enabled for Local Packages	Changes to any software components can have significant effects to the overall security of the operating system. This requirement ensures the software has not been tampered and has been provided by a trusted vendor. Accordingly, patches, service packs, device drivers, or operating system components must be signed with a certificate recognized and approved by the organization.	yum should be configured to verify the signature(s) of local packages prior to installation. To configure yum to verify signatures of local packages, set the localpkg_gpgcheck to 1 in /etc/yum.conf.	- operant-ndn-ids-sensor1 - operant-ndn-ids-sensor2 - operant-ndn-ids-sensor3 - operant-ndn-ids-server	- xccdf_org.ssgproject.content_profile_cui	Low. Medium if testbed availability is critical or if operational environments will be set up using the same steps as the testbed.
Ensure gpgcheck Enabled for All yum Package Repositories	Verifying the authenticity of the software prior to installation validates the integrity of the patch or upgrade received from a vendor. This ensures the software has not been tampered with and that it has been provided by a trusted vendor. Self-signed certificates are disallowed by this requirement. Certificates used to verify the software must be from an approved Certificate Authority (CA)."	To ensure signature checking is not disabled for any repos, remove any lines from files in /etc/yum.repos.d of the form: gpgcheck=0	- operant-ndn-ids-server	- xccdf_org.ssgproject.content_profile_cui	Low. Medium if testbed availability is critical or if operational environments will be set up using the same steps as the testbed.
Verify and Correct File	Permissions on system binaries and	The RPM package management system can check file access permissions of installed software packages, including many that are	- operant-ndn-	- xccdf_org.ssgproject.content_profile_standard	Low.

Permissions with RPM	configuration files that are too generous could allow an unauthorized user to gain privileges that they should not have. The permissions set by the vendor should be maintained. Any deviations from this baseline should be investigated.	important to system security. Verify that the file permissions of system files and commands match vendor values. Check the file permissions with the following command:
		$ sudo rpm -Va | awk '{ if (substr($0,2,1)=="M") print $NF }'
		Output indicates files that do not match vendor defaults. After locating a file with incorrect permissions, run the following command to determine which package owns it:
		$ rpm -qf *FILENAME*
		Next, run the following command to reset its permissions to the correct values:
		$ sudo rpm --setperms *PACKAGENAME*
			ids-server		Medium if testbed availability is critical or if operational environments will be set up using the same steps as the testbed.

Verify File Hashes with RPM	The hashes of important files like system executables should match the information given by the RPM database. Executables with erroneous hashes could be a sign of nefarious activity on the system.	Without cryptographic integrity protections, system executables and files can be altered by unauthorized users without detection. The RPM package management system can check the hashes of installed software packages, including many that are important to system security. To verify that the cryptographic hash of system files and commands match vendor values, run the following command to list which files on the system have hashes that differ from what is expected by the RPM database:
		$ rpm -Va | grep '^..5'
		A "c" in the second column indicates that a file is a configuration file, which may appropriately be expected to change. If the file was not expected to change, investigate the cause of the change using audit logs or other means. The package can then be reinstalled to restore the file. Run the following command to determine which package owns the file:
		$ rpm -qf *FILENAME*
		The package can be reinstalled from a yum repository using the command:
		$ sudo yum reinstall *PACKAGENAME*
		Alternatively, the package can be reinstalled from trusted media using the command:
		$ sudo rpm -Uvh *PACKAGENAME*
			• operant-ndn-ids-server	• xccdf_org.ssgproject.content_profile_standard	Low.
					Medium if testbed availability is critical or if operational environments will be set up using the same steps as the testbed.

8 CONCLUSION

We have conducted a cybersecurity evaluation of the novel NDN protocol with respect to its use for DER systems. We noted and exercised key attack scenarios and methods that could potentially be used to exploit them but were shown to be unsuccessful. However, the attack scenarios are not a complete representation of every way in which an adversary could potentially exploit the system. Our evaluation included white box testing of the source codes in the ndn-ind and nfd-ind repositories. These security vulnerabilities have been evaluated and recommendations provided, and Operant has welcomed these findings and responded with a remediation plan. This report also contains the results of the penetration testing aimed at increasing Operant's awareness in prioritizing mitigations for the development of a robust NDN architecture.

REFERENCES

[1] https://ivisa.named-data.net/html/acm_icn_2014_ndn_tutorial_1.html accessed 4/20/2020

[2] https://named-data.net/wp-content/uploads/2019/02/SecurityOverview.pdf accessed 4/22/2020

[3] https://named-data.net/ndn-testbed/ accessed 4/21/2020

[4] Scalability, Security, and Trust in the Power Grid: An Information-Centric Smart Grid Architecture using Named Data Networking, presented 5/4/2020 by Prof Jay Misra for Operant

[5] https://www.nsnam.org, accessed May 20, 2020

[6] https://ndnsim.net/current/, accessed May 20, 2020

[7] G. Ravikumar, D. Ameme, S. Misra, S. Brahma and R. Tourani, "iCASM: An Information-Centric Network Architecture for Wide Area Measurement Systems," in IEEE Transactions on Smart Grid, doi: 10.1109/TSG.2020.2971429. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8982088, accessed 5/20/2020

[8] https://www.youtube.com/channel/UCxUlSAzwjTsR-nEz3cbll4w, accessed May 13, 2020

[9] https://named-data.net/project/

[10]https://named-data.net/project/execsummary/

[11] https://named-data.net/doc/NDN-packet-spec/current/

[12][https://named-data.net/project/ndn-design-principles/ accessed 4/21/2020

[13]*Chapweske, Justin (November 29, 2001). "HTTP Extensions for a Content-Addressable Web". www-talk. W3C.*

[14]https://en.wikipedia.org/wiki/Magnet_URI_scheme, accessed 5/20/2020

[15]https://www.youtube.com/watch?v=p26GODPxGGE, accessed 5/18, 2020

[16]https://www.caida.org/publications/papers/2017/named_data_networking_2016-2017/named_data_networking_2016-2017.pdf

[17]https://named-data.net/wp-content/uploads/2018/04/ndn-0057-2-ndn-security.pdf accessed 5/27/2020

[18]https://ieeexplore.ieee.org/document/8925064

[19]T. Nguyen, X. Marchal, G. Doyen, T. Cholez and R. Cogranne, "Content Poisoning in Named Data Networking: Comprehensive characterization of real deployment," 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Lisbon, 2017, pp. 72-80, doi: 10.23919/INM.2017.7987266.

[20]https://named-data.net/project/faq/ accessed June 8, 2020

[21]Dannewitz, Christian & Golic, Jovan & Ohlman, Börje & Ahlgren, Bengt. (2010). Secure Naming for a Network of Information. 1 - 6. 10.1109/INFCOMW.2010.5466661.

[22]https://en.wikipedia.org/wiki/Named_data_networking accessed June 8, 2020

This page left blank

DISTRIBUTION

Email—Internal

Name	Org.	Sandia Email Address
Summer Ferreira	08812	srferre@sandia.gov
Brian Gaines	09366	bgaines@sandia.gov
Ifeoma Onunkwo	09366	ionunkw@sandia.gov
Technical Library	01977	sanddocs@sandia.gov

Email—External [redacted]

Name	Company Email Address	Company Name
Mayank Saxena	mayank.saxena@operantnetworks.com	Operant Networks, Inc.
Randall King	randy.king@operantnetworks.com	Operant Networks, Inc.

This page left blank

Sandia
National
Laboratories

Sandia National Laboratories
is a multimission laboratory
managed and operated by
National Technology &
Engineering Solutions of
Sandia LLC, a wholly owned
subsidiary of Honeywell
International Inc. for the U.S.
Department of Energy's
National Nuclear Security
Administration under contract
DE-NA0003525.