Real-Time and Interactive Attacks on DNP3 Critical Infrastructure Using Scapy

Nicholas R. Rodofile[1] **Kenneth Radke**[2] **Ernest Foo**[3]

Information Security Discipline,
Queensland University of Technology,
Email: { n.rodofile[1], k.radke[2], e.foo[3] } @qut.edu.au

Abstract

The Distributed Network Protocol v3.0 (DNP3) is one of the most widely used protocols, to control national infrastructure. Widely used interactive packet manipulation tools, such as Scapy, have not yet been augmented to parse and create DNP3 frames (Biondi 2014). In this paper we extend Scapy to include DNP3, thus allowing us to perform attacks on DNP3 in real-time. Our contribution builds on East et al. (2009), who proposed a range of possible attacks on DNP3. We implement several of these attacks to validate our DNP3 extension to Scapy, then executed the attacks on real world equipment. We present our results, showing that many of these theoretical attacks would be unsuccessful in an Ethernet-based network.

Keywords: substations , Distributed Network Protocol 3.0, DNP3, critical Infrastructure security, Scapy

1 Introduction

Despite the prevalence of Industrial Control System (ICS) networks connected to corporate IT networks, there are limited techniques that can be used to correctly detect cyber-attacks or forms of intrusion on industrial networks. To develop signature-based Intrusion Detection Systems (IDSs), attack signatures are required to help identify malicious network traffic. By using real critical infrastructure equipment for the attacks, it shows the realism of the attacks. Prior to this paper, not only did few sets of DNP3 attack signatures exist, but there was no easy way to create malicious network traffic. Hence, there is a need to enhance tools, such as the interactive packet manipulation Python library Scapy, that exist for traditional IT communication networks, such as Scapy, to include control system protocols. Our DNP3 extension of Scapy allows sets of DNP3 attack signatures to be easily created.

The outline of our paper proceeds as follows: Section 2 describes the relevant background information of previous research in Critical Infrastructure security for DNP3. In Section 3 we provide an overview of DNP3. We describe our attack tool that executes our DNP3 extension to Scapy in Section 4 and our attacks and their results in Section 5, and we then conclude in Section 6.

2 Related Work

There is currently ongoing research to help secure control systems using DNP3 and various other control system protocols. Lee et al. (2014) simulated a DNP3 attack using OpenDNP3. The research involved a small scale testbed. The testbed simulated actuators and sensors of a hydro-power system. The attacks using DNP3 were conducted on computers simulating DNP3 systems and were not tested on real equipment used in industry.

Researchers at the Sandia National Laboratory Urias et al. (2012) created testbeds to demonstrate attacks on SCADA Systems by exploiting network vulnerabilities in a corporate network. The teams managed to perform several attacks on an ethernet-based SCADA system which had, among other SCADA protocols, DNP3-based communications. These attacks were general to an industrial network.

East et al. (2009) identified 28 attacks and 91 threats through their taxonomy and analysis of the DNP3 specification. These attacks were all theoretical attacks, based on the specification of DNP3.

Although there is clearly research interest in ICS, in particular DNP3, there has been little research into attacks that work on real-world critical infrastructure equipment, to aid in the development of IDS attack signatures. We selected three of East et al. (2009) theoretical attacks, the Length Overflow attack, Address Alteration attack, and Configuration Capture attack, as we found these attacks interesting. We tested each of the attacks on real-world equipment using our DNP3 extension to Scapy.

3 DNP3 Overview

A typical electricity distribution company's substation will have an operations centre, making use of master devices, to manage multiple slave devices running in outstations that are usually at a remote location. These devices can be configured to communicate using DNP3 over an Ethernet connection using TCP/IP. The DNP3 frames, shown in Figure 1, are layered within a TCP frame.

A DNP3 frame begins with a start (**START**) field to identify the beginning of the frame, which is then followed by the length field (**LEN**), used to provide the length in octets of the entire DNP3 frame. The control (**CTRL**) field defines the frames direction, transaction initiator, and function. The destination field (**DST**) identifies the destination for the frame, whereas the source field (**SRC**) identifies the source of the frame. A Cyclic Redundancy Check (**CRC**) field in the data link layer provides integrity for the other eight octets of the data link segment. A CRC field is inserted after every sixteen octets (data

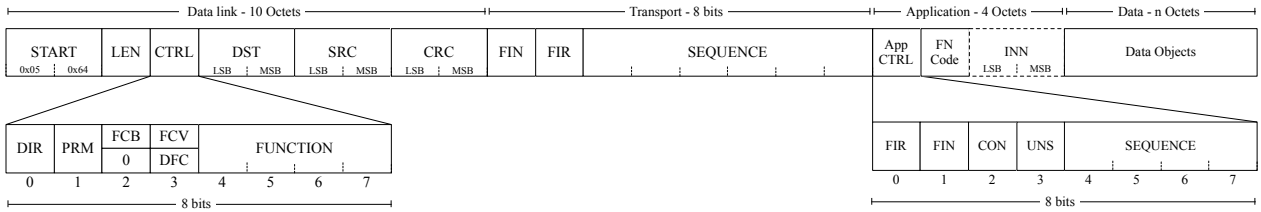

Figure 1: Structure of a DNP3 Frame

chunks) to keep the DNP3 frame's Integrity (East et al. 2009, *IEEE Standard for Electric Power Systems Communications-DNP3* 2012, Curtis 2005).

The Transport segment, as shown in Figure 1, is used to for data reassembly in the receiving device for fragmented DNP3 Frames. We implemented the Transport segment in our DNP3 Scapy extension, allowing us to manipulate and corrupt the reassembly process in the receiving device.

The application fragment begins with an application control (**App CTRL**) field, which is made of several sub-fields. The unsolicited (**UNS**) field flags that the fragment is an unsolicited response. If the UNS flag is unset, then the fragment is associated with a sequence number. The sequence (**SEQ**) field is used to assure the segments are not duplicated, missing and that they are in order, as the sequence number increments on each fragment. The Function Code (**FN Code**) field is used to identify the purpose of the fragment. There exists 34 defined function codes for application requests. A response DNP3 frame would contain additional the Internal Indications (**IIN**) to indicate the state and conditions of a slave device.

4 Attack Tool

To ensure that an attack tool is able to create appropriate attack signatures for the development and testing of IDSs, we identify the necessary requirements for our DNP3 extension by analysing the 34 theoretical attacks described in East et al.'s 2009 taxonomy, by which we identified the following three critical requirements for our DNP3 attack tool. Spoofing frames, Capture frames, and modify Frames. These requirements allow the user to send or modify messages to target devices, in order to manipulate the target's functionality or cause it to malfunction.

The DNP3 extension allows a user to craft a DNP3 frame, and layer the frame with each DNP3 segment to manipulate the fields inside the crafted frame. To create each of the DNP3 frame segments, we extended the `Packet` class of the Scapy library, and closely followed the DNP3 specification (*IEEE Standard for Electric Power Systems Communications-DNP3* 2012). Our Scapy extension is able to process the data chunks and generate the CRC values for each of the data chunks when a segment within the DNP3 frame is updated. To determine if the IIN field is required, we checked the DIR field in the data link control function. As part of our extension we did not implement the DNP3 data objects, instead we must construct the object payload from raw hex values.

5 Attacks

To evaluate our Scapy implementation of DNP3, we implemented a series of theoretical attacks described by East et al. (2009). The successful execution of each attack required the interception of all TCP frames, followed by the modification of the DNP3 payload, before forwarding the reconstructed frame on to its intended destination.

5.1 Setup

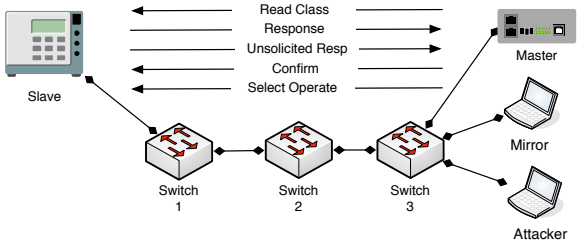

Figure 2: Testbed network setup

To perform our attacks, we used a testbed setup consisting of critical infrastructure equipment used in the real world. The testbed setup made use of the one-to-one network architecture (see Figure 2). The testbed used a Supervisory Control and Data Acquisition (SCADA) gateway as the master device and an Intelligent Electronic Device (IED) as the slave device. To connect the master and slave devices, we used three interconnecting layer 2 industrial network switches. The master was attached to switch 1, and the slave to switch 3. The attacker laptop was connected to switch 3. To allow us to monitor the experiments, we configured a mirror port to listen to all network traffic on the master device's port. All of the mirrored traffic was captured on a separate laptop running Wireshark. We had the ability to see, via Wireshark and switch mirror ports, what was being sent to and from the slave IED and the master SCADA Gateway, and being received and sent by the attacker.

As part of the setup, we configured the master to request class object variables from the slave device using a "READ" DNP3 request message. The master would request data using DNP3 class variables, identified as Class 1, Class 2, Class 3, then finally Class 123. The slave device would send a DNP3 response message containing the relevant values of the requested classes after each DNP3 request. To physically indicate DNP3 communication, we also configured a mechanism to illuminate an LED on the slave, in response to a button push on the LED's corresponding button. The mechanism involved (see Figure 2) the slave IED sending an unsolicited DNP3 response to the master device. The Confirm is followed

by a "SELECT OPERATE" DNP3 message from the master, resulting with the LED on the IED to turn on. Therefore, if the LED on the IED illuminates then a message has flowed from the slave device to the master device, advising a change of state. Our physical indicator can represent something as significant as a circuit breaker opening or an indication of a running motor.

5.2 Eavesdropping

Before any other attack could successfully be executed, we needed to take control of the communications between the slave IED and the master gateway. Therefore, for our first attack, our objective was to intercept, and eavesdrop all network traffic between the master and slave devices, through the use of Address Resolution Protocol (ARP) cache poisoning.

Results As a result of the ARP cache poisoning, the attacker intercepted all frames, then redirected the frame on to it's intended destination. During the eavesdropping attack, there was a manipulation of the MAC addresses by the attacker using our tool, this can be seen in Figure 3. As can be seen, the Source (MAC xx:15:b8) on the first line of Figure 3 is attempting to send a TCP message to a Destination (MAC xx:80:e2). However, the message first goes to our attacker machine, Toshiba 58:77:b3, on row one of Figure 3. The attacker's machine then updated the addresses and then the same message is sent from Toshiba 58:77:b3 on to the intended recipient, being xx:80:e2, on row two of Figure 3. Similar manipulations can also be seen in the pair of messages on rows 3 and 4 of Figure 3.

Protocol	Source MAC	Source IP	Destination MAC	Destination IP
TCP	:15:b8	10.192.228.5	Toshiba_58:77:b3	10.192.226.19
TCP	Toshiba_58:77:b3	10.192.228.5	_ :80:e2	10.192.226.19
DNP 3.0	:80:e2	10.192.226.19	Toshiba_58:77:b3	10.192.228.5
DNP 3.0	Toshiba_58:77:b3	10.192.226.19	:15:b8	10.192.228.5
TCP	:15:b8	10.192.228.5	Toshiba_58:77:b3	10.192.226.19

Figure 3: Eavesdropping Attack - Wireshark capture screen-shot of communication on the Attacker's machine

During the faithful forwarding, the physical indication mechanism was successful as the LED did illuminate on the button press from the slave device, but we note there was an increased delay between the button press and the actual lighting of the LED. This meant that the delay added by the frame manipulation of our attacker computer was inside the tolerances of the critical infrastructure devices we tested.

5.3 Address Alteration

The second attack performed was East et al's destination address alteration attack (East et al. 2009). The objective of the attack is to intercept and modify the DNP3 destination address of each DNP3 Frame (shown in the DST field of Figure 1). Note: the DNP3 destination address, for example "10", is different from the ethernet destination address (which is a MAC address). Our master device was assigned the address of '0', whereas the slave device was assigned the address of '10'. For the address alteration attack, we pass the intercepted frame into a function where the DNP3 destination address is modified to be '2'. According to East et al, changing the destination of a the DNP3 frame may cause other devices to reply or the intended device will fail to receive the message. We performed this attack to see if our extension is capable of altering the DNP3 destination address. After updating the DST field, we then had to recalculate and update the DNP3 data link CRC field (see Figure 1).

Results After analysing the network capture from the attacker, we were able to see that all DNP3 addresses passing through the attacker were updated regardless of its DNP3 destination address. Our mirror capture of the master device shows all messages sent to the master had the destination address of '2'.

During the address alteration attack our physical indication mechanism failed meaning the LED did not illuminate on the button press from the slave device.

This attack has some interesting practical limitations. If the intention is to forward a frame meant for one physical device to another physical device, this will not work as the surrounding TCP connection would need to be established with the second device. Therefore, the only way this attack could have a result of an action being taken by a different master, would be if there were multiple masters configured on the one device.

As a further development for this test, we configured a second address on our device, but the slave device had a limitation that, although it could have multiple masters, each of the masters needed to be at a separate IP address. Our intention was to have two physical indication mechanisms, for the two different addresses on the one device. The attack would mean that pressing the button for one physical indication mechanism would activate the other physical indication mechanism. The limitation that different IP addresses are required for different DNP3 addresses, means that the TCP layer would send the response to the updated DNP3 address to a different physical device. This attack may work in a serial environment, but there seems no conceptual way for the attack to work in an ethernet-based environment.

5.4 Length Overflow

For our next experiment, the objective of the attack is to intercept and modify the length field of a DNP3 payload to a new length of 44 octets. The CRC field also needed to be recalculated and updated to reflect the new length value. We conducted this attack on every frame containing a DNP3 payload. East et al expects that the length overflow attack will result in "data corruption, unexpected actions and device crashes" (East et al. 2009).

Results We analysed the traffic capture from the attacker's interface. We can see that all DNP3 frames from the slave device had the length field updated to 44. Analysis of the new frames using Wireshark reveal that manipulated frames are flagged as malformed. Further, since the DNP3 message is part of the TCP payload, adjusting the length means that some of the encapsulating TCP frame is now shown as part of the DNP message. Wireshark flagging an error is only an indication that the target device would also reject the frame. However, during this length overflow attack, our physical indication mechanism failed as the LED did not illuminate on the button press from the slave device meaning that the DNP3 devices we were testing also rejected these frames. As such, the attack becomes a Denial of Service (DoS) attack. We also inspected each device's commissioning tool, and found no indication of errors or data corruption. It

may seem that there is potential, if there were multiple DNP3 payloads inside the one TCP frame, that the first DNP3 message could consume part of the second message and still be a valid message. However, since the CRC check at the end of a DNP3 message is every 16 octets and at the start of the (next) DNP3 message is after 8 octets there is almost no combination of two messages that would result in a working combined message due to length alteration.

5.5 Configuration Capture

The objective of this attack is to assert that the configuration file of the target outstation is corrupted. East et al. (2009) believed that this would cause the master to transmit a new configuration file to the slave. allowing the attacker to intercept the configuration file to use in a replay attack.

To implement this attack we set the 5th bit in the second Internal Indications (IIN) field (see Figure 1) to true. We performed this attack to see if firstly our extension is capable to updating the IIN field of a DNP3 fragment without interrupting the unsolicited response, and secondly to capture the upload of a new configuration.

We conducted two attacks. For the first attack we were able to intercept all messages between the master and slave devices, but we only manipulated frames that did not contain an unsolicited message. This means that our physical indication mechanism was excluded from the attack. For the second attack, the attacker intercepted all DNP3 frames, manipulating the 5th bit of the second octet in the IIN of all frames.

Results During the first configuration capture attack, the physical indication mechanism was successful as the LED did illuminate the button press from the slave device. After analysing the traffic capture from the attackers interface in Wireshark, we noticed there was no traffic indicating the transfer of configuration files in response to the solicited messages which did have their IIN field updated. During the second configuration capture attack, our physical indication mechanism failed, meaning the LED did illuminate on the button press from the slave device. After analysing the captured traffic from the attacker, again there was no traffic indicating the transfer of configuration files.

The equipment we were using had a full deployment configuration used in critical infrastructure protection, before the specifics of our tests. Even so, the master did not have configurations for the slaves stored, and thus there was no potential for this attack to result in the master sending new configuration files. The configuration of the slave takes several minutes from an engineering workstation, all of which time the device is not functioning. Since such a downtime is unacceptable in a critical infrastructure environment, many critical infrastructure environments may not store configurations on their master devices for automatic re-deployment at unspecified times due to this corruption assertion. Our result and the standard configuration of these critical infrastructure devices suggests that East et al.'s theoretical attack would rarely work.

6 Conclusion

In conclusion, we presented our DNP3 Scapy extension and implement a series of East et al.'s 2009 theoretical DNP3 attacks on real critical infrastructure equipment. In Table 1 we outline the number of frames intercepted by the attack tool, the number of intercepted frames that contained a DNP payload that were modified beyond changing the IP addresses the intended recipient's IP address, and the result of the physical response mechanism as outlined in Section 5.1. For each of the attacks conducted we observed that our attack tool was able to intercept frames from both the master and slave devices. As can be seen by the results of our tests, many of East et al.'s theoretical attacks have limited chances of success on real world equipment. However, other attacks are more plausible and we will present these attacks in future work.

Attack	Intercepted	Modified	LED
Forwarding	400	0	True
Address Alteration	50	30	False
Length Overflow	400	15	False
Config Capture	100	8	True
Config Capture 2	100	18	False

Table 1: Results of Attacks. Only DNP3, not ethernet, modifications shown

The results from our attacks on the critical infrastructure equipment validate that Scapy is a suitable tool for attack signature generation for DNP3, as our tool was able to create and parse DNP3 frames in real-time. In closing, our tool will assist with the development attack signatures for IDSs.

Acknowledgment

This work was supported in part by Australian Research Council Linkage Grant LP120200246, Practical Cyber Security for Next Generation Power Transmission Networks.

References

Biondi, P. (2014), 'Scapy'.
URL: *http://www.secdev.org/projects/scapy/*

Curtis, K. (2005), 'A DNP3 Protocol Primer', *DNP User Group* .

East, S., Butts, J., Papa, M. & Shenoi, S. (2009), A Taxonomy of Attacks on the DNP3 Protocol, *in* 'Critical Infrastructure Protection III', Springer, pp. 67–81.

IEEE Standard for Electric Power Systems Communications-DNP3 (2012), Technical report.

Lee, D., Kim, H., Kim, K. & Yoo, P. D. (2014), Simulated attack on dnp3 protocol in scada system, *in* 'Proceedings of The 31th Symposium on Cryptography and Information Security Kagoshima (SCIS 2014)'.

Urias, V., Van Leeuwen, B. & Richardson, B. (2012), Supervisory Command and Data Acquisition (SCADA) system cyber security analysis using a live, virtual, and constructive (LVC) testbed, *in* 'Proceedings of Military Communications Conference 2012-MILCOM 2012', IEEE, pp. 1–8.