TRITON: The First ICS Cyber Attack on Safety Instrument Systems
Understanding the Malware, Its Communications and Its OT Payload

Alessandro Di Pinto, Younes Dragoni, Andrea Carcano
Black Hat USA 2018 - Research Paper
Table of Contents

1. Abstract................................................................................................................................. 1

2. Introducing SIS and the TRITON cyber attack ................................................................. 2
   2.1. SIS (Safety Instrumented Systems) .............................................................................. 2
   2.2. The TRITON cyber attack and why it is important ..................................................... 2
   2.3. The key components of the attack ................................................................................. 3

3. Understanding TRITON through Reverse Engineering .................................................... 4
   3.1. Turning an ‘Undocumented Device’ into malicious code .............................................. 4
   3.2. Obtaining the TRITON engineering toolset ................................................................. 5
   3.3. Obtaining the Triconex controller .................................................................................. 5
   3.4. Reverse engineering the TriStation suite ...................................................................... 7
   3.5. Undocumented power users .......................................................................................... 9
   3.6. Understanding the TriStation protocol ......................................................................... 12
   3.7. Parsing the Triconex hardware definition ................................................................... 13
   3.8. TriStation protocol stack implementation .................................................................... 14

4. Defending against TRITON: new tools to help ................................................................. 15
   4.1. Wireshark dissector (LUA script) for TriStation protocol .......................................... 15
   4.2. Triconex Honeypot Tool: simulating a real Triconex Controller ............................... 16

5. Demonstrating a working malicious TRITON payload ...................................................... 17
   5.1. TRITON’s malicious payload was missing ................................................................. 17
   5.2. Demonstrating a working TRITON malicious payload ............................................... 18

6. What TRITON means for securing Industrial Control Systems ....................................... 19

Appendix A – Triconex Hardware Definition List .................................................................... 20

Bibliography ............................................................................................................................. 25

About the Authors ..................................................................................................................... 26

About Nozomi Networks ........................................................................................................... 26
1. Abstract

In December 2017 it was reported that a Middle Eastern oil and gas petrochemical facility [1] had undergone a safety system shutdown as the result of a malware attack. The malware, named TRITON (also known as TRISIS or HatMan), went beyond other industrial cyber attacks by directly interacting with a Safety Instrumented System (SIS). SIS are the last line of automated safety defense for industrial facilities, designed to prevent equipment failure and catastrophic incidents such as explosions or fire.

Based on the significance of this industrial cyber attack, it warranted an in-depth analysis. We were determined to understand the TRITON malware itself, as well as the resources it took to create it. We also sought to gain insights that would help industrial operators defend their control systems from such attacks in the future.

Our challenge was to learn how to turn an undocumented device – the Triconex controller from Schneider Electric, which was the target of the attack – into malicious code. To do so we first focused on obtaining the TRITON engineering toolset. We combined Internet sleuthing with asking the right people the right questions, to obtain the information we needed.

Our next hurdle was obtaining the Triconex controller. Employing a variety of global ecommerce websites, we purchased the components needed and assembled them into a working environment. We were unable to find one key component – the marshalling cables, but we overcame that problem by using brute force to directly connect two panels.

Now that we had a working system, we proceeded to reverse engineer the TriStation suite of software used on the engineering workstation that communicates with the SIS controller. That activity, combined with malware analysis, allowed us to deeply dissect the TriStation proprietary communication protocol used by the Triconex controller.

Our findings allowed us to develop two new tools [2] to help the ICS community secure Triconex SIS. The first tool, the TriStation Protocol Plug-in for Wireshark, allows an engineer to visually see and comprehend TriStation communications. It also identifies hardware connected to the safety controller and passively detects TRITON activity in network communications.

The second tool, the Triconex Honeypot Tool, can be used by defense teams to simulate SIS controllers on the network, using them like a honeypot [12] to detect reconnaissance scans and capture malicious payloads.

While the TRITON malware attack failed to deliver a malicious OT payload, we successfully used its capabilities to implement new programs in the Triconex controller and to execute a malicious payload.

Our research shows that the effort, skills and financial resources needed to create the TRITON malware are not that high – certainly not at the level where nation state-sponsored resources are required. Knowing this, industrial asset owners should act immediately to monitor their SIS and secure them against external attacks. We also urge SIS equipment makers to provide more robust built-in security for these vitally important systems.
2. Introducing SIS and the TRITON cyber attack

2.1. SIS (Safety Instrumented Systems)

Safety Instrumented Systems (SIS), also known as Industrial Safety Systems, are designed to prevent industrial incidents, such as equipment or operational failures, from causing damage, injury, loss of life, or serious environmental harm. Examples of extreme consequences would be explosions, fires, oil spills, floods or even nuclear system meltdowns.

The equipment used in SIS are a special type of PLC (Programmable Logic Controllers) designed with predictability and reliability in mind. They include multiple main processors, built-in diagnostics, redundancy management systems, and failure detection for inputs and outputs. Best practices for securing SIS include running them on isolated networks and carefully restricting access rights.

If out-of-range operating conditions occur, SIS perform control functions that shut the process down in a safe, predictable way. Also, while designed to never fail, should an SIS failure actually occur, it will do so in a predictable manner. Thus, the worst-case scenario is known and planned for ahead of time.

SIS are the last line of automated defense for industrial facilities, though it should be noted that mechanical fail-safes also exist.

2.2. The TRITON cyber attack and why it is important

In December 2017, FireEye reported that it had recently worked with an industrial operator whose facility was attacked by a new type of ICS malware, which they named TRITON (other organizations have named it TRISIS or HatMan).

The attack reprogrammed the facility’s SIS controllers, causing them to enter a failed state, and resulting in an automatic shutdown of the industrial process. The investigation following the shutdown led to the detection of the hacking attempt.

The targeted facility was subsequently identified as a Saudi Arabian petrochemical processing plant, while the SIS that was attacked was a Triconex Safety Instrumented System from Schneider Electric. This type of SIS is commissioned in a consistent way across many industries and is widely used.

TRITON is one of only a handful of malware with the ability to impact the physical process of an industrial control system. Previous examples include Stuxnet (Iran, uranium enrichment centrifuges, 2010) and Industroyer/Crash Override (Ukraine, power grids, 2015 and 2016).

TRITON goes well beyond earlier attacks and is considered a milestone industrial cyber attack because it directly interacts with, and controls, SIS. It raises the possibility of a cyber attack leading to unpredictable and dangerous plant outcomes, without the protection of a last line of safety defense.

Based on the significance of this industrial cyber attack, the Nozomi Networks security research group decided to do an in-depth analysis of it. We were determined to understand TRITON and the resources needed to create
it. We also sought to gain insights that would help industrial operators defend their control systems from such attacks in the future.

2.3. The key components of the attack

The attack began with penetration of the IT network using well-documented [9] easily-detected attack methods. The attackers moved to the OT (Operational Technology) network through systems that were accessible to both environments.

Once on the OT network, the threat actors were able to infect the engineering workstation for the SIS system, usually situated in an isolated network segment. The infection probably used a social engineering technique whereby the engineer received or downloaded a file with a legitimate file name, in this case “trilog.exe”. The name suggests that the dropper file is a clean executable dealing with something related to Triconex and its logging capabilities (TRIconex LOGging filename).

The main purpose of the dropper file was to deliver the malicious payload into the target, in this case the SIS controller. Soon after the execution, the dropper connected to the targeted Triconex, injecting the real malware payload inside its memory.

The malware payload was contained in two separate binary files called inject.bin and imain.bin. One of the actions taken by the dropper was to read, inject and execute these files into the memory of the Triconex.

- **inject.bin** contained the code which exploits a specific 0-day in order to execute the content of the file "imain.bin".

- **imain.bin** contained the final code that allows a remote user to gain full control of the SIS device.

Following is reported information about the malicious files involved in the infection phase:

<table>
<thead>
<tr>
<th>Filename</th>
<th>MD5</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>trilog.exe</td>
<td>6c39c3f4a08d3d78f2eb973a94bd7718</td>
<td>Dropper</td>
</tr>
<tr>
<td>inject.bin</td>
<td>0544d425c7555dc4e9d76b571f31f500</td>
<td>Backdoor injector</td>
</tr>
<tr>
<td>imain.bin</td>
<td>437f135ba179959a580412e564d3107f</td>
<td>Backdoor code</td>
</tr>
</tbody>
</table>

The dropper was developed in Python and compiled inside the trilog.exe executable. It contains the implementation of the TriStation protocol reverse-engineered by the threat actors and used to interact with the targeted device.

The first action performed by the researchers who got the TRITON sample was to decompile the executable, extract the Python code, and use it as the starting point for the further analysis.
3. Understanding TRITON through Reverse Engineering

3.1. Turning an ‘Undocumented Device’ into malicious code

Despite the routine techniques employed to gain access to the victim’s OT environment, the threat actors behind the TRITON malware attack had to go through a significant learning curve. They had to learn about the Triconex SIS controller itself and TriStation, the proprietary network communication protocol it uses.

Obtaining both the Triconex hardware and software related to it was essential for recreating the full working environment needed for experimenting and writing malicious code. With a working system, the threat actors were able to intercept and analyze traffic transmitted on the wire. They were also able to reverse engineer the TriStation software.

Both methods were needed to dissect the proprietary protocol, extracting information from it and re-implementing it inside TRITON.

To study TRITON ourselves, we needed to recreate the targeted SIS in our lab environment and obtain as much documented information as possible about its functioning and communications. The following sections describe how we went about this.
3.2. Obtaining the TRITON engineering toolset

We used multiple channels to procure the necessary components, as follows:

- Vendor website – the Schneider Electric website contains useful information
- Consultation with key experts – sometimes asking the right person, nicely, is effective
- Asset owners – operations and security staff are our friends – and the best sources of information!
- Internet searches – there is a lot of freely available information, such as:
  - Installation CDs sold on e-commerce sites
  - Loose executables & archives on forum sites
  - Open directories, FTP servers, etc.

In short, a combination of Internet sleuthing and asking the right people the right questions allowed us to obtain a great deal of the information required.

3.3. Obtaining the Triconex controller

Of course, the key item we needed was a Triconex controller. This is where “free” ended and we had to spend some money. A budget of $5-10K is required – the reality is that most ICS equipment is expensive.

Another thing we took into consideration was getting multiple copies of the controller for teardown purposes – and in case we bricked (damaged) one, and it no longer worked.

Here are some of the sources we considered for acquiring the SIS controller:

- From the vendor – it’s possible to buy new equipment directly from Schneider Electric, but it is not the least expensive way.
- From online e-commerce sites such as eBay or Alibaba – we found components and used devices on these sites. In some cases, new ones were listed for sale, including full-warranties.

We needed to keep in mind that the systems had to be compatible for everything to work together, and we needed the same model that was used in the TRITON attack. The TRITON vulnerable SIS controllers are:

- Triconex MP3008 main processor modules running firmware versions 10.0–10.4[^3]

Note that these models of the Triconex use a MPC860 PowerPC processor. Newer models use a different processor (ARM) and are thus not vulnerable to the version of TRITON we studied.
The equipment used for our research is described below.

Main Low-Density Chassis:

- 1.02 3008/N Tricon Enhanced Main Processor v10.3 - Firmware Meta Number: ETSX6236
- 1.05 4329/N/G NCM (Network Communications Module)
- 1.09 3503/E/EN Discrete Input, 24 V, 32 points
- 1.10 Marshalling Connector 2652 -310 DO
- 1.12 3604/E/EN Discrete Output, 24 VDC, 16 points
- Terminator Panel 2652-1

We also tested the injection phase and did some analysis with the 3008/N Tricon Enhanced Main Processor v9.10 BUILD 66.

One item needed was the marshalling cables, which are used to connect the terminal panel to the connector module. This allows communication with field devices through the digital output module.

- We wanted to get this communication to work so that we could test variations of TRITON that would succeed in delivering commands that disrupt the safety process, as described in Section 5.

It turned out to be the only equipment challenge we had – it seems marshalling cables are very hard to find.

Since we were unable to locate the cables, the only solution was to manually, using brute force, connect the terminal panel directly to the connector module. Fortunately, it worked!

Figure 2 - Marshalling cables were impossible to locate
3.4. Reverse engineering the TriStation suite

The software installed on the engineering workstation is a gold mine for threat actors because it contains all the information needed to interact with the controller, including how to recognize different statuses and attached modules. TriStation 1131 v4.9.0 (build 117) is able to connect with the hardware version targeted by the malware, and all of our analysis has been performed on that version.

The Schneider Development Team has kindly included useful information about their files – which unfortunately can also be used by an attacker to better understand the software architecture and its general structure.

Figure 3 - Lacking the marshalling cables, these two boards were directly connected.
Starting with a detailed description of every file installed by the software, it is quite simple to understand where to look for different components.

Figure 4 - The TriStation suite files are nicely identified
3.5. Undocumented power users

The TriStation software also stores project files containing key information about the Triconex program, configuration and behavior inside a password-protected file with .PT2 extension. The assumption is that only authorized engineers can access the project files containing the proprietary code executed inside the SIS devices.

However, the TriStation software v4.9.0 (and prior versions), also contain two undocumented power users able to open any project file regardless of the robustness of its password. These users are likely used by the product support team to help customers with technical issues. While having operational value, these users can also be abused by threat actors to access password-protected project documents without the proper credentials.

The two undocumented users are:

*T****FD
*T****BD

For user *T****FD, the login requires a password that is available through reverse engineering and additional authentication in the form of a support ticket number. This confirms that the role of the user is for customer support purposes.

![Image of TriStation software interface with login screen](image)

*Figure 5 – Undocumented user T****FD requires the extra step of a support ticket number to login*
User "****BD" however, has the capability to access a project file just by inserting the hard-coded credentials extracted from the TriStation software.

![Figure 6 – Undocumented user ****BD accessing a password protected project file with its static password](image)

Once user "****BD" is logged in, a hidden menu is enabled that is not available to other users.

![Figure 7 – A hidden menu is enabled after logging-in as user ****BD](image)
From an attacker’s prospective, this scenario could be useful. The hidden menu provides access to a great number of extra features not available to a normal user.

Moreover, when logged as T****BD, the TriStation software exposes internal information about the linking/compilation phases of the system logic. This additional data has a high value for threat actors because it describes in detail the commands involved during program compilation.

The level of information available to this undocumented power user makes it significantly easier for threat actors to create malicious OT payloads. However, our research found no connection between the TRITON malware and this hidden menu, and the malware did not leverage these undocumented users.

It should be noted that these undocumented users exist for TriStation 1131 v4.9.0 and earlier versions only, according to Schneider Electric.
3.6. Understanding the TriStation protocol

In accordance with the file descriptions, the code delegated to manage the network communication is located inside the DLL “tricom.dll”. By analyzing this file, we extracted a wealth of information about the protocol’s definition that documents its behavior.

```c
int __thiscall CAPLTricon::SendMessageA(CAPLTricon *this, int a2, unsigned __Int8 *a3, Int a4)
{
    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]
    v4 = this;
    if (*((__WORD *)this + 274) >= 1u )
    {
        v5 = sub_16005380(a2);
        CAPLTricon::DisplayDebugMessage(v5, 1u, aReqS, v5);
    }
    v6 = *((__DWORD *)v4 + 150);
    if ( v6 == 2 )
    {
        v10 = 0;
        v11 = (char *)v4 + 1828;
        if ( a4 )
        {
            qmemcpv(v11, a3, a4);
            v10 = 0;
            *((__WORD *)v4 + 916) = a4;
        }
        else
        {
            *((__WORD *)v4 + 916) = 10;
            *((__BYTE *)v4 + 1831) = *((__BYTE *)v4 + 595);
            v12 = *((__WORD *)v4 + 918);
            *((__BYTE *)v4 + 1830) = a2;
            v13 = v12;
            *((__WORD *)v4 + 916) = 0;
            *((__WORD *)v4 + 917) = 0;
            v11 = 0;
            *((__BYTE *)v4 + 1829) = 0;
            if ( (signed int)v12 > 0 )
            {
                do
                {
                    v12 += (unsigned __Int8)v11[v10++];
                    while ( v10 < v13 );
                } while ( *(v10 < 1v13 ));
                *((__WORD *)v4 + 917) = v12;
                *((__BYTE *)v4 + 593) = 2;
                CAPLTricon::SetState(v4, stateRunning|statePaused);
                CAPLTricon::DumpMessage(v4, aSend, (CAPLTricon *)(%char *)v4 + 1828));
```

Figure 8 - Decompiled code showing low-level packet management
3.7. Parsing the Triconex hardware definition

The Low-Density and High-Density chassis used by the Triconex hardware supports several different modules which are described inside a proprietary file called "TR1HWDEF.HWD". The malware does not contain any information related to the hardware definition, probably because the threat actors did not invest time to reverse engineer that part.

However, the definition file contains useful information for network traffic analysis that identifies which modules are attached to the remote Triconex hardware.

We invested time in reverse engineering, and successfully parsed the hardware information. The details are given in Appendix A – Triconex Hardware Definition List.

![Figure 9 - Automatic parsing of the file that describes Triconex hardware information](image)

We also added this parsing capability to our Wireshark dissector (see section 4.1), which thus provides information about deployed Triconex hardware modules.
3.8. TriStation protocol stack implementation

TriStation is not only the software suite used inside the engineering workstation. There is also a proprietary network protocol called “TriStation Protocol” operating on UDP/IP over port 1502.

The malware files related to TriStation implementation are found inside these files:

<table>
<thead>
<tr>
<th>Filename</th>
<th>MD5</th>
<th>Compilation time</th>
</tr>
</thead>
<tbody>
<tr>
<td>TsBase.pyc</td>
<td>288166952f934146be172f6353e9a1f5</td>
<td>2017-08-03 16:52:33</td>
</tr>
<tr>
<td>TsHi.pyc</td>
<td>27c69aa39024d21ea109cc9d944a04</td>
<td>2017-08-04 08:04:01</td>
</tr>
<tr>
<td>TsLow.pyc</td>
<td>f6b3a73c8c87506aca430671360ce15</td>
<td>2017-08-03 16:46:51</td>
</tr>
<tr>
<td>TS_cnames.pyc</td>
<td>e98f4f3505f0bf90e17554fbc97bba9</td>
<td>2017-08-03 12:26:36</td>
</tr>
</tbody>
</table>

All the protocol definitions are contained inside the Python-compiled file “TS_cnames.pyc” and can be easily decompiled and extracted as shown in the image below. The code contained in the file is a great explanation of the function codes implemented by the TriStation protocol, showing how deep the threat actors went during reverse engineering.

![Figure 10 - Part of TRITON's Python code, showing TriStation protocol states](image)
Combining the malware analysis with reverse engineering activity performed with the workstation software, we were able to deeply dissect the TriStation protocol. It allows for communications between engineering workstations (master) and Triconex controllers (slave), equipped with specific network modules (NCM).

Through our reverse engineering of the Triconex we were able to develop tools for helping industrial organizations and researchers understand SIS communications (section 3.6), and create and demonstrate a malicious TRITON payload (section 5).

4. Defending against TRITON: new tools to help

4.1. Wireshark dissector (LUA script) for TriStation protocol

During our analysis, we developed an extended Wireshark dissector using Lua script, called the **TriStation Protocol Plug-in for Wireshark**. [2]

It offers several useful features for engineers working with the TriStation protocol:

- Indication of the direction of communication
- Function codes translated as descriptive text
- Extraction of transmitted PLC programs
- Identification of connected hardware
- Detection of the TRITON malware in network communications
- […]

![Figure 11 - The TriStation protocol plug-in for Wireshark detects a sample of TRITON during the injection phase](image)
Our Wireshark dissector allows an ICS engineer to see useful information during a network packet inspection. This includes the direction of the packet and the name associated with the function code (also known as a "command"). It also easily parses the packets, extracting the PLC programs/functions downloaded to the Triconex devices, and presenting them in the Wireshark GUI.

Furthermore, the dissector automatically detects TRITON malware using specific indicators obtained during malware analysis performed in the laboratory. This feature was developed only with the purpose of demonstrating that it’s possible to identify ICS malware on the network using passive techniques.

We strongly recommend that organizations do not depend on Wireshark for intrusion detection. If the threat actors change just one byte of the malware, it will not be detected. Instead, applications specifically designed for detecting ICS malware intrusion, which use multiple techniques for identifying irregularities, should be employed.

4.2. Triconex Honeypot Tool: simulating a real Triconex Controller

The extensive knowledge we gained during our analysis allowed us to develop a tool that simulates a Triconex controller. This tool was created to help us during the deep inspection of specific packets related to how the hardware modules and the modules’ LEDs information are encoded inside the packets.

The main goal was to create a script that simulates a Triconex controller and responds to the TriStation diagnostic software developed by Schneider Electric. This diagnostic tool can be used by an OT engineer to query a specific controller deployed in the field and obtain its information.

Using our script, we simulated the controller’s behavior, convincing the diagnostic tool that we are a real controller sending its status, including:

- Controller version
- Controller status
- Controller memory
- Chassis type
- Connected modules
- Status LEDs
- LED type and color
- Hardware key position (RUN/STOP/PROGRAM)
- Project name
- Modules configuration match/mismatch (project \(\Leftrightarrow\) chassis)

Although the script is currently only a proof-of-concept, it can be expanded to support an extensive number of functions. Its realism can be increased to the point where it is indistinguishable from a real controller. In addition, the script can be executed by a regular, inexpensive computer attached to the network.
The *Triconex Honeypot Tool* can be used by defense teams to simulate SIS controllers with particular system configurations, using them like a honeypot to detect reconnaissance scans and capture malicious payloads. It can therefore play a useful role in detecting unknown traffic targeting a SIS network.

5. **Demonstrating a working malicious TRITON payload**

5.1. **TRITON’s malicious payload was missing**

TRITON gave the threat actor the ability to read, write and execute code directly inside the controller’s memory. The main advantage of this architecture was its modularity, providing the attackers the means to dynamically send a specific payload inside the memory and execute it.

But this approach also assists the security analyst or researcher, as the malware itself exposes the basic capabilities for low-level interaction with the OT device.

In the case of TRITON, the final stage of the attack is missing. Soon after the targeted Triconex was implanted with the malware, one of the main processors triggered a redundancy alarm. This forced all three main processors to start the safety shutdown process. It is unclear whether the issue was triggered by the dropper injecting the malware in memory, or by an unexpected fault generated while executing the OT payload.

The targeted version of the Triconex hardware is running a CPU based on the PowerPC architecture. During program updates, the code is sent directly in machine code. The malware author followed this technique, developing the ability to write and execute directly in the memory itself.

One of the more likely assumptions is that the attacker injected a specific OT payload containing an invalid memory access which resulted in a crash.
5.2. Demonstrating a working TRITON malicious payload

During our research we recreated a fully working Triconex infrastructure, including connecting a field device to the controller. We attached a compressor and a balloon, similar to the set-up first used to demonstrate the Stuxnet ICS malware. [11]

The program run by the controller supervises the compressor executing an inflating and deflating process in a specific, synchronized and ongoing way. Then we used the TRITON capabilities to inject a command that modified the behavior of the security supervisor, causing the balloon to overinflate and finally generate an explosion.

This achieved the final goal of our research, showing how we were able to use the knowledge we obtained through reverse engineering TRITON to compromise the safe functioning of the Triconex controller.
6. What TRITON means for securing Industrial Control Systems

Over the last twenty years it has become easier and easier for threat actors to launch ICS cyber attacks. More and more tools and examples are readily available, lowering the bar for the knowledge and skills needed by intruders.

For example, before Stuxnet (2010), there were no example ICS malware frameworks available over the Internet. Now, there are many, including TRITON. Other things that have changed include:

- Global ecommerce platforms that make it easy to buy the documentation and equipment needed to recreate a SIS environment
- A rapid increase in the number of disclosed ICS vulnerabilities
- Shodan and similar search engines which make it easy to find Internet-connected ICS devices
- Increased connectivity with IT and Internet-based systems has greatly increased the attack surface

When we began our research, we were determined to understand TRITON and the resources it took to create it. This paper shows that the effort, skills and financial resources needed to create the TRITON malware framework, while not insignificant, are not that high – certainly not at the level where nation state-sponsored resources are required. While the level of difficulty of executing a cyber attack varies according to the cyber security defenses, networking architecture and equipment of each facility, the development of the malware itself does not require high levels of resources.

Knowing this, industrial asset owners should act immediately to monitor their SIS and secure them against external attacks. We also urge SIS equipment makers to provide more robust built-in security for these vitally important systems.
## Appendix A – Triconex Hardware Definition List

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3501/E/T/TN Discrete Input, 115 V, 32 points</td>
</tr>
<tr>
<td>2</td>
<td>3502/E/EN Discrete Input, 48 V, 32 points</td>
</tr>
<tr>
<td>3</td>
<td>3503/E/EN Discrete Input, 24 V, 32 points</td>
</tr>
<tr>
<td>7</td>
<td>3505/E/EN Discrete Input, 24 V, Low Threshold, 32 points</td>
</tr>
<tr>
<td>11</td>
<td>3508/E Discrete Input, 230 V, 32 points</td>
</tr>
<tr>
<td>17</td>
<td>3601/E/T/TN Discrete Output, 115 VAC, 16 points</td>
</tr>
<tr>
<td>19</td>
<td>3603/B/E/T/TN Discrete Output, 120 VDC, 16 points</td>
</tr>
<tr>
<td>20</td>
<td>3604/E/EN Discrete Output, 24 VDC, 16 points</td>
</tr>
<tr>
<td>23</td>
<td>3608/E Discrete Output, 48 VAC, 16 points</td>
</tr>
<tr>
<td>24</td>
<td>3607/E/EN Discrete Output, 48 VDC, 16 points</td>
</tr>
<tr>
<td>29</td>
<td>6603 Discrete Output, 24 VDC, 16 points</td>
</tr>
<tr>
<td>30</td>
<td>6602 Discrete Output, 48 VDC, 16 points</td>
</tr>
<tr>
<td>31</td>
<td>6601 Discrete Output, 115 VAC, 16 points</td>
</tr>
<tr>
<td>32</td>
<td>3701/N Analog Input, 10 V input, 32 points</td>
</tr>
<tr>
<td>33</td>
<td>3700/A/AN Analog Input, 5 V input, 32 points</td>
</tr>
<tr>
<td>38</td>
<td>3510/N Pulse Input, 8 points</td>
</tr>
<tr>
<td>40</td>
<td>3801 Analog I/O, 10 V inp, 4-20ma out, 8 inputs, 4 outputs</td>
</tr>
<tr>
<td>41</td>
<td>3800 Analog I/O, 5 V inp, 4-20ma out, 8 inputs, 4 outputs</td>
</tr>
<tr>
<td>42</td>
<td>6810 Analog Output, 4-20ma, 4 points; Pulse Input, 4 points</td>
</tr>
<tr>
<td>45</td>
<td>3511 Enhanced Pulse Input, 8 points</td>
</tr>
<tr>
<td>47</td>
<td>3515 Pulse Totalizer Input, 32 Data points, 32 Reset points</td>
</tr>
<tr>
<td>48</td>
<td>4119/A/AN EICM (Intelligent Communications Module)</td>
</tr>
<tr>
<td>49</td>
<td>420-/N,421-/N Remote Extender Module, Primary/Remote</td>
</tr>
<tr>
<td>50</td>
<td>6211 ICM (Intelligent Communications Module)</td>
</tr>
<tr>
<td>51</td>
<td>4509 Honeywell Data Highway Interface Module (HIM)</td>
</tr>
<tr>
<td>52</td>
<td>4409 Safety Manager Module</td>
</tr>
<tr>
<td>53</td>
<td>6215 Honeywell Data Highway Interface Module (HIM)&quot;</td>
</tr>
<tr>
<td>54</td>
<td>GPSI Global Positioning System Interface</td>
</tr>
<tr>
<td>55</td>
<td>4329/N/G NCM (Network Communications Module)</td>
</tr>
<tr>
<td>56</td>
<td>4609/N ACM (Advanced Communications Module)</td>
</tr>
<tr>
<td>57</td>
<td>4351B TCM-B (Tricon Communication Module/B - Copper)</td>
</tr>
<tr>
<td>58</td>
<td>4352B TCM-B (Tricon Communication Module/B - Fiber)</td>
</tr>
<tr>
<td>59</td>
<td>4353 TCM/OPC (Tricon Communication Module OPC - Copper)</td>
</tr>
<tr>
<td>60</td>
<td>4354 TCM/OPC (Tricon Communication Module OPC - Fiber)</td>
</tr>
<tr>
<td>71</td>
<td>3531 Discrete Input (simplx), 115 V, 32 points</td>
</tr>
<tr>
<td>72</td>
<td>3532 Discrete Input (simplx), 48 V, 32 points</td>
</tr>
<tr>
<td>73</td>
<td>3533 Discrete Input (simplx), 24 V, 32 points</td>
</tr>
<tr>
<td>84</td>
<td>4351 TCM (Tricon Communications Module - Copper)</td>
</tr>
<tr>
<td>85</td>
<td>4352 TCM (Tricon Communications Module - Fiber)</td>
</tr>
<tr>
<td>86</td>
<td>4351A TCM-A (Tricon Communication Module/A - Copper)</td>
</tr>
<tr>
<td>87</td>
<td>4352A/N TCM-A (Tricon Communication Module/A - Fiber)</td>
</tr>
<tr>
<td>88</td>
<td>3664 Dual Discrete Output, 24 V, 32 points, Serial</td>
</tr>
<tr>
<td>89</td>
<td>3674 Dual Discrete Output, 24 V, 32 points, Fail-Safe</td>
</tr>
<tr>
<td>90</td>
<td>3667 Dual Discrete Output, 48 V, 32 points, Serial</td>
</tr>
<tr>
<td>91</td>
<td>3677 Dual Discrete Output, 48 V, 32 points, Parallel</td>
</tr>
<tr>
<td>92</td>
<td>3663 Dual Discrete Output, 120V, 32 points, Serial</td>
</tr>
<tr>
<td>93</td>
<td>3673 Dual Discrete Output, 120V, 32 points, Parallel</td>
</tr>
<tr>
<td>94</td>
<td>3720 Enh Analog Input, 5V, 64 points, Configurable</td>
</tr>
<tr>
<td>95</td>
<td>3721/N Enh Differential Analog Input, +/-5V, 32 points, Configurable</td>
</tr>
<tr>
<td>104</td>
<td>3805/E/H/EN Analog Output, 4-20ma, 8 points</td>
</tr>
<tr>
<td>105</td>
<td>3807 Servo Control Analog Output, -60 to +60mA</td>
</tr>
<tr>
<td>107</td>
<td>6613 Supv Disc Output, 24 V, 16 points</td>
</tr>
<tr>
<td>106</td>
<td>3806 Analog Output, 4-20ma (6 pts), 4-320ma (2 pts)</td>
</tr>
<tr>
<td>108</td>
<td>6612 Supv Disc Output, 48 V, 16 points</td>
</tr>
<tr>
<td>109</td>
<td>6617 Supv Disc Output, 120 V, 16 points</td>
</tr>
<tr>
<td>110</td>
<td>6703 Analog Input (isolated), 2 V, 16 points</td>
</tr>
<tr>
<td>111</td>
<td>3635/E Discrete Output (simplx), Relay Cntct, Norm clsd, 32 pts</td>
</tr>
<tr>
<td>112</td>
<td>3636/R/T/TN Discrete Output (simplx), Relay Cntct, Norm open, 32 pts</td>
</tr>
<tr>
<td>113</td>
<td>3611/E Supv Discrete Output, 115 VAC, 8 points</td>
</tr>
<tr>
<td>129</td>
<td>6507 Discrete Input, 120 VDC, 32 points</td>
</tr>
<tr>
<td>130</td>
<td>6502 Discrete Input, 48 VDC, 32 points</td>
</tr>
<tr>
<td>133</td>
<td>6503 Discrete Input, 24 VDC, 32 points</td>
</tr>
<tr>
<td>134</td>
<td>6501 Discrete Input, 115 VAC, 32 points</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>136</td>
<td>6508 Discrete Input, 48 VAC, 32 points</td>
</tr>
<tr>
<td>138</td>
<td>6708 Isol Thermocouple Input Type E $dgC$, 16 points</td>
</tr>
<tr>
<td>139</td>
<td>6708 Isol Thermocouple Input Type J $dgC$, 16 points</td>
</tr>
<tr>
<td>140</td>
<td>6708 Isol Thermocouple Input Type K $dgC$, 16 points</td>
</tr>
<tr>
<td>141</td>
<td>6708 Isol Thermocouple Input Type T $dgC$, 16 points</td>
</tr>
<tr>
<td>146</td>
<td>3706/A/AN Non-Isol Thermocouple Input Type J $dgF$ 32 points</td>
</tr>
<tr>
<td>147</td>
<td>3706/A/AN Non-Isol Thermocouple Input Type K $dgF$ 32 points</td>
</tr>
<tr>
<td>148</td>
<td>3706/A/AN Non-Isol Thermocouple Input Type T $dgF$ 32 points</td>
</tr>
<tr>
<td>149</td>
<td>3706/A/AN Non-Isol Thermocouple Input Type J $dgC$ 32 points</td>
</tr>
<tr>
<td>150</td>
<td>3706/A/AN Non-Isol Thermocouple Input Type K $dgC$ 32 points</td>
</tr>
<tr>
<td>151</td>
<td>3706/A/AN Non-Isol Thermocouple Input Type T $dgC$ 32 points</td>
</tr>
<tr>
<td>152</td>
<td>3704/E/EN Analog Input, 5 V, DnS, 64 points</td>
</tr>
<tr>
<td>153</td>
<td>3704/E/EN Analog Input, 10 V, DnS, 64 points</td>
</tr>
<tr>
<td>154</td>
<td>3704/E/EN Analog Input, 5 V, UpS, 64 points</td>
</tr>
<tr>
<td>155</td>
<td>3704/E/EN Analog Input, 10 V, UpS, 64 points</td>
</tr>
<tr>
<td>156</td>
<td>3504/E/EN Discrete Input, 24 VDC, 64 points</td>
</tr>
<tr>
<td>157</td>
<td>3504/E/EN Discrete Input, 48 VDC, 64 points</td>
</tr>
<tr>
<td>158</td>
<td>3625/N Supervised Discrete Output, 24V, 32 points, Configurable</td>
</tr>
<tr>
<td>160</td>
<td>6700 Analog Input, 5 V, DnS, 32 points</td>
</tr>
<tr>
<td>161</td>
<td>6700 Analog Input, 10 V, DnS, 32 points</td>
</tr>
<tr>
<td>162</td>
<td>6700 Analog Input, 5 V, UpS, 32 points</td>
</tr>
<tr>
<td>163</td>
<td>6700 Analog Input, 10 V, UpS, 32 points</td>
</tr>
<tr>
<td>180</td>
<td>3703/E/EN Enh Isol Analog Input, 5 V, DnS, 16 points</td>
</tr>
<tr>
<td>181</td>
<td>3703/E/EN Enh Isol Analog Input, 10 V, DnS, 16 points</td>
</tr>
<tr>
<td>182</td>
<td>3703/E/EN 3Enh Isol Analog Input, 5 V, UpS, 16 points</td>
</tr>
<tr>
<td>183</td>
<td>3703/E/EN Enh Isol Analog Input, 10 V, UpS, 16 points</td>
</tr>
<tr>
<td>184</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type J $dgC$ DnS, 16 points</td>
</tr>
<tr>
<td>185</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type K $dgC$ DnS, 16 points</td>
</tr>
<tr>
<td>186</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type T $dgC$ DnS, 16 points</td>
</tr>
<tr>
<td>187</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type E $dgC$ DnS, 16 points</td>
</tr>
<tr>
<td>192</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type J $dgF$ DnS, 16 points</td>
</tr>
<tr>
<td>193</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type K $dgF$ DnS, 16 points</td>
</tr>
<tr>
<td></td>
<td>Description</td>
</tr>
<tr>
<td>---</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>194</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type T dgF DnS, 16 points</td>
</tr>
<tr>
<td>195</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type E dgF DnS, 16 points</td>
</tr>
<tr>
<td>200</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type J dgC UpS, 16 points</td>
</tr>
<tr>
<td>201</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type K dgC UpS, 16 points</td>
</tr>
<tr>
<td>201</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type T dgC UpS, 16 points</td>
</tr>
<tr>
<td>202</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type E dgC UpS, 16 points</td>
</tr>
<tr>
<td>208</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type J dgF UpS, 16 points</td>
</tr>
<tr>
<td>209</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type K dgF UpS, 16 points</td>
</tr>
<tr>
<td>210</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type T dgF UpS, 16 points</td>
</tr>
<tr>
<td>211</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type E dgF UpS, 16 points</td>
</tr>
<tr>
<td>216</td>
<td>3614/E Supv Discrete Output, 24 V, OFF STATE SCD, 8 points</td>
</tr>
<tr>
<td>217</td>
<td>3614/E Supv Discrete Output, 24 V, 8 points</td>
</tr>
<tr>
<td>218</td>
<td>3617/E Supv Discrete Output, 48 V, OFF STATE SCD, 8 points</td>
</tr>
<tr>
<td>219</td>
<td>3617/E Supv Discrete Output, 48 V, 8 points</td>
</tr>
<tr>
<td>220</td>
<td>3613/E Supv Discrete Output, 120 V, OFF STATE SCD, 8 points</td>
</tr>
<tr>
<td>221</td>
<td>3613/E Supv Discrete Output, 120 V, 8 points</td>
</tr>
<tr>
<td>222</td>
<td>3615/E Supv Disc Output, 24 V, OFF STATE SCD, Low Power, 8 points</td>
</tr>
<tr>
<td>223</td>
<td>3615/E Supv Disc Output, 24 V, Low Power, 8 points</td>
</tr>
<tr>
<td>224</td>
<td>3564 Single Discrete Input, 24 V, 64 points</td>
</tr>
<tr>
<td>225</td>
<td>3564 Single Discrete Input, 24 V, 64 points, Non-Critical</td>
</tr>
<tr>
<td>226</td>
<td>3562 Single Discrete Input, 48 V, 64 points</td>
</tr>
<tr>
<td>227</td>
<td>3562 Single Discrete Input, 48 V, 64 points, Non-Critical</td>
</tr>
<tr>
<td>228</td>
<td>3561 Single Discrete Input, 120V, 64 points</td>
</tr>
<tr>
<td>229</td>
<td>3561 Single Discrete Input, 120V, 64 points, Non-Critical</td>
</tr>
<tr>
<td>230</td>
<td>356X Single Discrete Input, 115V, 64 points</td>
</tr>
<tr>
<td>231</td>
<td>356X Single Discrete Input, 115V, 64 points, Non-Critical</td>
</tr>
<tr>
<td>232</td>
<td>3624/N Supervised Discrete Output, 24 V, 16 points</td>
</tr>
<tr>
<td>233</td>
<td>3624 Supervised Discrete Output, 24 V, 16 points, Non-Supervised</td>
</tr>
<tr>
<td>234</td>
<td>3627 Supervised Discrete Output, 48 V, 16 points, Non-Supervised</td>
</tr>
<tr>
<td>235</td>
<td>3627 Supervised Discrete Output, 48 V, 16 points, Non-Supervised</td>
</tr>
<tr>
<td>236</td>
<td>3623/T/TN Supervised Discrete Output, 120V, 16 points</td>
</tr>
<tr>
<td>237</td>
<td>3623 Supervised Discrete Output, 120V, 16 points, Non-Supervised</td>
</tr>
<tr>
<td>Line</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>238</td>
<td>362X Supervised Discrete Output, 115V, 16 points</td>
</tr>
<tr>
<td>239</td>
<td>362X Supervised Discrete Output, 115V, 16 points, Non-Supervised</td>
</tr>
<tr>
<td>208</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type J dgF UpS, 16 points</td>
</tr>
<tr>
<td>209</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type K dgF UpS, 16 points</td>
</tr>
<tr>
<td>210</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type T dgF UpS, 16 points</td>
</tr>
<tr>
<td>211</td>
<td>3708/E/EN Enh Isol Thermocouple Input Type E dgF UpS, 16 points</td>
</tr>
<tr>
<td>216</td>
<td>3614/E Supv Discrete Output, 24 V, OFF STATE SCD, 8 points</td>
</tr>
<tr>
<td>217</td>
<td>3614/E Supv Discrete Output, 24 V, 8 points</td>
</tr>
<tr>
<td>218</td>
<td>3617/E Supv Discrete Output, 48 V, OFF STATE SCD, 8 points</td>
</tr>
<tr>
<td>219</td>
<td>3617/E Supv Discrete Output, 48 V, 8 points</td>
</tr>
<tr>
<td>220</td>
<td>3613/E Supv Discrete Output, 120 V, OFF STATE SCD, 8 points</td>
</tr>
<tr>
<td>221</td>
<td>3613/E Supv Discrete Output, 120 V, 8 points</td>
</tr>
<tr>
<td>222</td>
<td>3615/E Supv Disc Output, 24 V, OFF STATE SCD, Low Power, 8 points</td>
</tr>
<tr>
<td>223</td>
<td>3615/E Supv Disc Output, 24 V, Low Power, 8 points</td>
</tr>
<tr>
<td>224</td>
<td>3564 Single Discrete Input, 24 V, 64 points</td>
</tr>
<tr>
<td>225</td>
<td>3564 Single Discrete Input, 24 V, 64 points, Non-Critical</td>
</tr>
<tr>
<td>226</td>
<td>3562 Single Discrete Input, 48 V, 64 points</td>
</tr>
<tr>
<td>227</td>
<td>3562 Single Discrete Input, 48 V, 64 points, Non-Critical</td>
</tr>
<tr>
<td>228</td>
<td>3561 Single Discrete Input, 120V, 64 points</td>
</tr>
<tr>
<td>229</td>
<td>3561 Single Discrete Input, 120V, 64 points, Non-Critical</td>
</tr>
<tr>
<td>230</td>
<td>356X Single Discrete Input, 115V, 64 points</td>
</tr>
<tr>
<td>255</td>
<td>3008/N Tricon Enhanced Main Processor</td>
</tr>
</tbody>
</table>
Bibliography


About the Authors

Alessandro Di Pinto (@adipinto) is a Security Researcher at Nozomi Networks. He previously worked as Penetration Tester and Reverse Engineer at Emaze Networks and then continued his interest in advanced malware analysis at Symantec as Sr. Threat Analysis Engineer. During his career he has obtained the Offensive Security Certified Professional (OSCP) and the GIAC Reverse Engineering Malware (GREM) certifications.

Younes Drago (@br4zzor), is a Security Researcher at Nozomi Networks and has a Bachelor of Science degree in Security of Computer Systems and Networks. Younes is an enthusiastic white hat hacker and a member of the Global Shapers Community of the World Economic Forum.

Andrea Carcano (@andreacarcano) co-founded Nozomi Networks in 2013 and is the Chief Product Officer (CPO). He is an expert and international leader in industrial network security, artificial intelligence and machine learning. He received his Ph.D. in Computer Science in 2012 and during this program he collaborated with international research groups in the energy industry. From 2011 to 2013 he was a Sr. Security Engineer in Eni Spa, with responsibility for secure connectivity between critical infrastructure and business networks.

About Nozomi Networks

Nozomi Networks is the leader of industrial cyber security, delivering the best solution for real-time visibility to manage cyber risk and improve resilience for industrial operations. With one solution, customers gain advanced cyber security, improved operational reliability and easy IT/OT integration. Innovating the use of artificial intelligence, the company helps the largest industrial facilities around the world See and Secure™ their critical industrial control networks. Today Nozomi Networks supports over a quarter of a million devices in sectors such as critical infrastructure, energy, manufacturing, mining, transportation and utilities, making it possible to tackle escalating cyber risks to operational networks (OT).