A Rising Tide: Design Exploits in Industrial Control Systems

Alexander Bolshev
IOActive, Inc.
Madrid, Spain

Jason Larsen
IOActive, Inc.
Seattle, WA 98104, USA

Marina Krotofil
Honeywell
Duluth, GA 30097, USA

Reid Wightman
Digital Bond
Indianapolis, IN 46220 USA

Abstract

Security is an emergent property. It is the outcome of an interaction between many sub-components and processes. One of the biggest problems of ICS security today is that systems undergo security assessments without recognizing the environment in which they are used. This has led to a situation where many systems have undergone cyber security assessments without addressing the ICS component, leading to a false sense of security. While Industrial Control System (ICS) vulnerability researchers and vendors became quite practiced at finding and fixing implementation bugs, many have minimal experience with design bugs. In the near future, we can expect the attacker community to leverage this weakness, as it did with earlier technologies. Therefore, ICS vendors must begin focusing better on the design of the environment and protocols, and ICS audits must begin now to focus on design. This paper a joint effort of the authors who independently researched design vulnerabilities in ICS with the goal of attracting more attention to ICS-specific design vulnerabilities.

1 Introduction

Implementing security in the Industrial Control Systems (ICS) space is a complex task. For the past two decades, the traditional information security space has made massive improvements in both offensive and defensive security. Yet, the systems in the ICS space have remained largely unchanged. This was in part due to a lack of access to industrial equipment and applications by security researchers.

The situation has started to change recently. The past five years have seen many more security researchers getting involved in ICS security. However, the majority of their findings have focused on the IT security bugs that occur in ICS components. The sorts of issues found by these researchers tend to be implementation bugs, such as protocol parsing issues, authentication bypass techniques, etc.

There has been a growing disconnect between the offensive and defensive security communities. While Industrial Control System vendors and researchers have become quite practiced at finding and fixing aforementioned implementation bugs, many have minimal experience with design flaws. In the near future, we can expect the attacker community to leverage this weakness, as it did with earlier technologies. Therefore, ICS vendors must begin focusing better on the design of the environment and protocols, and ICS audits must begin now to focus on design. To understand, consider the differences between implementation bugs and design flaws.

Buffer overflows, SQL injection, and cross-site scripting all fall into a broad class of bugs known as implementation bugs. A programmer who hunts for implementation bugs in her code relies heavily on automated tools. Various broken messages are fed into a tool until a malfunction is detected. The bug hunter then investigates the cause of the malfunction and looks for an exploitable condition. The programmer may then develop and release a patch, which typically corrects the bug by adding a check or verification, and life goes on.

A buffer overflow is an example of an implementation bug type that has been a staple of exploits for more than two decades. Programming errors allow the attacker to corrupt memory and eventually execute code. The major drawback to this exploit is that corrupting memory can crash an application. As weaponization techniques become more exotic, the risk grows that something will go wrong and crash the application. Attackers who write exploits for these implementation bugs often spend more time manipulating the state of the program and getting the program to leak key values than finding the bugs – in most cases, much more time.

"Design flaws" on the other hand are baked into the basic architecture of the protocol or device architecture. Therefore, design flaws are often more difficult to find

and fix. Design bugs hunters take a different approach. They analyze the normally allowed functionality of the program or device and ask questions like "What is this and how can I use it?" and "Can this be used in some other way or some other order?" Design bugs are often unique to the situation. Therefore hunting design bugs is very time consuming and cannot be automated.

In order to fix a design flaw, the developer needs to redesign the system in some way, which can require much more effort than issuing a simple patch. Fixing a design bug in a protocol usually involves changing that protocol which defines the way one or more components talk to each other and track state. When the protocol changes, every piece of software that implements that protocol needs to be changed at the same time. The fix may not be backwards compatible with an older version. If the attacker can simply speak the vulnerable version, having a fixed version does not really help.

The attacker's payback for design flaws is that they work across vendors and versions of software. Design bugs are generally carried into the future. Microsoft recently disclosed a bug in its Kerberos implementation that works against all versions of Windows since Windows 95. If the bug had remained undiscovered, attackers would have continued using it for years to come. Likewise, an attacker can use the shellshock bug to take over thousands of machines in millions of ways. If not discovered, that vulnerability would have been propagated into future versions of all vendor software that used that design.

The topic on engineering safe and secure systems is not new, with a most notable work being by Levenson [16]. Understanding that further neglect of the systems design security is unacceptable, the standardization bodies have initiated work in this direction. Thus, most recently NIST has published a guide to engineering-driven approaches necessary to develop more defensible and survivable systemsincluding the components that compose and the services that depend on those systems [22].

The defensive community has started investing in technologies such as data diodes and whitelisting software while the offensive community has embraced firmware rootkits and exotic attack pathways. The research into ICS security is similarly disconnected where defensive technologies are not matched to offensive research. This paper is a joint effort of the authors who independently researched into design weaknesses in ICS with the goal to attract more attention to ICS-specific design flaws. The design flaws for this paper were selected in a way to represent different design vulnerabilities at the different layers of cyber-physical systems and are arranged in order of least cyber-centric to most. Note, that this paper is not intended to be a primer on ICS hacking and the readers are expected to be familiar with the basics of process control and associated terminology.

2 Cyber-Physical Exploitation

Cyber-physical systems (CPS) are physical and engineered systems whose operations are monitored, coordinated, controlled and integrated by a computing and communication core. Such systems use computations and communication deeply embedded in and interacting with physical processes to add new capabilities to physical systems. These cyber-physical systems range from miniscule (pace makers) to large-scale ICS systems such as power-grids. Fig. 1 illustrates logical layers of the CPS. The physical layer defines the relationship between field instrumentation (sensors and actuators) with the physics of the process. The control layer in general consists of the control logic and control algorithm. Control logic defines the logical execution of the process (what should happen, when, in which sequence and under which conditions). It is also defines the way how the operators and control equipment observe the state of the process and take control actions. The control algorithm describes the exact nature of the control output as a function of the input. The inputs to the algorithm are the sensor signals and other variables defined by the control logic (e.g. results of internal computations). The output from the algorithm is the control signal (e.g. voltage) sent to the actuators. The cyber layer corresponds to the design of the firmware/software and communication protocols of the supporting infrastructure required to carry out process control tasks. Exploitable vulnerabilities can exist at each of the CPS layers.

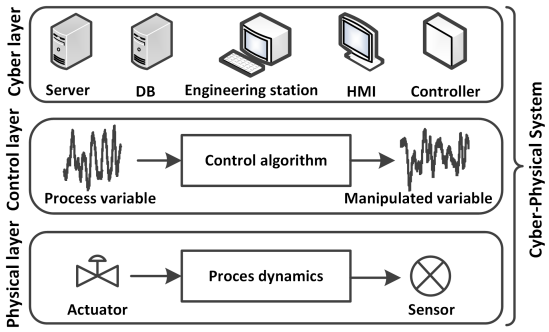


Figure 1: Logical layers of ICS

Once access is gained to a control network, an attack still needs to be performed. This is where the public literature falls short. E.g. in the ICS adoption of the Lockheed Martin cyber kill chain, "Attack Development and Tuning" is presented as a single stage [3]. In practical ICS exploitation this kill chain phase consists of own

stages [14, 11] which includes discovery of vulnerabilities, specific to target ICS environment.

The fundamental flaw in modern defensive computer security is the notion that a PC is only a single computer running a single operating system. In the past decade computer architecture has undergone tremendous update. Thus, nearly every hardware component that used to be "dumb" has been replaced with a "smart" component [10]. For example network cards now have built-in web servers and perform complex cryptographic tasks. The upside for the attacker is that these embedded computers lack almost all of the protections built into modern operating systems. It is currently less labor intensive to write and maintain a rootkit for a piece of firmware [28] than to maintain the same rootkit functionality in the main operating system.

Industrial automation has followed the IT hardware development path. What used to be a simple analog sensor is now an IP-enabled smart transmitter with multiple wired and wireless communication modes, a large number of configuration possibilities, and even a webserver so that maintenance staff can calibrate and setup the device without approaching it [19]. The sensors used in ICS are becoming cheaper and more distributed than ever before. Tank farms are placed in safer locations away from the main process. Weather sensors are distributed outside the fence. Predictive maintenance systems are being integrated into assets that were previously only mechanical machines. The attacker community is already investigating how to gain greater control of the full process from these outlying sensors. In [7] the authors use design and implementation bugs in an industrial HART protocol to access bugs in the traditional IT infrastructure. In ICS security this is sometimes referred to as "hacking upstream". It is now possible to initiate attack at the lowest levels of ICS (Level 0 - Level 1 of the reference architecture [9]) and execute it there, without ever reaching traditional IT networks and systems located at the higher levels of ICS.

In early exploitation of CPS, it was common to focus on payloads that required very little time to execute. This limited the attacker to targets where damage could happen suddenly and there would not be time to react. Many attacks take time and need to wait for a process to be in a particular state before they can be executed. When an attack starts, the attacker must notify the various parts of his apparatus that it is time to suppress alarms and override safety values. This creates race conditions between attack and process physics. The solution is to decouple the payload of the attack at any place where propagation delay might be a problem and break the attack into stages. Each stage needs its own process state detection routines and its own final payload. The detection routines can be used to infer the state of the upstream payload. This approach mitigates the race conditions and reduces the complexity of each stage to a point that can be tested and managed.

Decoupling stages of the attack requires finding exploitable attack scenarios in different parts of the process. Moreover, the attacker needs to include several attack instances into her final attack payload in order to increase chances of success in the complex environments such as ICS.

It is important to note that cyber-physical exploitation is not necessary achieved by delivering malicious bits over electronic media. Consider an electrical analog motor. Is it not hackable in the traditional way as it does not have a computational element in it. However, the motor configuration and testing engineers use Digital Maintenance and Test Equipment (M&TE). Configuration, calibration and diagnostic equipment run firmware, which is exploitable. Thus, it is possible to place malicous code on M&TE responsible for the motor test. This code could be created so as to report a bad motor as being good and/or a good motor as being bad [26]. As a result a facility may incur replacement costs before the end of useful life, or downtime costs from an unhealthy motor failing spontaneously during operation.

3 Design Exploits in ICS

Many implementation bugs follow distinct patterns that can be tested. Buffer overflows can be found by systematically sending broken messages to a target. SQL injection can be found by sending data with active SQL fragments. Design flaws are often a result of the specific conditions and cannot be tested systematically. However, once discovered, they can be applied to a wide range of processes with similar equipment or environments.

The primary thing the attackers break are assumptions. The attacker manipulate how a system really works, not how it is supposed to work. It many cases the attacker does not know the system. And by learning the system, she tends to discover how the system fails, redefining failure as a success condition. While initial design of any system aims at ensuring efficient and safe operations under occasional failures, security makes system engineers rethink design decisions to address the possibility of intentional failures. Specifically, ensuring secure control of operations requires the engineer to consider all the possibilities, such as how control decisions are met and to address design as a system security problem.

3.1 Physical Layer: Exploiting Analog-to-Digital Converters

In cyber-physical systems data originates in the physical space and the concerns about data reliability/integrity

starts from the first point a measurement being taken by the sensor (before the data becomes part of the communication infrastructure). A common design in sensors is to vary the voltage or amperage in response to a measured physical phenomenon (e.g. pressure or flow). The analog signal is then converted into a digital number by an analog-to-digital converter (ADC), before finally being supplied to a control algorithm. Each ADC has an analog reference voltage V_{ref} or current against which the analog input is compared. Therefore, the digital output word tells what fraction of the reference voltage or current is the input voltage or current.

In general, analog-to-digital conversion happens in two steps: (1) sampling and holding S/H and (2) quantization and encoding (Fig. 2). S/H circuit captures the voltage of a continuously varying analog signal and holds its value at a constant level for a specified minimum period of time. The quantization refers to the partitioning of the reference signal range into a number of discrete quanta, then matching the input signal to the correct quantum. During the encoding a digital code (discrete value) will be assigned to the input signal. A practical ADC cannot make an instantaneous conversion (typically > 10 clock cycles). The conversion time depends on the ADC's design and configuration, and on the ADC's clock frequency.

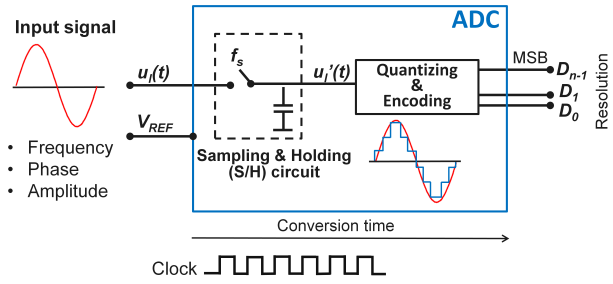


Figure 2: Analog-to-digital conversion process

There are many different ADC designs available at the market [5]. In our work we analyzed Successive-Approximation Register (SAR) and Delta-Sigma (Δ-Σ) ADC design as these types of ADCs are most widely used in industrial applications. SAR is the architecture of choice for nearly all multiplexed data acquisition systems (e.g. PLCs), as well as many instrumentation applications. Δ-Σ ADC is a choice for modern precision measurement instrumentation (e.g. temperature sensors), energy-monitoring and motor-control applications.

In a number of real world environments we observed architectures in which several devices are connected via an analog line in series as depicted in Fig. 3. A PLC responsible for the regulatory control sends an analog control command to the actuator (e.g. a motor) and a so called "safety PLCs" performs sanity checks of that command for unwanted conditions. Among other device we have seen connected to the line were DAQ (Data Acquisition) equipment for data logging and visualization purposes (e.g. on HMI) and other types of equipment.

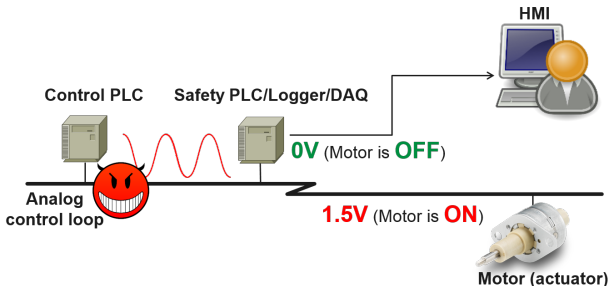


Figure 3: Threat scenario

Every device connected to the analog line will have inbuilt ADCs. It is expected by the engineers that the ADCs on all devices which consume the same analog signal will convert it into the same digital number. However, depending on the configuration of the ADCs in the control loop, this expectation might not always hold true. Our experiments have shown that it is possible to generate such an analog signal (e.g. from the control PLC) that it will be converted into different digital values by different ADCs. E.g. as shown in Fig. 3, the actuator converts analog signal into 1.5 V (ON command) and the loggers ADC outputs 0 V (OFF command).

For illustration, we setup an experimental architecture as depicted in Fig. 4 [6]. An Arduino board plays role of the control PLC. It generates analog manipulated variable (MV) using PWM (pulse-width modulation) which determines the speed of the motor. The generated MV is monitored with S7-1200 PLC which plays role of the "safety PLC" and also sends the acquired value to the web HMI interface (on smartphone). In the first part of the experiment, the Arduino generates constant MV value of 3.7 V, which is displayed on the HMI panel. In the second part of the experiment, we modified the amplitude and the frequency of the supplied signal. Specifically, we strategically decreased signal voltage to 0.5 V at the regular intervals (about 10% of the time) and tuned the frequency of the signal. The modified signal caused motor to shake severely while the HMI kept outputting normal value of ≈ 3.7 V. If such an attack is successfully executed on the plant floor, operator will lose awareness of the true state of the process and might make wrong and potentially harmful control decisions. Similarly the safety protection may not engage due to wrong notion about process state. In the ICS world such situations are often referred to as "loss of view" and "loss of control".

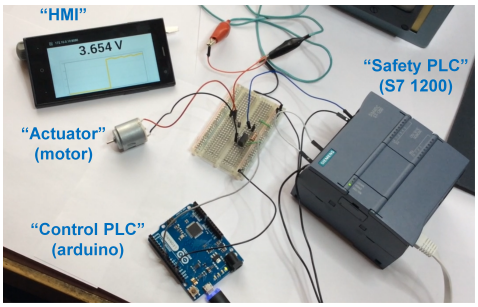


Figure 4: Experimental setup

Fig. 5 illustrates the impact of the signal frequency on the sampling process of two different ADCs. As can be seen, due to different configurations of the ADC, the result of the analog-to-digital conversion for these ADCs will differ substantially. "Secure" hardware design and configuration of the ADC circuits would help to eliminate the attacks presented in this section.

3.1.1 Vulnerabilities

Every ADC has its physical limitations. This results in deviations from the theoretically perfect conversion of the input analog signal referred to as distortions. Understanding ADC's limitations may allow an attacker to impact the accuracy of analog-to-digital conversion to her advantages. The exploitation conditions arise from the design flaws and misconfiguration of equipment connected to the analog line. In our work we took advantage of the sampling frequency and dynamic range of the ADC.

Sampling frequency. It is imperative that an ADCs sample time is fast enough to capture essential changes in the analog waveform. The highest-frequency waveform that an ADC can theoretically capture is called Nyquist frequency. It is equal to one-half of the ADCs (current) sample frequency. Thus, if an ADC circuit has a sample frequency of $5kHz$, the highest frequency waveform it can successfully resolve will be the Nyquist frequency of $2.5kHz$. If an ADC is subjected to an analog input signal whose frequency exceeds the Nyquist frequency for that ADC, the converter will output a digitized signal of falsely low frequency (Fig. 6a). This phenomenon is known as aliasing.

Note that simply obeying Nyquist rate is insufficient. The phase of the signal is important too. If in Fig. 6b the cosine wave is replaced with sine function, a sampling rate of $f_s = 2.0 * f$ would lead to no signal. The phase difference between sine and cosine would lead the sine being sampled in the zero-crossings (but the cosine is sampled at its peaks). This is one of the exploitable conditions.

Dynamic range. An ADC's dynamic range is the range of signal amplitudes which the ADC can resolve. If an input goes outside of the valid voltage range (but still within allowable electrical ratings), the ADC will saturate, feeding out the maximum or minimum resolution bit values (the ADC will clip the amplitude). If an ADC does not have a special output bit flagging the saturation condition, the clipping of the input signal may go unnoticed. The out of range signal may cause disturbances on the adjacent analog inputs as well. If the input signal exceed electrical ratings of the ADC, damage to the ADC is likely to occur.

A large number of scenarios and exploitable conditions were discovered when assessing security of the analog-to-digital conversion [6]. Both SAR [4, 20] and Δ-Σ [2] ADCs were vulnerable to malicious analog signal attacks. In certain cases, the exploitable conditions can be directly inferred from ADC datasheet review or reliably estimated in a lab environment (on the mock up architecture). In other cases the estimation of the attack parameters must be carried out on the live process (e.g. from the Ethernet switch).

ICS environment is especially vulnerable to the discovered attacks. Most of the industrial processes are sampled at a low rate. In ICS networks, it is rare to meet conversion frequencies more than 1kHz. Most MCUs inside transmitters and actuators are capable of generating arbitrary signals up to 500-1000 Hz. However, some devices allow generation of signals of up to 100kHz [18] which allows to generate a smooth sine wave of $\approx$ 5kHz. This is more than enough to craft the described attacks in typical ICS network.

3.1.2 Impact

The "Never trust your inputs" rule is well known in security and poor input validation is one of the most common types of attacks on any system. An attacker sends data that is in some way ill formed relative to what the receiving application expects, for example, sending a too large message, an undefined function code, or a time stamp that is much earlier or later than the actual time. Ensuring that all data is validated, i.e. meets expectations, before being further processed is an effective way to mitigate such attacks. Validation checks verify that data received is well-formed, meets design expectations, and makes sense given local context. This requirement is true for all types of systems.

In the ICS context, input validation refers to the content of the data (as opposed to its format). By tampering with the input analog signal, the attacker may cause an ADC to output a spurious process data stream, inflicting negative impact further down on the data chain and

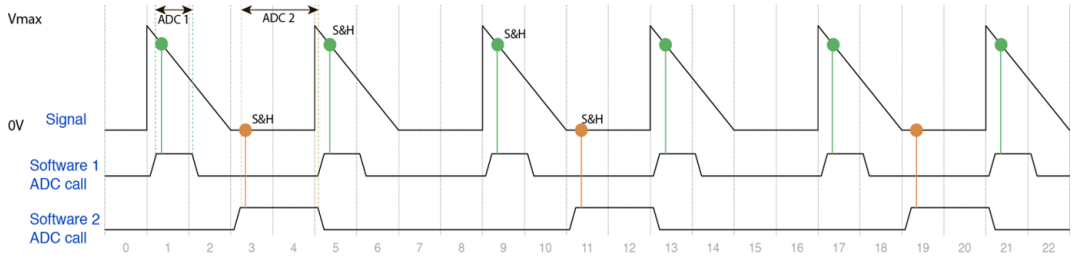
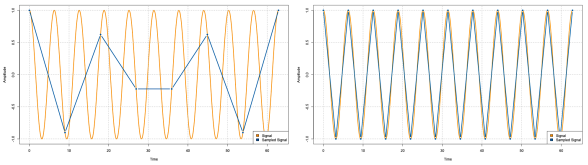


Figure 5: Impact of the signal frequency on the sampling process of the ADC



(a) $f_s = 1.5 * f$ (b) $f_s = 2.0 * f$

Figure 6: Impact of f_s on the resulted sampled signal

yielding loss of process view. When false data is submitted to the communication protocol stack on the device, false data will be delivered securely to the intended application. Therefore in ICS domain data security cannot be ensured by purely IT security means and must be supplemented with the engineering process-aware approaches.

Further, it is possible to create exploitable conditions on the device which consumes the malicious analog signal. Consider a 5–10 V signal which is fed into the ADC with 0–15 V range. Improper handling of signals lower than 5 V or higher than 10 V by the ADC may trigger a software vulnerability in the firmware. During a code review we have observed the following input signal handling routine:

```
uint8_t val = readADC(0);
//reading 8-bit ADC value with ranges 0V-15V
val = val - 85;
//normalization -> 85 == 5 V (255/3)
```

Any signal of the amplitude less than 5 V ($val < 85$) will cause integer overflow in val.

3.1.3 Mitigations

The cost of the tool kit required to launch the attack with a physical access to the analog line is $50 (1kHz) – $400 (50MHz). Limiting physical access to the analog line is effective against such attacks (nothing new). Notably, the authors recently reviewed several risk assessment reports from the industry in which asset owners ranked physical access threat as very plausible.

Hardware mitigations. Low-pass filter (LPF) attenuates signals with a frequency higher than its cut-off frequency f_c. Preceding ADC with a low-pass filter such that $f_s > f_c$ is a known approach to a good design in digital systems in order to mitigate the aliasing effect. Yet, in the reference designs of the industrial control input modules specifically suggested for the usage in process control programmable logic controllers (PLCs) and distributed control system (DCS) we find ADC with $f_s > 470Hz$ buffered with a low pass filter with cut off frequency $f_c = \approx 15kHz$ [1]. To remedy the defects of improper device design, one can simply connect a capacitor or an LPF to the terminals of the PLC. However, it is crucial to ensure that all devices connected to the same analog line have the same cut-off frequencies. E.g. if a PLC input is buffered with LPF $f_c = 1kHz$ and an actuator equipped with LPF $f_c = 5kHz$, the attack is not only possible, but the probability of success increases.

To avoid attacks against the dynamic range of the ADC, one should normalize the input signal amplitude before conversion. It could be done with a voltage divider, OpAmp, signal conditioning scheme or even dynamic range compression (depending on what is most suitable for the specific operational environment).

Using ADC with higher sampling frequency can mitigate many of the described attacks as the attacker will have to generate a signal of much higher frequency. Generating > 1 MHz signal and injecting it into an analog line is much harder than injecting < 1 MHz signal. High frequency signals are subjected to greater attenuation and are more affected by noise.

Software mitigations. Certain randomness in the sampling frequency will make attackers job much harder and many of the discussed attacks will be much more challenging to execute. Varying Sample Frequency [21] is recognized as an effective approach to deal with with the uncertainties in data sets [25] and is proved to be more effective for operational processes than constant sampling frequencies. A small variation in $f_s = f + rand(\Delta)$ will not degrade signal understanding process.

On the contrary, it will produce a signal sample of better quality. Alternatively, the signal can be sampled with the highest f_s of the ADC and the converted values should be averaged over a short time interval.

Last but not least, the ADC should never be put into sleep as shown in the code below (observed during the real-world code review).

```
Val = readADC();
Output(Val);
Sleep(Timeout);
```

Each time interval when the ADC does not sample the analog line creates a window of opportunity for the attacker to inject a malicious analog signal harmful to the other equipment connected to the same analog line.

3.2 Control Layer: Exploiting Variable Frequency Drives

Motors are one of the most widely used pieces of equipment in the industry, with typical applications being pumps, ventilators, compressors, belt conveyors, rolling mills and in different machines used in manufacturing and other industries. Most motors are designed to operate at a constant speed and provide a constant output. However, modern technology requires different speeds in many applications where electric motors are used. A variable frequency (speed) drive or VFD is a most common piece of equipment that regulates the speed and rotational force of an electric motor. The common architecture of VFD usage is presented in Fig. 7.

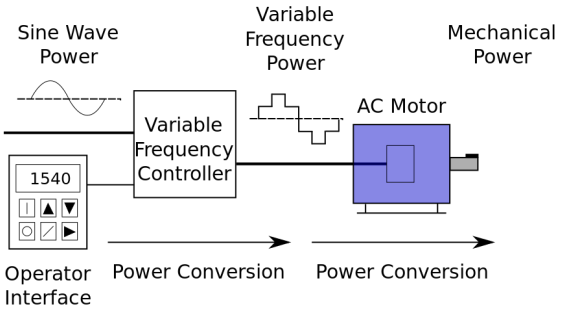


Figure 7: Common architecture of VFD usage

One of the biggest enemies of rotating equipment such as motors are vibrations. Vibrations can result from a number of conditions such as misalignment, looseness, wear and others, acting alone or in combination. Vibrations cause machinery to consume excessive power and may damage product quality. In the worst cases, vibrations can destroy bearings and ruin shafts leading to equipment failure and disruption of production. Collaterally, resonant machine components and supporting structures can magnify even small vibration problems enough to damage auxiliary equipment. Equipment damage due to vibrations was implemented in the widely-known Stuxnet attack [13].

Skip Bands. All rotating shafts, from motorcycles to industrial pumps, have mechanical resonance points. These are the frequency points (critical speeds) at which vibration can rapidly damage the equipment [8]. Equipment designers and operators are aware of the mechanical resonant frequencies and avoid them by either accelerating beyond or decelerating below them so the motor does not run at dangerous frequencies. Most VFD drives offer multiple skip frequency parameters to mitigate different resonance points. The VFD will not allow operators to run the equipment at its critical speed. If the motor must achieve a target running speed which is above the skip band, many VFDs will also power through such speeds by allowing the motor to briefly draw additional current.

3.2.1 Vulnerability

The first design flaw regarding skip bands is that many VFDs allow any remote user, including malicious users, to both read and modify the skip bands without authentication (Fig. 8). By leaving these values readable and writable, the VFD provides a recipe that an attacker may use to cause damage to equipment.

Code	Name	Logic address	Access	Type
TDC1	IDC injection time	16#28A2 = 10402	R/W	UINT (Unsigned16)
JOG	Jog assignment	16#2B66 = 11110	R/WS	WORD (Enumeration)
JGF	Jog frequency	16#2B67 = 11111	R/W	UINT (Unsigned16)
PS2	2 preset speeds	16#2C89 = 11401	R/WS	WORD (Enumeration)
PS4	4 preset speeds	16#2C8A = 11402	R/WS	WORD (Enumeration)
PS8	8 preset speeds	16#2C8B = 11403	R/WS	WORD (Enumeration)
SP2	Preset speed 2	16#2C92 = 11410	R/W	UINT (Unsigned16)
SP3	Preset speed 3	16#2C93 = 11411	R/W	UINT (Unsigned16)
SP4	Preset speed 4	16#2C94 = 11412	R/W	UINT (Unsigned16)
SP5	Preset speed 5	16#2C95 = 11413	R/W	UINT (Unsigned16)
SP6	Preset speed 6	16#2C96 = 11414	R/W	UINT (Unsigned16)
SP7	Preset speed 7	16#2C97 = 11415	R/W	UINT (Unsigned16)
SP8	Preset speed 8	16#2C98 = 11416	R/W	UINT (Unsigned16)
JPF	Skip frequency	16#2C25 = 11301	R/W	UINT (Unsigned16)
PIF	PID : PI function feedback assignment	16#2E7D = 11901	R/WS	WORD (Enumeration)

Figure 8: Configuration of VFD

The attacker strategy for causing vibration damage is quite simple: read the skip band information from the VFD, modify the skip band value to a new range, and finally set the output frequency of the drive to the center of the previously-observed skip band. Many VFDs allow this complete set of changes to be made *while the motor is still running*. The drive shaft, or possibly external housing for the drive, should begin vibrating at its natural frequency.

An experiment was performed using a Schneider Electric Altivar 12 VFD [23], a WEG 3-phase 240 Volt AC electric motor, and a 24-inch 5/8 diameter external drive shaft. The setup was observed to have a critical speed

of 933 RPMs, centered at an output frequency of 16 Hz from the VFD. At this frequency, the drive, shaft, and housing vibrated severely.

The Schneider VFD was configured with a skip band at 16 Hz and operated on a network with other industrial control equipment. An attack script was generated to read the Schneider VFD Skip Speed setting, modify it, and change the output of the drive to run at this speed. The script succeeded in changing the drive speed. On a larger motor, with sensitive equipment nearby, this change could result in damage to plant equipment.

A documentation review of other VFDs was performed on three additional manufacturers. Vacon and ABB VFDs appear to be vulnerable to the issue directly, meaning that no authentication is required to perform the attack. Allen-Bradley VFDs do require a passcode to change the skip band setting and the current drive setting, however the drive documentation indicates that an attacker may recover the required passcode thanks to a factory backdoor.

3.2.2 Impact

The side-effect of readable and writable parameters in VFDs is that an attacker may easily mask changes in the operating speed from the operators. VFDs calculate the drive speed for HMI in RPMs by using the nominal speed (referred to as the Case Speed) and nominal frequency (referred to as the Case Frequency) of the drive. The motors current speed in RPMs is calculated via the equation:

$$\frac{CaseSpeed(RPMs)}{CaseFrq(Hz)} \times OutputFrq(Hz) = CurrentSpeed(RPMs).$$

An attacker may change these nominal values such that the dangerous output speed appears to be the normal operating speed for the drive. Thus, if the attacker raises/lowers *CaseSpeed* RPM setting, the VFD will decrease/increase *OutputFreq* to maintain RPMs. As a result the drive will slow down/speed up. If the operators do not monitor or log the drives output frequency, and instead only monitor the drives output speed, they are unlikely to notice the change in speed until other changes in the process occur.

In addition, VFDs tested only support integrity-free protocols such as Modbus and Ethernet/IP. These protocols are subject to network-based data-tampering attacks, such as the use of a protocol VCR [27] to record and replay traffic in order to hide vibration attacks from operators.

The attacks outlined above may be easily automated. E.g. Skip Band attack against Schneiders Altivar VFDs requires less than 10 lines of program code using the popular Metasploit framework. This code may then be run against any networked Altivar VFD to induce vibration damage, assuming that the engineers configured the drive with the skip band setting. Similar attack scripts could be generated for other manufacturers VFDs with a similar amount of code. Importantly, scripts for many VFDs can be generated without access to the equipment. Only a process control map, available for free from vendors, is needed to write a script to launch such attacks.

This style of attack is likely to affect many controllers which are responsible for both control of a process component and protection of that component. VFDs are just one example of such controllers.

3.2.3 Mitigations

Adding adequate authentication and data integrity mechanisms to VFDs would provide the best protection against these types of attack. This is unlikely to happen for many reasons, especially compatibility with existing software. Implementing such a change would require many portions of the process control software system to be changed, including data historians, human-machine interfaces, and engineering software.

Another method to detect this type of attacks is to simply monitor the VFDs output frequency, as opposed to only monitoring the output speed of the motor. This will detect data tampering attacks which are implemented on the VFD itself, e.g. by modifying the motor *Case* parameters in the VFD. This mechanism does, of course, have the drawback that protocol-level attacks could go undetected.

Installation of an external motor tachometer could also help in mitigating this attack. Note that addition of a physical tachometer will require engineering changes to the process. For many installations, this may require installation of a new belt or a shaft to allow connection of the tachometer. Note also that tachometers are not themselves maintenance-free, and many tachometers have a maintenance interval that is considerably shorter than the motor which is to be monitored. Also, output of the tachometer will require an additional ADC or an entire PLC to monitor the data generated.

Finally, installing a vibration monitoring system on all VFD-controlled motors could help to detect the attack. From the conversation with a large predictive maintenance solutions provider, 85% of AC motors lack any sensors (tachometers, vibration sensors, or temperature sensors) for diagnostic of motor's health. If used, this mechanism may be most effective when the monitoring is performed out-of-band with the monitoring of the VFD itself: using a separate network decreases the likelihood that an attacker will spoof both the VFD output and the vibration monitoring system output simultaneously. The other indication of excessive vibrations are noticeable changes in heat (for example, at bearings), noise, or mo-

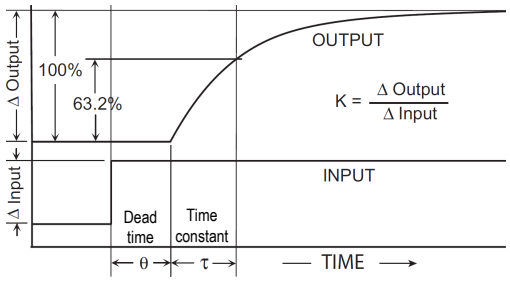


Figure 9: Time constants in process control [24]

tion when the operating frequency changes.

3.3 Cyber Layer: Exploiting Protocol Stack Implementation

The most common algorithm for controlling a feedback loop is the PID (Proportional, Integral, Derivative) algorithm. While the algorithm operates at the control layer, it makes assumptions about the underlying physical (process) and communications (cyber) layers. PID loops are generally tuned using observations from the actual equipment. Typically the controller is manually set to a value in the lower part of the expected operational range and allowed to stabilize. The controller is then manually set to a value in the upper part of the range and the change is observed. Two values are extracted from the observation: the dead time θ and the lag τ or time constant (Fig. 9). The dead time represents the delay between when the controller changes the actuator and the process actually responds. Time constant represents the rate at which the process responds once it starts to change. The PID algorithm assumes that the values are fairly constant during the operation of the process and acts accordingly.

PLCs, RTUs, aggregators, and other pieces of process control equipment often work on slightly out of date data. The control logic can cycle quickly and running through the whole control logic thousands of times a second is common. New data about the state of the process may arrive at a slower rate. For example an ADC converter may only be able to sample at hundreds of times a second. Smart sensors often only report changes of the process value when they have changed enough to be significant. This scheme is often referred to as "report by exception". Whatever the cause, the main loop of the controller usually does not stop waiting for new data to be made available. It uses the last known good process value in its calculations assuming that fresh data will arrive shortly.

The presence of stale data in the communications pathway of a PID loop can be used to invalidate the assumption that the dead time is a constant and cause the PID controller to push the process further out of control instead of in control [15]. The single value for dead time accounts for all the delays in the loop including both communications delay and actual physics delays. If the attacker can delay the messages without modifying them, the dead time for the loop will now be different. If the data delay is timed strategically, the PID controller causes the fluctuations in the process to be amplified instead of dampened. Eventually this will lead to a fault or abnormal shutdown of the process.

3.3.1 Vulnerability

During a test (vendor withheld), a design bug was found in the way the device internally handled session identifiers that allowed the attacker to achieve a stale data attack. The devices preferred method of communications was via an undocumented vendor proprietary protocol, but the device also supported a number of standard protocol such as DNP and IEC870. The RTU had expansion slots that could be populated with various Ethernet and Serial cards that supported the suite of protocols.

During the initial handshake of the vendor proprietary protocol, a session ID was assigned. For each session, the ID was predictably incremented by a fixed number. It was first noted that an attacker could determine how many new sessions were established on the other RTU interfaces by repeatedly establishing sessions and looking for missing IDs in the predicted sequence. After further investigation, it was noted that establishing a new standards-based protocol also caused a gap in the sequence as shown below.

```
Vendor Protocol Handshake --  Session ID 0x4000
Vendor Protocol Handshake --  Session ID 0x5000
Vendor Protocol Handshake --  Session ID 0x6000
Standards-Base Protocol Handshake
Vendor Protocol Handshake --  Session ID 0x8000
Vendor Protocol Handshake --  Session ID 0x9000
```

The sessions on the proprietary protocol were not tied to a transport layer session so a logical session could be started in one TCP stream and then continued via a different TCP stream or for that matter via serial communications. If an attacker crafted a message containing the ID of another session, a reply would be sent revealing the internal sequence numbers of that session. Additionally, if the attacker crafted an acknowledgment message with the ID of another session, he could advance the sequence numbers of other sessions causing valid datagrams to be ignored as repeats. Since the valid messages were silently dropped, the remote devices simply retransmitted the messages.

Communications could be restored by advancing the sequence number completely through the 32-bit number space back to its original value. In practice this could

be achieved in three messages. By strategically manipulating the sequence numbers of the other sessions, the attacker gains the ability to pause and then resume any of the other data streams flowing through the RTU.

This particular design bug is straightforward to understand and exploit, but none of the standard testing techniques would find it. The exploit works across all the versions and configurations of this RTU. Unless the vendor changes the way session management is done in the internal code, it is likely to continue to work in the future.

3.3.2 Impact

One of the key differences between a cyber attack and a cyber-physical attack is when the digital infrastructure goes down the physics part of the process keeps running. It has always been known that interruption of the data flowing on process control cables could be used to disrupt production at a facility. What was previously unknown was that targeted interruption of the data could be used to drive the process to an arbitrary state [12]. This gives rise to the possibility that an attacker with access to what was previously considered less critical could actively attack the process and is not restricted to nuisance disruption.

In the ICS world denial-of-service attack on a data stream can be used to manipulate the process at will, even when the attacker cannot modify the data stream due to cryptographic protections. Using nothing but delayed or dropped messages the attacker can influence the model a controller has about the current state of the process, and make it take wrong control decisions. E.g. if the controller is already compensating for a disturbance, it will continue to compensate and eventually overshoot the equilibrium point of the process. This attack scenario calls into question the effectiveness of many cryptographic protections proposed by industry vendors.

A number of examples can be formed where interfering with a data channel is possible but fully viewing and manipulating the data channel is not. The most common case is networking equipment where the attacker only has access to the administrative interface but not to the data plane. For example, in common routers there is a ”fast path” where packets are routed in custom designed application-specific integrated circuits (ASICs) and the ”slow path” where packets are transferred to the CPU. Access to the administrative interface can give code execution on the device, but only packets transiting the slow path can be observed and manipulated. If all packets are requested by the slow path, the CPU is saturated resulting in a DOS on the communications link. If packets are sampled periodically, they can be examined and with that allowing an attacker to perform a DoS attack at on opportune time.

This attack scenario shows one of the fundamental flaws of CPS defensive thinking. The physics of the process is important, the data that drives the physics is important, but the synchronization between the physics and the data is equally important. Modern protocol stacks are based on buffers, streams, and queues that do not always guarantee time synchronization.

3.3.3 Mitigation

The most common root cause of stale data is when the protocol stack was implemented in a stateless manner. The master maintains a set of timers for the data tied to individual data points or a set of data points assigned to a class. When data is received by exception the timer is restarted. If the data changes often enough, the master never sends a poll request. If the timer expires, the point is placed into a point list to be requested during the next poll. If many cases, the timer is reset when the poll request is sent with the assumption that the point value will be returned in the next reply. In other cases, the timer is only reset upon a successful receipt of the updated data.

Most process control protocols do not contain timestamps of when the observation took place. In the places where timestamps are available, they are often ignored. If the controller were made aware of the time delay in the message, it could be compared to the dead time and then rejected if the message is too stale.

4 Conclusions

The offensive community continues to evolve and research. The goal of this paper was to provide a flavor for what can be expected in the near future. The ICS field can take a lesson from the history of other technologies. A decade ago the attacker community started learning to hack cell phones and gaming consoles and was forced to learn about JTAG ports and encrypted bootloaders. Soon after, the number of attacks against embedded devices such as wireless routers and cars exploded. The attacker community took its new-found knowledge and applied it to many embedded products. Even today, every conference includes presentations on hacking systems with embedded microcontrollers. A good example is work by Lindner who developed disassembler for Siemens PLCs as a tutorial on how to approach the task of building custom disassemblers [17].

This pattern is highly likely to repeat for more ICS components and even complete systems of components. High-end control systems are at least partially delivered as source code and are modified for each individual customer. Attackers who need reliable exploitation across multiple targets are already using exotic techniques to deal with the uncertainty of the target envi-

ronment. However, the cost of testing for implementation bugs skyrockets as patches are produced by vendors, making the cost of finding design flaws look very reasonable. The skills required have so far kept a high barrier to entry, so real-world control system exploitation has remained in the hands of a few. But at some point, attackers will improve their skills and will start finding design flaws everywhere.

Recommendation. Audits for industrial control systems need to evolve to emphasize the actual design of the environment and protocols. The skill set needed to spot a design bug is very different than the skill set used to spot a buffer overflow. Therefore the industrial engineers must be included into testing teams.

In addition software and hardware vendors must start wrestling with the soundness of their basic systems design now, or else the barrier to entry for attackers may fall enough that we will see real-world exploitation of industrial processes by amateurs.

References

[1] ANALOG DEVICES . Circuit Design: Fully Isolate Input Module Based on the AD7793 24-Bit Σ-Δ ADC. `http://tinyurl.com/jdvu6jd`. Retreived: September 2015.

[2] ANALOG DEVICES. AD7705/AD7706 – Datasheet. `http://tinyurl.com/jkt6egr`, 2006. Retreived: March 2015.

[3] ASSANTE, M. J., AND LEE, R. The Industrial Control System Cyber Kill Chain. SANS Institute InfoSec Reading Room, Oct 2015.

[4] ATMEL. ATmega16U4/ATmega32U4 – Datasheet. `http://tinyurl.com/gs9lmdm`, 2016. Retreived: September 2015.

[5] BLACK, B. Analog-to-Digital Converter Architectures and Choices for System Design. *Analog Dialogue 33*, 8 (1999).

[6] BOLSHEV, A., AND KROTOFIL, M. Never Trust Your Inputs (or how to fool ADC). Black Hat Asia 2016. `http://tinyurl.com/zzz97tw`. Retreived: April 2016.

[7] BOLSHEV, A., AND MALINOVSKY, A. HART (In)Security. Zero Nights 2012. `http://tinyurl.com/pyqxgbl`. Retreived: April 2016.

[8] DE SWARDT, H. Critical Speed on an Electric Motor Explained. Tech. rep., Marthinusen & Coutts, 2007.

[9] DIDIER, P., MACIAS, F., HARSTAD, J., ANTHOLINE, R., ET AL. Converged Plantwide Ethernet (CPwE) Design and Implementation Guide. Cisco Systems and Rockwell Automation, 2011. `http://tinyurl.com/6a8gmch`. Retreived: April 2016.

[10] KAMINSKY, D. Yet Another Dan Kaminsky Talk (About much more than RNG). DefCon 2014. `http://tinyurl.com/mf4bqq2`. Retreived: April 2016.

[11] KROTOFIL, M., AND LARSEN, J. Rocking the Pocket Book: Hacking Chemical Plants for Competition and Extortion. DefCon, 2015. `http://tinyurl.com/jsutrht`. Retreived: June 2016.

[12] KROTOFIL, M., AND LARSEN, J. What You Always Wanted and Now Can: Hacking Chemical Processes. HITB, 2015. `http://tinyurl.com/gnbuh7b`. Retreived: April 2016.

[13] LANGNER, R. To kill a centrifuge. Tech. rep., Langner Communications, 2013.

[14] LARSEN, J. Breakage. black hat federal 2008. `http://tinyurl.com/jvdsxjl`. Retreived: April 2016.

[15] LARSEN, J. Hacking Critical Infrastructure Like Youre Not a N00b. RSA 2016. `http://tinyurl.com/hrshtlo`. Retrieved: May, 2016.

[16] LEVESON, N. G. *Engineering a Safer World: Systems Thinking Applied to Safety*. The MIT Press, 2012.

[17] LINDNER, F. Building Custom Disassemblers – Instruction Set Reverse Engineering. Black Hat Europe 2011. `http://tinyurl.com/hf78b54`. Retreived: April 2016.

[18] MAXIMUM INTEGRATED. NOVATO (MAXREFDES16#): 4-20MA Loop-Powered Temperature Sensor with HART. `http://tinyurl.com/hetmjxu`. Retreived: September 2015.

[19] MCINTYRE, C. Using Smart Instrumentation. plant engineering: online magazine. `http://tinyurl.com/jb3pouj`, 2011. Retreived: April 2016.

[20] MICROCHIP. MCP3201 – Datasheet. `http://tinyurl.com/jfo87mu`, 2011. Retreived: October 2015.

[21] PERRON, P. Test Consistency with Varying Sample Frequency. *Econometric Theory 7* (1991), 341–368.

[22] ROSS, R., MCEVILLEY, M., AND CARRIER OREN, J. NIST 800-160: Systems Security Engineering – Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems. NIST Special Publication, May 2016.

[23] SCHNEIDER ELECTRIC. Altivar 12 – Variable speed drives for asynchronous motors. User manual. `http://tinyurl.com/h55kp8j`, 2009. Retreived: September 2016.

[24] SMUTS, J. F. *Process Control for Practitioners*. OptiControls Inc, 2011.

[25] STAMATIS, D. H. *Six Sigma and Beyond Series – Statistical Process Control*, vol. 4. CRC Press, 2002.

[26] TOECKER, M. Digital Maintenance and Test Equipment and Impact on Control System Security. 4SICS, 2015. `http://tinyurl.com/gqa5klh`. Retreived: June 2016.

[27] WIGHTMAN, R. Modbus-VCR. GitHub repository, 2013. `http://tinyurl.com/zxhjkho`. Retreived: April 2016.

[28] ZADDACH, J., KURMUS, A., BALZAROTTI, D., BLASS, E., FRANCILLON, A., GOODSPEED, T., GUPTAK, M., AND KOLTSIDAS, I. Implementation and Implications of a Stealth Hard-Drive Backdoor. In *Proceedings ACSAC'13* (2013), pp. 279–288.