Creating Effective Industrial-Control-System Honeypots

Neil C. Rowe, Thuy D. Nguyen, Marian M. Kendrick,
Zaki A. Rucker, Dahae Hyun, and Justin C. Brown
U.S. Naval Postgraduate School
Monterey, CA 93943 USA
Contact: ncrowe@nps.edu

Abstract

Cyberattacks on industrial control systems (ICSs) can be especially damaging since they often target critical infrastructure. Honeypots are valuable network-defense tools, but they are difficult to implement for ICSs because they must then simulate more than familiar protocols. This research compared the performance of the Conpot and GridPot honeypot tools for simulating nodes on an electric grid for live (not recorded) traffic. We evaluated the success of their deceptions by observing their activity types and by scanning them. GridPot received a higher rate of traffic than Conpot, and many visitors to both were deceived as to whether they were dealing with a honeypot. We also tested Shodan's Honeyscore for finding honeypots, and found it was fooled by our honeypots as well as others when, like most users, it did not take site history into account. This is good news for collecting useful attack intelligence with ICS honeypots.

1. Introduction

Industrial control systems (ICSs) are important concerns for cybersecurity even though they are not attacked often, because an attack on critical infrastructure such as a power grid can have catastrophic effects on the operation of business and government [1]. ICSs are vulnerable due to their proprietary software and protocols, legacy devices, and outdated operating systems. Furthermore, they are difficult to update because many must provide continuous operation.

Honeypots (decoy digital systems) are useful defensive tools to investigate cyberattack threats and other kinds of malicious activity. It is useful to develop honeypots for ICSs to collect intelligence on their kinds of attacks. However, creating an effective honeypot for ICSs is more difficult than for most network nodes because they must simulate a wide variety of real-time industrial processes using proprietary protocols as well as standard protocols like TCP/IP. If ICS honeypots are not simulated accurately, attackers may realize they are being deceived and go away, failing to provide useful intelligence.

We installed two open-source industrial-control-system honeypots, Conpot [2] and GridPot [3], and studied the live traffic to them to evaluate their effectiveness. GridPot is a modified Conpot designed to simulate different electric-grid models. Conpot is simpler and served as a control experiment to see if GridPot was a significant advance. While they have been assessed separately, no previous experiments have compared their performance in the same environment. We also checked whether a commonly used network scanning tool could identify these and similar honeypots.

2. Background

2.1. Electric grids

Our experiments focused on ICSs simulating electrical grids. An electric grid (or bulk electric power system) includes generation, transmission, distribution, and end use [4]. The generation of electricity occurs in many ways including coal-fired plants, natural-gas plants, solar farms, wind turbines, and hydroelectric plants. Generated electricity is passed through transformers to step up voltage to a very high level where it is then transferred to transmission lines. Transmission lines deliver electricity to substations. Substations use a transformer to step down the voltage from high to low voltage before it is distributed to end users. A switch is a device used to direct the flow of current by opening and closing a circuit. A

URI: https://hdl.handle.net/10125/63967
978-0-9981331-3-3

substation employs each of these devices to safely control the transfer of current from transmission to distribution. Regulators ensure a constant and safe voltage level is maintained throughout the bulk power system.

Software now allows operators to monitor and control portions of an electric grid without being at each physical location. SCADA (Supervisory Control and Acquisition) devices, a subset of ICSs, in particular allow operators to monitor many devices over a wide area. An IED (Intelligent Electronic Device) like a controller or a digital relay can send or receive data or control to or from an external source. ICSs are a subtype of cyber-physical systems, which integrate physics and logic to allow interaction between digital, analog, physical, and human components.

Electrical grids and related infrastructure have been targeted by a number of types of malware [5]. Well-reported cases involved CRASHOVERRIDE or Industroyer, STUXNET, BLACKENERGY 2, and HAVEX. In December 2016, a transmission-level substation was attacked in the Ukraine using CRASHOVERRIDE [6]. Proof that grid operations can be severely affected by a cyberattack was demonstrated in a U.S. Department of Energy test which caused the self-destruction of a replica power plant generator by means of a cyberattack [7].

Cyber threats that target the distribution portion of the bulk power system are load shedding, advanced metering infrastructures, and demand-side management [8]. The U.S. has seen load-shedding incidents in recent years that have caused cascading power outages. In 2007, Tempe, Arizona experienced large-scale load shedding which affected 98,700 customers for almost an hour.

2.2. Honeypots for ICSs

Our research explored the use of honeypots to collect data of cyberattacks on electric utilities. Honeypots can be most useful for gathering information about attacks when they entice attackers into revealing a rich set of information about their attacks [9]. Honeypots that conceal their purpose through deception are more productive because attackers do not want to interact with honeypots [10]. High-interaction honeypots can confuse attackers through program-based or scripted interaction designed to encourage further exploration. However, some botnets have evolved to become aware of honeypots [11] by detecting firewalls and filters on outbound traffic. Honeypots desire to limit their liability in case they are used to launch an attack on a third party. This could be done with an intrusion-detection system that filters for outbound activity. A bot that is prevented from spreading malicious data from a honeypot would detect that and recognize it is on a honeypot.

Several projects have used honeypots to monitor attacks on ICSs. One project deployed a large-scale cloud-based low-interaction honeypot system for 28 days using Amazon's EC2 cloud environment [12]. This experiment monitored the protocols DNP3, ICCP, IEC-104, Modbus, SNMP, TFTP, and XMPP. The researchers concluded that reconnaissance occurred more often than actual attacks and it targeted single protocols rather than combinations of protocols. They also identified a positive correlation between Modbus reconnaissance and discovery by the Shodan network-scanning tool of Modbus-enabled devices.

HoneyPhy [13] addressed the problem that ICS honeypots could be unrealistic in modeling device physics and device-actuation times and therefore could be identifiable. One honeypot they designed provided general structure-modeling processes and devices implementing a simple heating-ventilation system. Another modeled a simplified water-treatment system.

The GasPot honeypot simulated the Veeder-Root Guardian aboveground storage-tank monitoring system [14]. Logs revealed unauthorized reads and writes, defacement, and denial of service attacks. GasPot was subsequently integrated into Conpot as the guardian_ast template. Similarly, the kamstrup_382 template provided by Conpot mimics a Kamstrup model 382 smart electrical meter, providing an electrical power metering service on port 1025 and a management service on port 50100. The hardware on which this is based provides measurement of electrical circuits up to three-phase ones, allows remote access by way of optional modules for TCP/IP networking over Wi-Fi, GSM, and GPRS connectivity, and enables local interaction via optional serial and infrared interfaces.

Another honeypot architecture used geographically dispersed nodes hosted on Amazon Elastic Cloud Compute with emulation support for the protocols DNP3, ICCP, IEC 104, Modbus, SNMP, TFTP, and XMPP [15]. In experiments, the Shodan network scanner provided the first unsolicited interaction with five of the six honeypots, and attacks began only after each honeypot was listed on Shodan. This suggests that attackers are exploiting network-scanning databases.

Other similar honeypot projects were CryPLH [16] and the Digital Bond SCADA Honeynet [17].

2.3. Network scanning

Honeypots may be detectable by distinctive clues they provide to network protocols. Network scanning can look for these. Transport-layer scanners send some combination of TCP, UDP, and ICMP packets to a remote host, waiting an assigned time for responses. Nmap (nmap.org) is popular tool for transport-layer scanning. Protocol scanners interact with specific application-level protocols and require expertise with proprietary communications protocols. For instance, Digital Bond's Redpoint uses Nmap's NSE tool to use ICS protocols [18]. Within Redpoint the s7-info.nse script can do simple interactions with Siemens PLC devices with port 102 and S7Comm.

Some scanners focus on specific protocols like zmap (zmap.io), and others scan more broadly such as ZoomEye (www.zoomeye.org). The Shodan scanner (www.shodan.io) scans Internet-connected hosts continuously. It appears to pick IP addresses randomly and is more successful in the IPv4 address space [19]. The Shodan website provides a service called Honeyscore which uses a proprietary algorithm to identify honeypots.

Project SHINE for two years queried Shodan for selected search terms, starting with manufacturer names from trade magazines and blogs, and continuing by query terms derived from the results of searches on manufacturer names [20]. Eventually they sampled 2,186,971 devices from which they derived 578 unique search terms for traditional industrial-control system devices and 349 search terms for "non-traditional" devices but having physical controls of some kind. The protocols studied were S7Comm, Modbus, DNP3, EtherNet/IP, and BACNet. Scanning did include the possibility of duplicate devices and NAT connections.

In another experiment, a Siemens RuggedCom RS910 was configured to respond as a water pump [21]. This RUGGEDTRAX honeypot was configured to run SSH, HTTP, HTTPS, and DNP3 services. The device firmware name and version were displayed on its HTTPS web page, alongside a fictitious System Name indicating a water well in a specific location. The firmware sent a variant of the "goahead" embedded web server banner. The honeypot was indexed by Shodan two days after being connected to Internet.

An experiment with the Bodenheim tool [19] observed the success of Shodan in identifying Internet-connected industrial-control system devices. For 55 days Bodenheim connected a set of four Allen-Bradley ControlLogix 1756-L61 controllers to the Internet as honeypot. Two controllers connected with an unmodified "Standard" HTTP banner, one with an "Obfuscated" banner, and one with an "Advertised" HTTP banner. All four were probed within four days of deployment, and two within a day; data from all four was visible on the Shodan website within 19 days of deployment, despite never having provided their addresses to Shodan. This could reflect historical data about the addresses.

3. Experiments with Conpot

We first tested a low-interaction ICS honeypot, Conpot from conpot.org [2]. It simulates an ICS such as a power plant and collects information of cyberattacks. It acts as a master server for commonly used ICS network protocols and provides multiple templates that simulate simple forms of them. Conpot served as our control experiment for the subsequent experiments with GridPot. More details are in [22].

3.1. Methodology

Our experiments used a laptop computer with a Linux Ubuntu 16.04.3 LTS operating system. A Linux virtual machine was installed using Oracle VM Virtualbox 5.1.20, and Conpot 0.5.1 was installed in it. A local network was set up outside of our school's firewall to make it easier for live attackers to discover the honeypot without advertising it. Both the host and virtual machines used statically assigned IPv4 addresses and communicated with internal bridged networking. While our local network could not be mistaken for a major ICS installation, it could simulate a small node on an electrical grid.

Conpot offers four different templates. In our first experiments we used the "default template" which simulates an electric-power plant using Siemens SIMATIC S7-200 Programmable Logic Controllers that communicate with at least two slaves. Conpot simulates the initial interactions with the protocols HTTP, Modbus over TCP/IP, S7Comm, SNMP, BACnet, IPMI, EtherNet/IP, and CIP. We created parsers to extract clues from log data for each protocol such as IP addresses, ports, and basic protocol-specific data. The IPMI emulator was special in that it mimics a baseboard management controller supporting functions such as "chassis status" and "user list", and permits manipulation of system power [23].

3.2. Conpot results

Our honeypot collected live traffic over four months from October 2017 to February 2018 except when the main log was backed up. The network protocol analyzer Wireshark (www.wireshark.org) monitored and captured network traffic. Table 1 summarizes the traffic counts by protocol seen by Conpot and in the subsequent experiments with GridPot.

Table 1: Traffic percentages by protocol in live-traffic testing of the two honeypots.

Protocol	Conpot count	Conpot percentage	GridPot count	GridPot percentage
HTTP	7,366	66.7%	9,641	93.0%
Modbus	2,316	21.0%	621	6.0%
S7Comm	645	5.8%	102	1.0%
BACnet	311	2.8%	0	0.0%
IPMI	262	2.4%	0	0.0%
EtherNet/IP	154	1.4%	0	0.0%
Total	11,054	100%	10,364	100%

Only 59 of the 2,316 Modbus activities had a valid Modbus function code of the 19 possible: 0x03 (Read Holding Register), 0x2b (Read Device Identification), and 0x11 (Report Server ID). These suggest reconnaissance only. Spikes of activity occurred on November 13, November 23, December 8, and January 10. On November 16, ICS-CERT released a security advisory about Siemens SICAM equipment; Modbus is supported by this product, and the equipment emulated by Conpot is also a Siemens product, so that probably explains two of the activity spikes.

With EtherNet/IP activities, only NOP, RegisterSession, and ListIdentity commands were used. Invalid commands or null commands were also observed; these suggest probing attempts. For S7Comm, all packets had a data length of 0 or 8 and a request ID of 0. For BACnet, some data apparently was sent for all 78 established connections using an invalid type, resulting in 78 decoding errors, but we could not identify where it came from. For IPMI activities, 129 had new traffic, 30 were returning traffic, and only 2 sessions were properly closed.

More overall activity occurred in October and November, and then it gradually declined (Figure 1), as is typical of new honeypots. The decline was predominantly due to HTTP and the spikes were predominantly due to Modbus activities. Claimed nationalities of the attacks were all over the world, suggesting diverse international reconnaissance (Table 2).

Table 2: Country claimed by Conpot attack traffic observed.

Country	Percent	Country	Percent
U.S.	27%	Netherlands	5%
China	22%	Hong Kong	4%
Brazil	8%	France	4%
Russia	8%	Indonesia	3%
Egypt	6%	Indonesia	3%
India	6%	Japan	3%

As could be expected with a low-interaction honeypot, overall traffic was mostly reconnaissance. Conpot traffic concentrated on the most familiar protocol, HTTP, despite its limited capability on this honeypot. Modbus traffic appeared to be testing access permutations using many malformed packets. EtherNet/IP and Modbus activities showed that Conpot had difficulty distinguishing between real protocol activities and embedded protocol requests sent to the ports on which Conpot listened. Since Conpot only logs basic flow data, embedded payloads to unnoticed.

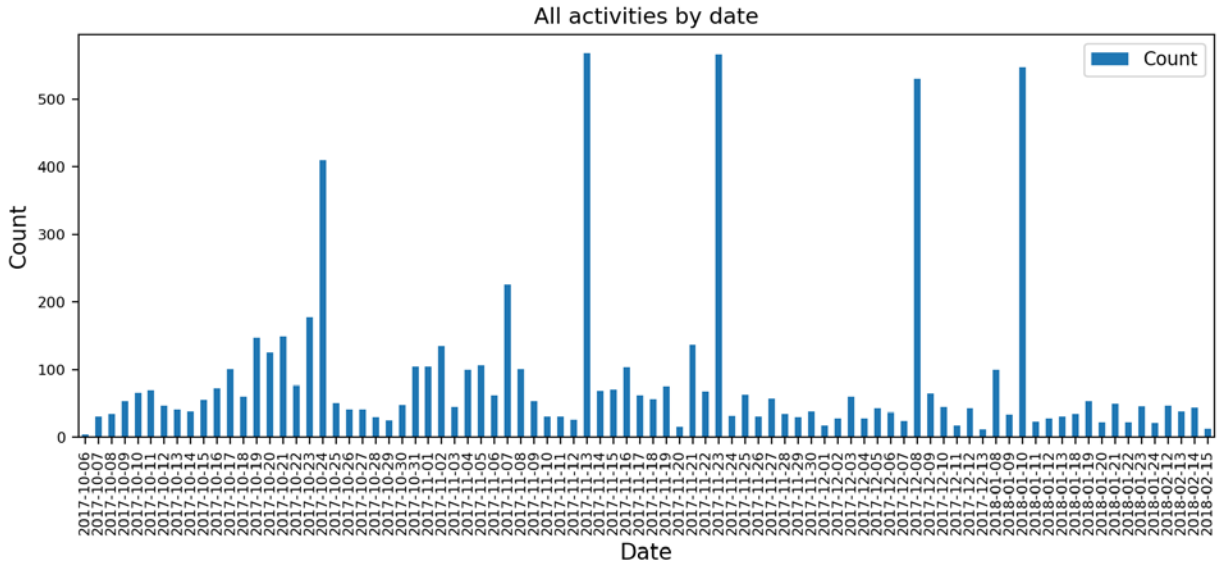

Figure 1: Conpot activity count of all protocols over time.

4. Experiments with GridPot

We also tested the GridPot open-source high-interaction ICS honeypot framework (github.com/sk4ld/gridpot). It uses GridLAB-D, a simulation and analysis tool for power-distribution systems (gridlab.org), developed by the U.S. Department of Energy and Pacific Northwest National Laboratory to enable modeling and testing of power distribution systems at low cost [24]. It was run on the same local network as our Conpot experiments to enable a fair comparison. More details are in [25].

4.1. Setup

GridLAB-D Model objects used in our honeypot were the node, link, switch, transformer, and regulator. Object node properties include phases, object connections, open status, power flow, temperature, tap position, and configuration. Objects can include schedules of parameter values over time. Network protocols we supported with GridPot were HTTP, Modbus, S7Comm, SNMP, and IEC 61850.

GridPot uses a honeypot layer and a modeling layer to add electrical components and integration between GridLAB-D and Conpot, including IEC 61850 communication. GridPot's honeypot layer is derived from Conpot, adding an XML-formatted GridPot template that specifies to which GridLAB-D model to link. Additional Python-coded GridPot files are included in the honeypot layer to retrieve parameter values from the running model in real time using port 6267.

GridPot's primary modeling layer uses GridLAB-D's Powerflow module, adding GridPot model (GPM) configuration files. Powerflow simulates voltage and current values across an IEEE 13 node grid model with 15 houses. GridPot source code includes additional modeling features for "intelligent electronic devices" under an electric components subdirectory. This contains code to simulate a GE Brick Merging Unit and a generic input/output switch control device.

Our experiments used both a test environment and a live environment. The test environment altered the Conpot code to use its localhost IP address instead of the host environment's external IP address, which kept traffic internal to our machine for testing. The live environment enabled external user access and threat data collection from outside the school firewall. We used network-address-translation, host-only-adapter, and bridged-adapter network settings in both environments.

We used Oracle VM Virtualbox 5.2.22 to install a virtual machine in which to place GridPot. It ran the same operating system as the host. The honeypot layer initialized Conpot using the GridPot template and the modeling layer initialized the GridLAB-D model IEEE_13_Node_With_Houses. We updated the "gridpotmodel_file" field value to link with our custom GPM file for the latter. Four protocol servers were started upon launch as written in the original source code. Modbus is used on TCP/IP port 502 connecting to one client and two servers.

The IEEE_13_Node_With_Houses model contained switch, transformer, and regulator objects used for the Conpot integration and required minimal code modifications. A schedule based on local time was used to alter power flow readings across the switch. Real-time power-in and power-out simulated switch parameter values which were displayed on our web-based interface. We created a GPM file to link with the switch, transformer, and regulator objects specified in the IEEE_13_Node_With_Houses GLM file by modifying an existing GPM file.

4.2. Testing

We used the Wireshark network-protocol analyzer (www.wireshark.org) to confirm that Conpot and Gridpot logs were complete; the Nmap (nmap.org) and Linux Netstat built-in network scanners to check which ports were open; the Nessus vulnerability-assessment tool (www.tenable.com) to check for obvious vulnerabilities; and the Metasploit penetration tester (www.metasploit.com) to test logging of attacks. Nessus and Metasploit were used with SCADA plugins.

First we tested if our web-based interface display was accurate to the running model by pointing a web-browser to GridPot's HTTP server using localhost IP address and TCP port 80, and comparing the results to the GridLAB-D model instance that listened on port 6267. We then used the Netstat tool to determine which ports were opened by GridPot. We then ran scans using Nmap, Nessus, and Metasploit against our honeypot. We focused these scans on open ports and probed for operational-technology devices using the Modbus protocol by running detection, discovery, and interaction scans. We focused on Modbus since it is the most common ICS protocol.

Host-to-virtual-machine baseline testing first required altering our network connectivity from disabled-network status to host-only status. We tested the host-only network status using "ping" commands between our host and virtual machine and confirmed receipt of a "network is unreachable" error when trying to ping an arbitrary IP address. The same scans using Nmap, Nessus, and Metasploit were then performed, changing the IP address to our GridPot virtual machine instead of the localhost address. To generate useful log data for comparison against live denial-of-service attacks, we conducted a scan using an auxiliary Metasploit module.

We modified portions of the default Conpot configuration in the template to eliminate well-known clues to Conpot and fool more attackers. Testing confirmed our web-based interface accurately displayed values of the running GridLAB-D model. Netstat results confirmed that our four protocol ports (HTTP, Modbus, S7Comm, and SNMP) were open. Results of the Nmap, Nessus, and Metasploit scans also saw these ports as open and that Modbus-enabled devices were running on our honeypot.

Our live honeypot collected data over 19 days from April 11-30, 2019. GridPot ran continuously except when we fixed a broken link. Conpot stopped logging twice, which could have been due either to bugs or malicious activity that we could not distinguish.

4.3. GridPot results

Live GridPot traffic data collected by Wireshark totaled 1,525,059 packets and 165 MBs. This was a higher traffic rate of 545 interactions per day versus 92 with Conpot. The GridPot protocol distribution differed from that of Conpot (Table 1). BACnet, IPMI, and EtherNet/IP were not included in the GridPot template we used and so were not logged, though there was likely a small amount of their traffic judging by the Conpot results. HTTP traffic was a larger percentage of traffic with GridPot. This is likely due to the additional deceptions beyond Conpot provided by GridPot that were accessible by HTTP, though a contributing factor could be the increasing numbers over a year of real electric grids that use HTTP [1]. It thus appears that GridPot's additional deceptions are justified and effective.

Heavy scanning with Modbus was seen twice. 39 unique source addresses sent packets to our honeypot multiple times, some of which demonstrated information had been learned from the first interaction.

The greatest number of packets (1,013,726) came from a California-based cloud-hosting

corporation. It came from an address registered to Fastly, a content delivery network provider. GridPot exchanged 84,588 packets with just one Fastly address using MaxMind. Traffic from this address occurred throughout our collection, and contained over 26,000 retransmissions of nearly identical ACK messages, so this campaign was not intelligent.

The second-highest source of packets was an IP address registered to an LLC in St. Petersburg, Russia, which was responsible for 56,280 packets of 3,221KB. Censys.io traced this address to a Debian-based SSH server in Amsterdam. 38,754 were SYN packets sent to GridPot, and there were also RST packets.

Seven different HTTP methods were seen in the HTTP requests, including 79 "None" and 78 "Bad". Significant spikes in the number of HTTP requests occurred almost daily (Figure 2). Each peak contained roughly the same number of GET and POST request methods in the same order with varying speeds. This suggests these attackers used a single HTTP scanning tool to conduct the attacks, and were not inspecting their results carefully. We compared this apparent scanning to our host-to-virtual-machine logs and concluded that steps of the real scans did not match any of our test scans because of the quite different distributions in the times of HTTP commands for these attacks.

Most Modbus traffic used the "none" function code, and the remainder split between function codes 17 and 43, totaling 597, 5, and 2, respectively. Protocol scanning using Modbus was visible in the form of incrementing slave ID numbers with each new request seen. By comparing to our host-to-virtual-machine logs, we inferred that Nmap and the "modbus-discover.nse" script were used in both cases by the similarity in the sequencing of function code, the slave ID, the request values, and the response values.

We observed 20 new S7Comm sessions, 102 S7Comm connections, 13 COTP connection requests, and 19 S7 packets. S7Comm messages were only of types 1 and 7, with counts 6 and 13 respectively.

Configuring GridPot was difficult due its lack of updating. But once configured, it proved successful at collecting intelligence for threat analysis. It is apparent that attackers thoroughly explore Web-based vulnerabilities in ICS interfaces, as seen in the large amount of HTTP traffic captured. The Modbus scanning indicated that our simulated grid was realistic enough to encourage specialized-protocol reconnaissance.

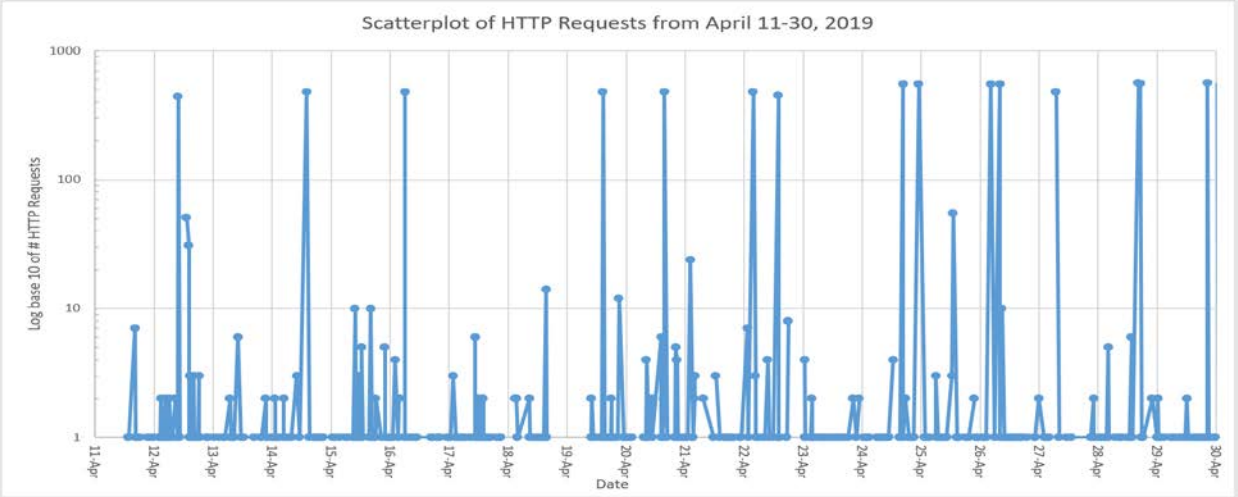

Figure 2: GridPot HTTP requests during live testing.

5. Testing scanning for honeypots

We also conducted tests using network scanning tools to try to detect honeypots [26]. Most of experiments used Shodan's Honeyscore, which rates a site from 0.0 (not a honeypot) to 1.0 (definitely a honeypot). The Honeyscore tool was sure that our final GridPot implementation was a honeypot when its history was taken into account, but was sure it was not a honeypot without the history. This is likely because we have reused the IP addresses often for other honeypot projects [27, 28], so Shodan's automated

scanning has found them many times, but the customization of the configuration of our GridPot implementation appeared to be a sufficient disguise of the honeypot without knowing the history. In fact, the disguise of our GridPot implementation may have been better than that of previous honeypots on the site because its traffic was significantly higher than the others, so not many visitors were inspecting the scanning history.

To further explore these issues, we examined the records of 122,668 Internet sites in Shodan's database collected between April 22, 2016 and April 20, 2017 that had records for one of eight ports known to be specifically related to ICSs according to the Digital Bond ICS Enumeration plugin: Modbus, S7Comm, BACnet, CODESYS, Niagara Fox, OMRONFINS, ProconOS, and Ethernet/IP. For 114 of these, no Honeyscore was reported for reasons unexplained. Of the remainder, 1063 sites had a Honeyscore of 0.5 or larger for a rate of 0.87%. Shodan does not publish their criteria for Honeyscore, apparently to discourage honeypot developers from engineering easy countermeasures. However, some clues are obvious, so as a simple test, we explored three heuristics:

- H1: A device which services the S7Comm protocol on tcp/102 and returns the terms "Technodrome", "Mouser", or "88111222" is a honeypot. These strings occur in the PLC Name, Plant ID, and Serial number fields in Conpot default implementations. These are implausible as values in a production S7Comm service.

- H2: A device providing the same ICS services as Conpot's default template, plus or minus one service, is a honeypot. Those services are HTTP (port 80), S7Comm (102), SNMP (161), Modbus (502), IPMI (623), and BACnet udp (47808). It is unusual to see these services together otherwise.

- H3: A device providing industrial-control services from a public cloud location is running a honeypot. Cloud locations came from hosts identified as matching H1 in the Shodan data, names with the keywords "cloud" or "hosting", and names listed in a "Most Reliable Hosting Company Sites" page at Netcraft.com.

None of these rules applied to our GridPot site:

- H1 does not apply because we changed default strings in our implementation. However, when we ran Conpot a year previously, it had the default strings [22].

- H2 does not apply because using Nmap against our GridPot honeypot, we found open ports 80, 102, 502, 6267, 8834, and 11211, so a scanner could match only 3 of the 6 target ports with 3 extra ports.

- H3 does not apply because our site did not offer any of those clues to directory services. It was listed by our Internet Service Provider (AT&T) as being associated with our school, but that is not one of the clues.

To test the heuristics, we supplemented the 1063 high-Honeyscore sites with all sites matching either H1, H2, or H3 in the Shodan database, to get a test set of 8127 sites. We manually inspected other scanning data to estimate that 748 of these were honeypots (9.2%). This inspection used DNS node data, IP geolocations, and "whois" service data whenever there was a strong disagreement between Honeyscore and our heuristics. For some hosts we transmitted active probes using the Nmap OS detection method and observed the response. Among the sites found from manual inspection were 550 Conpot clones identified by duplicated ICS serial numbers, duplicated SSH fingerprints on controllers, and in one case, an FTP honeypot on the same site.

Overall counts on the test set are shown in Table 3. We assumed a Honeyscore ≥ 0.5 as "ground truth" of a honeypot, and ignored the missing Honeyscore values. Then in the sample of 8127, H1 had a precision of 1.00, recall of 0.20, and an F-score of 0.33; H2 had a precision of 0.28, recall of 0.92, and an F-score of 0.43; H3 had a precision of 0.03, recall of 0.14, and an F-score of 0.14. So H2 was the best and H3 was the worst. Note there is a tradeoff between precision and recall which will need to be best set based on information-security priorities.

As for identifying honeypots using Honeyscore alone, the precision for Honeyscore ≥ 0.5 was $748/1063 = 0.703$. The recall was difficult to estimate, but it must be close to 1.0 because for H1 we found 7 sites with lower Honeyscores that we confirmed were honeypots; for H2 we found only 1 of 108 lower Honeyscores were honeypots; and for H3 we found no honeypots in a sample of 100 with lower Honeyscores. Thus the F-score is close to 0.70. Thus our heuristics can model some but not all of Honeyscore functionality.

Thus we conclude that Shodan can be fooled by an ICS honeypot with a few simple modifications to the default configuration that we can find by testing heuristics, provided Shodan is not examining site history. But on the other side of the coin, these results suggest that probes and attacks that do use history should be easy to fool with "fake honeypots" [29], real ICSs that have artifacts, services, and history of honeypots; Shodan's outdated information will "scare

Table 3: Results of testing a sample of 122,068 sites in Shodan's database for honeypot clues.

H1	H2	H3	Honeyscore ≥ 0.5?	Count
Y	Y	Y	Y	81
Y	Y	N	Y	16
Y	N	Y	Y	3
Y	N	N	Y	41
N	Y	Y	Y	25
N	Y	N	Y	566
N	N	Y	Y	234
N	N	N	Y	97
Y	Y	Y	N	1
Y	Y	N	N	2
Y	N	Y	N	0
Y	N	N	N	2
N	Y	Y	N	1
N	Y	N	N	1762
N	N	Y	N	4175
N	N	N	N	115548
N	Y	N	missing	2
N	N	Y	missing	3
N	N	N	missing	109

away" attacks and help protect these sites. Then if attackers try to counter this by ignoring the historical data and just testing the current properties of the site with heuristics like H1, H2, and H3, some small modifications to the site like those of our GridPot implementation will cause a Shodan-like system to conclude a real honeypot is not a honeypot. The nice thing about this strategy is that the sort of attackers for which this will work best are the more sophisticated and intelligent attackers who gather thorough intelligence before focused attacks, so these sites can provide some sorely needed defensive techniques for attackers to which we are especially vulnerable.

6. Conclusions

Due to their real-time requirements and proprietary protocols, ICSs are more difficult to simulate with honeypots than other kinds of network nodes. The two ICS-honeypot frameworks we tested, Conpot and GridPot, did seem to be effective, however; we saw more traffic to them, and more varied traffic, than to our previous secure-shell and Web honeypots despite the rarity of ICS sites on the Internet. GridPot was definitely more successful at deception than Conpot because it generated a higher rate of traffic, mostly HTTP. Deception was effective for both honeypots because most traffic either did not recognize features of a honeypot or did not care. For the minority of attackers who either inspect sites or use scanning tools against them, our sites were probably easy to recognize as honeypots since they were not on a specialized subnetwork. However, this means that a different kind of deception, "fake honeypots" that are real ICSs with honeypot artifacts like default Conpot configuration names, could encourage these attackers to leave.

Future work will explore this as well as adding more simulated devices and services to ICS honeypots to keep attackers interested longer. Future work will also involve industry collaborators. Our data is available for other researchers to use under restrictions.

Acknowledgements

This work was supported in part by the NPS Foundation. The views expressed are those of the authors and do not represent the U.S. Government.

References

[1] E. Knapp and J. Langill, Industrial Network Security. 2nd ed. Waltham, MA, USA: Syngress, Palo Alto, CA, 2015.

[2] A. Jicha, M. Patton, and H. Chen, "SCADA Honeypots: An In-Depth Analysis of Conpot," in Proc. of the 2016 IEEE Conf. on Intelligence and Security Information, Tucson, AZ, USA, 2016, pp. 196–198.

[3] W. Redwood, "Cyber Physical System Vulnerability Research". Ph.D. Dissertation, Florida State University, 2015.

[4] S. Blume (Ed.), Electric Power System Basics for the Nonelectrical Professional, Wiley, New York, 2007, pp. 53-89.

[5] C. Bodungen, B. Singer, A. Shbeeb, K. Wilhoit, and S. Hilt, Hacking Exposed Industrial Control Systems: ICS and SCADA Security Secrets and Solutions. New York, NY, USA. 2016.

[6] NCCIC, "CrashOverride Malware," Washington, DC, USA, Alert ICS-ALERT-17-206-01, 2017.

[7] C. Theohary, "Cyber Operations in DoD Policy and Plans: Issues for Congress," CRS Report No. R43848, 2015.

[8] S. Sridhar, A. Hahn, and M. Govindarasu, "Cyber–Physical System Security for the Electric Power Grid," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 210–224, Jan. 2012.

[9] R. Joshi and A. Sardana, Honeypots: A New Paradigm in Information Security, CRC Press, Boca Raton, FL, 2011.

[10] N. Rowe, "Honeypot Deception Tactics", Chapter 3 in E. Al-Shaer, J. Wei, K. Hamlen, and C. Wang (Eds.), Autonomous Cyber Deception: Reasoning, Adaptive Planning, and Evaluation of HoneyThings, Springer, Chaum, Switzerland, 2018, pp. 35-45.

[11] C. Zou and R. Cunningham, "Honeypot-Aware Advanced Botnet Construction and Maintenance," in International Conference on Dependable Systems and Networks (DSN'06), 2006, pp. 199–208.

[12] A. Serbanescu, S. Obermeier, and Y. Der-Yeuan, "Threat Analysis Using a Large-Scale Honeynet," in Proc. of the 3rd Int. Sym. for ICS & SCADA Cyber Sec. Res., 2015.

[13] S. Litchfield, "HoneyPhy: A Physics-Aware CPS Honeypot Framework," M.S. thesis, Dept. of Elec. and Comp. Eng., Georgia Inst. of Tech., Atlanta, GA, USA, 2017.

[14] L. Rist, "Gas Tank Monitoring System Honeypot", September 2015, available at www.honeynet.org/node/1269, accessed July 17, 2016.

[15] A. Serbanescu, S. Obermeier, and D.-Y. Yu, "A Flexible Architecture for Industrial Control System Honeypots," in 2015 12th International Joint Conference on e-Business and Telecommunications, July 2015, vol. 4, pp. 16–26.

[16] D. Buza, F. Juhász, G. Miru, M. Félegyházi, and T. Holczer, "CryPLH: Protecting Smart Energy Systems from Targeted Attacks with a PLC Honeypot, Springer Chaum, Switzerland, 2014, pp. 181–192.

[17] Digital Bond, Inc., "SCADA Honeynet", available at digitalbond.com/tools/scada-honeynet, 2016, accessed August 20, 2016.

[18] Digital Bond Inc., "Digital Bond's ICS Enumeration Tools", available at github.com/digitalbond/Redpoint, accessed May 26, 2016.

[19] R. Bodenheim, J. Butts, S. Dunlap, and B. Mullins, "Evaluation of the Ability of the Shodan Search Engine to Identify Internet-Facing Industrial Control Devices," International Journal of Critical Infrastructure Protection, vol. 7, no. 2, pp. 114–123, 2014.

[20] B. Radvanovsky and J. Brodsky, "Project SHINE (SHodan INtelligence Extraction) Findings Report", October 2014, available at www.slideshare.net/BobRadvanovsky/project-shine-findings-report-dated-1oct2014, October 2014, accessed August 31, 2016.

[21] B. Radvanovsky, "Project RUGGEDTRAX SCADA/ICS Analysis Findings Report", available at www.slideshare.net/BobRadvanovsky/project-ruggedtrax-findings-report-28nov2015, November 2015.

[22] D. Hyun, "Extraction and Analysis of IOCs Using Honeypots", M.S. thesis, U.S. Naval Postgraduate School, March 2018.

[23] P. Soòky, "Extented Functionality of Honeypots", B.S. thesis, Brno University of Technology, available at dspace.vutbr.cz/bitstream/handle/11012/52363/16127.pdf?sequence=2, 2015, accessed September 3, 2016.

[24] N. Lu, Z. Taylor, D. Chassin, R. Guttromson, and S. Studham (June 16, 2005). "Parallel Computing Environments and Methods for Power Distribution System Simulation", Proc. IEEE Power Engineering Society General Meeting, San Francisco, CA, US, June 2005, pp. 215-219.

[25] M. Kendrick and Z. Rucker, "Energy-Grid Threat Analysis Using Honeypots", M.S. thesis, U.S. Naval Postgraduate School, June 2019.

[26] J. Brown, "Identifying Honeypots among Internet-Connected Industrial Control Devices", M.S. thesis, U.S. Naval Postgraduate School, expected September 2019.

[27] W. Chong and C. Koh, "Learning Cyberattack Patterns with Active Honeypots", M.S. thesis, U.S. Naval Postgraduate School, September 2018.

[28] B. Henderson, S. McKenna, and N. Rowe, "Web Honeypots for Spies", Intl. Conf. on Computational Science and Computational Intelligence, December 2018, Las Vegas, NV, USA, pp. 1-6.

[29] N. Rowe and J. Rrushi, Introduction to Cyberdeception, Springer, New York, 2016.