POSTER: Towards Highly Interactive Honeypots for Industrial Control Systems

Stephan Lau
Freie Universität Berlin
stephan.lau@fu-berlin.de

Johannes Klick
Freie Universität Berlin
johannes.klick@fu-berlin.de

Stephan Arndt
Freie Universität Berlin
stephan.arndt@fu-berlin.de

Volker Roth
Freie Universität Berlin
volker.roth@fu-berlin.de

ABSTRACT

Honeypots are a common tool to set intrusion alarms and to study attacks against computer systems. In order to be convincing, honeypots attempt to resemble actual systems that are in active use. Recently, researchers have begun to develop honeypots for programmable logic controllers (PLCs). The tools of which we are aware have limited functionality compared to genuine devices. Particularly, they do not support running actual PLC programs.

In order to improve upon the interactive capabilities of PLC honeypots we set out to develop a simulator for Siemens S7-300 series PLCs. Our current prototype XPOT supports PLC program compilation and interpretation, the proprietary S7comm protocol and SNMP. While the supported feature set is not yet comprehensive, it is possible to program it using standard IDEs such as Siemens' TIA portal. Additionally, we emulate the characteristics of the network stack of our reference PLC in order to resist OS fingerprinting attempts using tools such as Nmap.

Initial experiments with students whom we trained in PLC programming indicate that XPOT may resist cursory inspection but still fails against knowledgeable and suspicious adversaries. We conclude that high-interactive PLC honeypots need to support a fairly complete feature set of the genuine, simulated PLC.

Keywords

Honeypot; Programmable Logic Controller (PLC); Industrial Control Systems (ICS); SCADA

1. INTRODUCTION

Connecting devices to the Internet often increases their utility – at the price of significant risks to their integrity and availability. Of particular concern are espionage and low intensity conflicts that target critical infrastructures such as industrial processes. However, unless a cataclysmic event occurs, the impeding risks are unlikely to stem the flood of devices being connected to the Internet.

In order to defend against these risks, it is necessary to build capabilities to detect and deter threats to critical infrastructures. Effective deterrence requires that one attributes attacks to their origins. At the time of writing, attacks often proceed for weeks or even months before they are detected. Once an adversary realizes that his attack is detected, he will likely take measures to make attribution harder. Therefore, it is attractive to observe and study how an adversary proceeds and what level of sophistication he has, without him becoming aware his attack has been detected.

Honeypots are a classical detection mechanism. In order to be effective, honeypots must keep adversaries occupied or even resist attempts of suspicious adversaries to tell honeypots apart from genuine systems. Most honeypot technology has been developed for Internet services and not for industrial processes. While honeypots can be built using genuine control system hardware, this is not a particularly cost effective or scalable approach. A software-based solution would certainly be preferable.

In our poster, we present the current status of our ongoing work to develop honeypot technology for industrial control systems (ICS). We focus on Siemens S7-300 series controllers specifically, because Siemens is a market leader in the ICS market.

Several other research projects aim at developing honeypots for industrial control systems as well. We are only aware of projects that target Siemens PLCs as we do. However, we found that these projects still offer limited interactivity. With our project we aim to improve interactivity so that we can monitor and analyze adversaries' actions in addition to detecting them.

In what follows, we introduce a hierarchical classification system with clearly delineated classes and we categorize related work according to it. Subsequently, we introduce XPOT, our medium-interactive honeypot technology. We continue with a description of our evaluation efforts and end with brief conclusions.

2. HONEYPOT CLASSIFICATION

Honeypots are typically classified according to their level of interactivity. However, the pertinent taxonomy is not standardized and several different flavors of definitions ex-

CCS'16 October 24-28, 2016, Vienna, Austria

ACM ISBN 978-1-4503-4139-4/16/10.

DOI: http://dx.doi.org/10.1145/2976749.2989063

ist [6, 8, 11]. The terms *low-interactive* and *high-interactive* are widely used, *medium-interactive* or *pure* are seen less often. We suggest a more rigorous classification according to clearly delineated criteria. Consider a simple abstraction of a PLC, which consists of a *host* and a *program* (run by the host). Both the host and the program have inputs and outputs. A honeypot can have:

1. **Low interactivity:** The adversary can interact with the host but not with the program.
2. **Medium interactivity:** The adversary can interact with the host and the program.
3. **High interactivity:** The adversary can interact with the host and the program and he can read and write programs.

Interaction is defined as sending queries and receiving replies. Within each class we can establish a partial order of interactivity by defining sets of interactions an adversary is allowed to make. For example, we can limit the adversary to subsets of instructions or functions of the PLC. One honeypot is *more interactive* than another within the same class if and only if the set of interactions it allows is a proper superset of what the other honeypot allows. In what follows we apply our classification system to related work.

3. RELATED WORK

Conpot [10] is a low-interactive honeypot according to our classification. Among other protocols, it simulates a Siemens SIMATIC S7-200 PLC with Modbus and S7comm connectivity. Its default setup can be extended to simulate other Siemens PLCs which use the proprietary S7comm protocol. The implementation of S7comm protocol is fairly incomplete, though. At the time of writing, it is only possible to read entries of the *System State List* (SSL). Conpot adds two of them by default, which identify the model and version of the Siemens PLC. A genuine Siemens PLC has about 1000 of these entries. The missing ones make it easy for an adversary to identify Conpot.

CryPLH [1] simulates a S7-300. The authors identify CryPLH as a high-interactive honeypot with the stated goal to improve upon the interactivity, configurability and indistinguishability of previous developments. CryPLH reproduces static copies of the web interface of a genuine PLC which appear identical to the originals. The login is disabled so that adversaries cannot access status information. CryPLH offers a SNMP service that identifies as the PLC and provides network statistics obtained from the host OS. Adversaries can even connect to CryPLH using Siemens' SIMATIC Step 7 software. However, CryPLH simulates the highest protection level and rejects any passwords submitted to it. Thereby, CryPLH prevents further exploration by adversaries. Since adversaries can neither observe nor modify the program the PLC supposedly runs, CryPLH would still be classified as low-interactive in our classification scheme. However, the Nmap TCP/IP OS fingerprint is distinct from that of a genuine PLC, which renders CryPLH readily identifiable as a honeypot.

The ICS Security Workspace [3] operators apparently host PLC honeypots. Since they did not publish descriptions of their honeypot setup or its capabilities we can only make inferences from honeypot logs they released. Their logs resemble Snap7 [9] output and show many requests for identification entries in the SSL, most of which originated from Shodan [7] and Censys [2]. Two connections are particularly interesting. One connection queried additional SSL entries and some program and data blocks. An engineering workstation might have been used. The other queried for a single configuration data block and some unusual SSL entries. They mention three attempted attacks in their blog [4]: adversaries tried to stop program execution, modify memory areas and adjust the system clock. Overall it seems that the honeypot of ICS Security Workspace allows greater interactivity than the two we discussed before but still does not support interaction with simulated PLC programs. Based on this limited information, we categorize their honeypot as low-interactive.

4. THE XPOT HONEYPOT

We actively develop a medium to high interactive honeypot that simulates a Siemens SIMATIC S7 314C-2 PN/DP, our reference model. It is also possible to simulate almost any other S7-300/400 model, since all the models in this family are rather similar. In what follows we summarize our adversarial model and we highlight two distinctive properties that set our honeypot XPOT apart from other related projects.

Model of the Adversary.

We target an adversary model that allows adversaries to interact with the honeypot freely. In particular, we expect that adversaries will attempt the following interactions in order to quickly determine whether they are presented with a low-interactive honeypot: (1) perform an Nmap scan, (2) connect to the honeypot with Step7 software or the TIA portal software, (3) read the complete configuration and state, (4) download and upload programs, (5) debug running programs, (6) inspect and modify memory areas. Obviously, an adversary will defeat our honeypot if he probes features of a genuine PLC that XPOT does not support. XPOT will be defeated as well if its supported features are measurably different from those of a genuine PLC. This yields two classes of adversaries: those operating within the set of supported features and those operating without.

TCP/IP Stack Manipulation.

In order to avoid identification by OS fingerprinting software, we process all incoming and outgoing TCP/IP connections so that they appear equal to our reference model. The manipulations we make involve adjustments to TCP sequence numbers, ACK numbers, and TCP options, to name a few examples. We achieve a fingerprint for XPOT that is nearly identical to that of our reference model. For example, Nmap will report the same OS for XPOT and our reference model. Our implementation builds on *nfqueue*, a project that redirects network packets that traverse the kernel packet filter into a queue that is processed by a userspace application.

Programmability.

In order to achieve high interactivity according to our classification, we support the execution of PLC programs that an adversary may load onto XPOT using, for example, Siemens' TIA portal. PLC programs consist of blocks of bytecode encoded in the proprietary MC7 format. MC7 resembles an assembler language with an instruction set comparable to *Instruction List* (IL) as standardized in IEC 61131-3 [5].

MC7 consists of 1900 opcodes, which yield 146 different instructions of which we currently support more than 100. A Siemens PLC can execute MC7 instructions fairly fast, which makes it attractive to compile bytecodes to native code. The downside is that compiling the bytecode takes a few seconds up to minutes, depending on the underlying platform and the size of the compiled program. A real PLC starts to execute new programs immediately and therefore adversaries could easily tell XPOT apart from a genuine PLC. In order to avoid this, we interpret the bytecode until its compilation has finished. Towards this end we leverage LLVM's capabilities. Interpretation and compilation build on the same LLVM Intermediate Representation (IR) implementation of MC7 semantics, which avoids the duplication of efforts. The IR representation lends itself to potential future uses such as code analysis and model checking.

5. EVALUATION

Pilot Experiments.

We regularly offer a hands-on course on PLC programing and hacking at our university and we conducted a pilot study with students who completed our course. The course typically lasts three weeks of which one week focuses on PLC programming skills using our reference PLCs. At the end of our most recent course (10 participants), we asked six randomly picked participants to distinguish a genuine PLC from XPOT. Our objective was to explore how one might study the resistance of PLC honeypots against adversaries with diverse skill sets. For this reason, we separated the experiment into different stages. We gave participants two IPs (one for the honey pot and one for the reference PLC) but participants did not know which was which. We started with simple tasks and restricted the sets of PLC features we asked students to explore. In subsequent stages, we gradually extended the features and the freedom given to students. As might be expected, they were not successful in earlier stages but became increasingly successful in later stages. All students succeeded in the last stage when we removed all limitations to the tools and features they could use. The differences participants found ranged from non-working exploits to missing features and different PLC diagnostic output. Not all differences led to a correct identification of the honeypot, participants were sometimes unsure what constituted correct behavior. The results we obtained were consistent with a comparable pilot study we had conducted in the year before. The outcomes indicate that it is difficult to fool adversaries who have an understanding of PLC programming and who suspect that they encountered a high-interactive honeypot. Highly interactive honeypots must therefore support a fairly complete set of the features of a genuine PLC in order to remain convincing for extended periods of time.

Cursory Internet Exposure.

We made XPOT publicly available on the Internet for one month using a dynamic consumer DSL link, that is, a connection not associated with our university. In this experiment, our motivation was curiosity rather than the intent to conduct a formal study. We observed several full S7comm handshakes and queries for typical identification entries. Beyond that we did not observe suspicious activity. These results are similar to those of CryPLH [1].

6. CONCLUSION

If Internet-facing PLCs of the Siemens variety are attacked then it appears that adversaries are selective in their choices of targets. This is only to be expected given the stakes at hand. The barrier of entry for careless amateurs is high, on the other hand, because PLC programming is a comparably obscure topic. Any attempts that go beyond simple applications of the tools that can be found on the Internet require considerable effort and bear the risk of serious investigations. Perhaps for these reasons we have not registered attacks by amateurs either.

XPOT is still an early prototype even though it provides significantly enhanced interactivity and indistinguishability compared to related efforts. Despite our ongoing effort, there still remain numerous features and opportunities for adversaries to distinguish XPOT from a genuine Siemens PLC. However, doing so requires some degree of determined probing for uncommon features or knowledge of a particular feature that XPOT does not yet support, which we consider a step forward towards our goal.

Of course, the accurate simulation of a PLC is only one aspect of a convincing honeypot. Determined adversaries may scrutinize the process that the PLC appears to control, perhaps even to the point that they disturb the process slightly in order to measure whether or not sensors register the effect. In order to make this more difficult for adversaries, we are in the process of attaching XPOT to simulations of industrial processes.

7. REFERENCES

[1] D. I. Buza, F. Juhász, G. Miru, M. Félegyházi, and T. Holczer. CryPLH: Protecting Smart Energy Systems from Targeted Attacks with a PLC Honeypot. In *Smart Grid Security - Second International Workshop*, 2014.

[2] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman. A Search Engine Backed by Internet-Wide Scanning. In *Proc. of the 22nd ACM CCS*, Oct. 2015.

[3] ICS Security Workspace. ICS/SCADA Honeypot Log. http://plcscan.org/blog/dataanalysis/icsscada-honeypot-log/.

[4] ICS Security Workspace. Security Analysis from Siemens S7 PLC CPU Buffer [chinese]. http://plcscan.org/blog/2016/03/security-analysis-from-siemens-s7-plc-cpubuffer/.

[5] IEC. IEC 61131-3 International Standard, Programmable controllers. Part 3: Programming Languages. 2003.

[6] M. H. López and C. F. L. Reséndez. Honeypots: Basic Concepts, Classification and Educational Use as Resources in Information Security Education and Courses. In *Proc. of the Informing Science and IT Education Conference*, 2008.

[7] J. Matherly. Shodan. https://www.shodan.io/.

[8] I. Mokube and M. Adams. Honeypots: Concepts, Approaches, and Challenges. In *Proc. of the 45th annual southeast regional conference*, 2007.

[9] D. Nardella. Snap7. http://snap7.sourceforge.net/.

[10] L. Rist et al. Conpot. http://conpot.org/.

[11] C. Seifert, I. Welch, and P. Komisarczuk. Taxonomy of Honeypots, 2006.