Elektrotechnik & Informationstechnik (2018) 135/3: 264–269. https://doi.org/10.1007/s00502-018-0616-5

Malware propagation in smart grid monocultures

P. Eder-Neuhauser, T. Zseby, J. Fabini

Smart power grids require a communication infrastructure to collect sensor data and to send control commands. The common trend for cost reduction influences the architecture, implementation, networking, and operation of smart grid devices. Whereas hardware and software reuse are imperative for vendors to lower device costs, utility companies substantially decrease their operational costs by deploying a homogeneous device base. Thousands of smart meters that feature identical hardware, firmware, and software, are one main prerequisite for automated maintenance, support, and device replacement. However, these cost savings create optimum conditions for malware propagation and infection in the grids' control networks.

In this paper we show how monocultures in device types can lead to critical situations if malware exploits a common vulnerability. Although we assume that classical defensive measures, e.g., firewalls, virtual networks, and intrusion detection, are in place, we argue that new or unpatched vulnerabilities cannot be ruled out and may lead to a very fast distribution of malware in large parts of the smart grids' control network. Besides showing how fast malware can spread in device monocultures, we also discuss effective defensive measures that can support utility companies in preventing or containing malware distribution.

Keywords: malware attacks; smart grids; communication networks; network security

Malware-Ausbreitung in Smart Grid-Monokulturen.

Intelligente Stromnetze benötigen Kommunikationstechnologien, um Sensordaten und Kontrollinformationen zu übertragen. Der modulare Aufbau von Hardware, Firmware und Software sowie deren teilweise Wiederverwendung in verschiedenen Komponenten des Smart Grids ermöglichen eine Senkung der Herstellungs- und Investitionskosten. Je geringer die Anzahl ausgerollter Hardware-, Firmware- und Softwareversionen im Feld ist, desto geringer sind die zu erwartenden Betriebskosten wie etwa für automatisierte Updates, Gerätewartung, Geräteersatz und Schulung. Diese finanziellen Anreize haben eine äußerst homogene Gerätebasis im Smart Grid zur Folge. Das führt zu optimalen Bedingungen für die Ausbreitung von Malware in Smart Grid-Kommunikationsnetzen.

Schlüsselwörter: Malware-Attacken; Smart Grids; Kommunikationsnetze; Netzwerksicherheit

Received February 19, 2018, accepted May 18, 2018, published online June 5, 2018

1. Introduction

Modern smart grids consist of numerous devices that must be managed and controlled. Commonly one single authority, the utility company, administrates a huge number of field devices through its control centers. However, this huge number contrasts with only few deployed device types like, e.g., one smart meter type and one gateway type for a specific subnet. Reasons for deploying only few device types include, but are not limited to (a) requirements of a particular grid operator that are fulfilled by few vendors and devices, (b) proprietary protocols or extensions that question interoperability, (c) national regulations that lead to different feature sets, and (d) maintenance, replacement and cost considerations. For instance national regulations constrain the minimum reading intervals, the ability to switch power off and on remotely, and many other features of smart meters. Moreover, testing new firmware releases for these smart meters on functionality, interoperability, or compliance with national regulations is a demanding task in terms of time, effort, and cost. This aggregation of social, technical, and legislative reasons result in utilities creating monocultures or groups of monocultures in their network. This offers unique incentives and opportunities for attackers who can exploit one single vulnerability, or very similar vulnerabilities in common device types or reused software frameworks across the entire hierarchy to infect, control, and abuse a huge set of devices. For instance, the recent Spectre attacks, cf. [11, 12], confirm that as little as sharing the same processor family can result in common vulnerabilities of distinct devices.

Network segmentation can help to limit malware spreading once a device has been compromised. Nevertheless, it may fail, e.g., due to misconfigured firewalls, or unforeseen propagation paths. This can have catastrophic consequences if malware propagation remains undetected and malware can roam through an entire population of nodes in critical infrastructures.

The main contributions of this paper include a model on how malware propagates in critical infrastructure monocultures if network segmentation fails. We consider the stages that malware will transition through while infecting the key points in the network and the countermeasures that utilities can adopt to defend themselves during these stages. The proposed attack model considers three representative malware categories and discusses their impact in critical infrastructures. The findings are based on simulations and results are

Eder-Neuhauser, Peter, TU Wien, Institute of Telecommunications, Gußhausstraße 25/E389, 1040 Vienna, Austria (E-mail: peter.eder-neuhauser@nt.tuwien.ac.at); **Zseby, Tanja,** TU Wien, Institute of Telecommunications, Gußhausstraße 25/E389, 1040 Vienna, Austria (E-mail: tanja.zseby@tuwien.ac.at); **Fabini, Joachim,** TU Wien, Institute of Telecommunications, Gußhausstraße 25/E389, 1040 Vienna, Austria (E-mail: joachim.fabini@tuwien.ac.at)

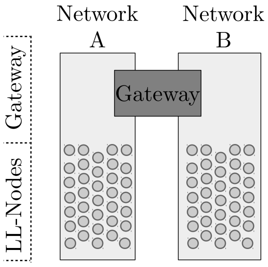

Fig. 1. Network Topology of 2 networks, connected through a gateway

presented using an exemplary simulation run for a hypothetical but representative topology.

This paper is structured as follows. Section 2 introduces the proposed network topology and the corresponding communication model. The attack model is discussed in Sect. 3. We simulate malware propagation using one example for a monoculture network (described in Sect. 4) and discuss the respective failure of network segmentation in the results in Sect. 5. Section 6 concludes the paper.

2. Network topology and communication model

Figure 1 illustrates the network topology as basis for subsequent discussions and simulations. It consists of two networks, with each 49 low level (LL) nodes (i.e., $2 \cdot 7 \cdot 7$ nodes in a symmetric setup, roughly representing two city blocks) and one connecting gateway. The LL-nodes represent the last controlling instance that operators can access before entering customers' premises. We argue that with the evolution of Smart Grids the LL-nodes will act as proxies and interfaces through which operators can control services such as decentralized power management or demand side management within customer's premises.

The connecting gateway node represents a local control entity that is located in a central location such as the transformer station and is in charge of the management of data flows. Once the gateway is infected an attacker could enter the overlain control network (not simulated) of the utility company.

Our setup represents two subnets of different energy infrastructures which are connected via network equipment in order to benefit from shared resources and control technologies. Such a scenario, of interconnecting different networks for optimizing control is commonly found in the literature, for instance in [2, 4, 7, 13]. Network A and B could represent either two power grids or grids for different utilities, e.g., network A could represent a gas grid and network B a power grid.

Our sample topology contains only a small number of devices compared to typical large grid installations, which, e.g., include approximately 1.6 million smart meters for the city of Vienna, Austria [17], but is sufficient to show the effects of monocultures in smart grids.

Figure 2 illustrates the used communication model with legitimate and malicious communication that is sent to the gateway. LL-nodes can only communicate with the gateway (a medium level (ML) node) but not with other LL-nodes. The gateway is aggregating and analyzing all data, thus, there is no need to send data from one LL-node to another in the other network. The gateway does not route information to another destination in the other network, except to the control center which is positioned hierarchically above the ML-node, being outside of the simulation scope.

All nodes can initiate the communication to the gateway and the gateway can poll LL-nodes for additional data or push firmware updates. Figure 2 shows the legitimate communication and also illegitimate communication of infected nodes that are targeting other nodes in their own network. Communication attempts across the infrastructure should be mitigated with appropriate defense measures, e.g., firewalls or intrusion detection systems. But these systems can sometimes fail because of technological constraints or vulnerabilities in the network protocol stack like, e.g., in the case of wireless ad-hoc networks.

Moreover, active scanning activities by infected LL-nodes can be detected at the gateway if it is capable of intrusion detection functions and listening for anomalous traffic.

3. Attack model

Network segmentation is crucial to prevent malware from spreading in the network. One example of a rogue broadcast message that originated inside a gas grid control network, proliferating into a power grid control network and resulting in a denial of service, is presented in [2]. Although this example was an accident and not an intentional malicious attack, it shows that improperly segmented networks can, in this case by a configuration error, lead to large-scale node failure and possibly to catastrophic consequences. In theory a gateway should strictly segment the two networks but the ML-node may be affected by vulnerabilities or misconfiguration, too.

In our work we simulate malware propagation that could shut down all nodes or mount similar denial of service attacks if all nodes are infected.

Our attack model is based on a malware type named *endemic malware*, as introduced in [6]. We argue that the assumption of such automated self-propagating malware is reasonable because manual infection of all nodes in a large smart grid does not scale well [5]. Endemic malware aggregates capabilities of advanced existing malware types that are capable of extracting network information from the infected host, optimize the scanning strategy to the network setup, and obfuscate the propagation behavior. It uses a permutation hitlist scanning strategy that copies the target hitlist including the scanned nodes among malware children in order to minimize rescans (i.e., scanning nodes again that already have been scanned). Endemic malware is highly modular, requires a high level of development effort, and therefore could also have the capability to utilize multiple vulnerabilities to infect nodes. Moreover, it opens dedicated connections to its victims, making the malware detectable by connection-based anomaly detection in the network.

Nevertheless, capabilities that enable the malware to stay undetected on a host for a period of time make endemic malware a challenging adversary. Therefore, we assume that host based defenses, e.g., virus scanning software may not detect this advanced malware type, as was the case initially with the Stuxnet worm and its cousins [1]. The goal of endemic malware in our simulation is to infect all nodes in both infrastructures to commence different attack types, e.g., shutting down all nodes, shutting down selected nodes, or building a botnet for denial of service attacks, cf. [6]. Infected field nodes can disguise themselves as gateways toward other field nodes to infect victims laterally. A similar behavior has been observed for instance in the Flame malware [10]. The endemic malware is described in detail in [6].

We use the endemic malware as basis for our simulations. Additionally, as a comparison we show the capabilities of two other malware types, namely, pandemic and contagion malware, cf. [6], and compare them with the features of the endemic malware. However, we do not simulate them.

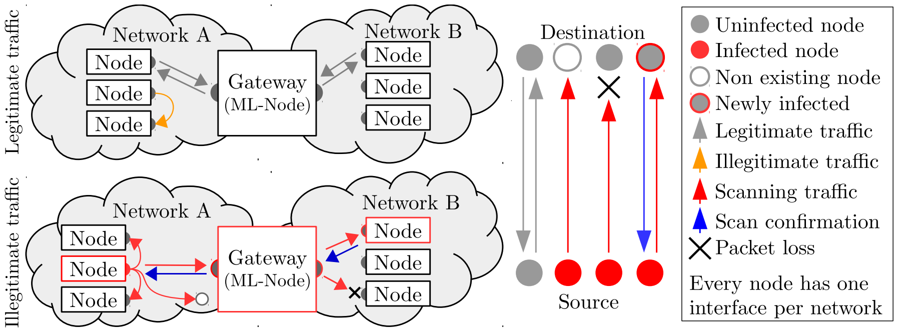

Fig. 2. Communication traffic and patterns

Furthermore, we discuss *pandemic malware* which represents aggressive malware types that use random scanning to discover new victims in a brute-force manner, enabling defenders to quickly identify it in the network. Pandemic malware implementation lacks sophisticated features, is generally simple and involves a relatively low development effort. These characteristics facilitate its use by a broad range of user groups. With the source code of some variants [14] being accessible on the Internet as a template, even less skilled attackers can modify and implement their own version. However, besides the aggressive scanning, victims are contacted by opening a dedicated connection, which should also be detectable by connection-based intrusion detection.

The last malware type, called *Contagion malware*, represents highly advanced malware that obfuscates all aspects of its presence, cf. [6]. Contagion malware exploits application layer vulnerabilities [8] to propagate hidden within established legitimate TCP connections instead of opening new dedicated connections to the victims. This is why it is invisible to connection-based anomaly detection. Such a malware requires defenders to inspect every legitimate connection for anomalies, e.g., anomalous peaks of packets, anomalous packet timings, or anomalous packet size variations inside a flow. With encrypted communication channels at network or transport layer, such anomaly detection becomes a challenging task.

We define the starting point of the infection at one compromised LL-node as patient zero. It represents, e.g., an infected smart meter, which can initially be compromised via unpatched or new vulnerabilities. We assume that the utility operates a device monoculture and that an exploitable set of similar vulnerabilities exists throughout the entire population and across all hierarchy levels of the deployed devices. This is typically the case when all devices are manufactured by one vendor and operate the same firmware release, or reuse parts of the same hardware or software like, e.g., operating system, software development framework, or common libraries. Recently reported vulnerabilities, which have persisted over generations of CPUs (cf. [11, 12]) confirm that exploitable hardware monocultures are highly realistic assumptions in today's systems. The alarming conclusion is that a sufficient condition for the establishment of a huge cross-vendor device monoculture involves as little as two smart meter vendors to use the same CPU family in their product designs. Our fictional vulnerability affects all network nodes and allows remote code execution and administrator rights upon infection.

Figure 3 illustrates the different capabilities and characteristics of pandemic (yellow), endemic (blue), and contagion malware (red), as introduced in [6].

The figure depicts features, characteristics, capabilities, and particular strengths of the different malware types. The more sophisticated a malware feature, the more distant the corresponding point is located from the diagram origin. Therefore, assuming equal weighting for all features, a larger area represents a greater threat to defenders.

General features, e.g., the development effort shows that pandemic malware is simple, therefore, accessible to a larger group compared to the more advanced malware types. Endemic and contagion malware require increased resources, which may be a drawback for the attacker, thus, a benefit for defenders. Moreover, increased development effort coincides with advanced on-board defense features against detection that are available in modular extensions. This increased effort pays off and represents a benefit for attackers in terms of improved attack capabilities.

Increased stealthiness features, which support reduced network scanning and stealthy malware propagation in networks, on one hand decrease propagation speed and frequently coincide with increased development effort. On the other hand, advanced on-board capabilities enable the malware to optimize resource consumption in the network and/or on the host, supporting advanced features like, e.g., additional attack vectors, obfuscation capabilities, or advanced scanning strategies. The insert in Fig. 3 differentiates between general features, network domain features, and the host domain features.

An additional malware categorization is with respect to pre-infection, the initial propagation phase, vs. post-infection, the operational phase. Pre-infection includes all actions that happen in the first few moments of a malware lifetime. This is the instant in time when the malware must propagate itself autonomously in a network, as simulated in our example. This step is intentionally automated due to the assumed large scale smart grid.

The operational phase, however, represents the malware's capability to stay hidden (unobserved) and persistent for an extended period. This includes low CPU usage by the malware such that defending software may not identify CPU overload for a system that is supposed to operate within specifications. Failure of the malware to do so opens opportunities for defending software that can detect either processes on the host that act suspiciously or excess network traffic for malware command and control (C&C) or propagation activities.

Other host based features that correlate well with the development effort include the malware's payload structure. A monomorphic payload represents a simple construct that may change in size but produces similar signatures. Therefore, it can be detected reliably whenever heuristic signatures are available. A polymorphic payload complicates detection by scrambling its shape and size through encryption. Still, decrypted payloads will produce identical signatures on the local drive of the host, being detectable by heuristic methods. Malware featuring metamorphic payload requires the higher

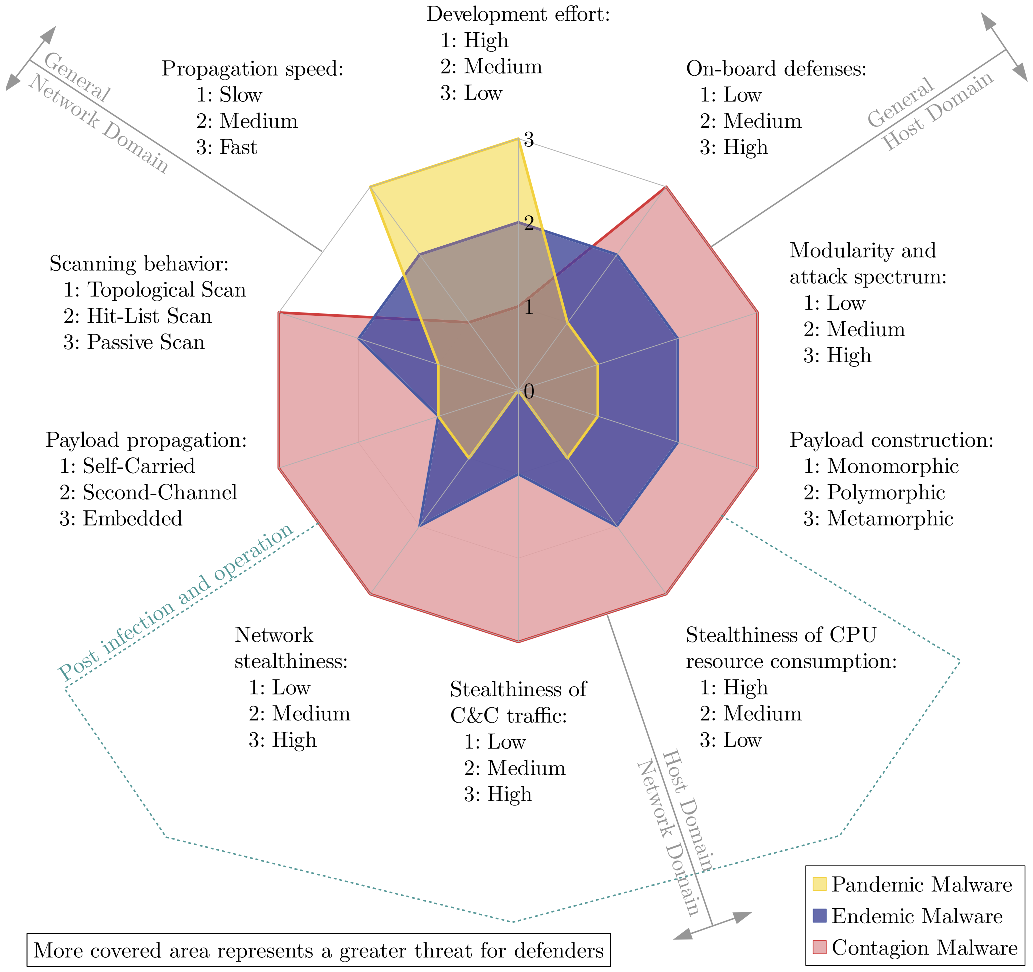

Fig. 3. A threat matrix for smart grid enabled malware, its capabilities and defensive efforts

development effort, varying in size, shape, encoding, and encryption. Moreover, recompilation on the host system can be used to obfuscate any trace of the payload's presence, cf. [9].

We use these three malware types, cf. [6], as the starting base and simulate one example of the endemic malware.

4. Simulation model

Our simulations are based on the ns3 network simulation environment [15], version 3.26. We simulate two mesh networks that use the Optimized Link State Routing (OLSR) protocol [16] for lower layer routing. The two mesh networks are connected via a gateway, which has two network interfaces, one for each network, whereas each LL-node has one single network interface. At IP layer, the LL-nodes are supposed to communicate only with the gateway interface, although messages may pass several OLSR nodes during the lower-layer routing process.

Our malware model consists of a self-carried propagation model including the dropper and the payload which is unpacked on the infected host nodes [6].

5. Network segmentation and monocultures

This section discusses the failure of network segmentation measures and elaborates on the drawbacks of homogeneous infrastructures, i.e., monocultures. The simulation is restricted to endemic malware, results and conclusions being applicable to other malware types, too, as detailed in [6]. We omit simulations of the pandemic and contagion malware types but discuss the differences between the three malware types.

First, network segmentation, by e.g., firewalls or virtual networks, is often used as an effective measure to contain malware propagation, cf. [6]. In case proactive measures fail to protect the gateway, e.g., because of a new or unpatched vulnerability, both parts of the network may be affected. Monocultures therefore can support malware propagation. Figure 4 shows a simulation of two networks of the same size, i.e., each network has 49 nodes. Since this simulation is only an example, the results are specific to the chosen network size and parameters.

Five key events in the simulation time-line in Fig. 4 characterize any simulation run. Parameters that influence the timing of these events include malware behavior, network size and network topology. When computing the timing of these events for distinct malware types and/or for distinct networks, the results can help to compare the performance of specific malware in specific network settings. In particular, the timings can help to rate and compare the robustness of specific network topologies when attacked by specific malware types.

These five key events are:

- *Patient zero:* Infection time of the first node, by definition at simulation start.

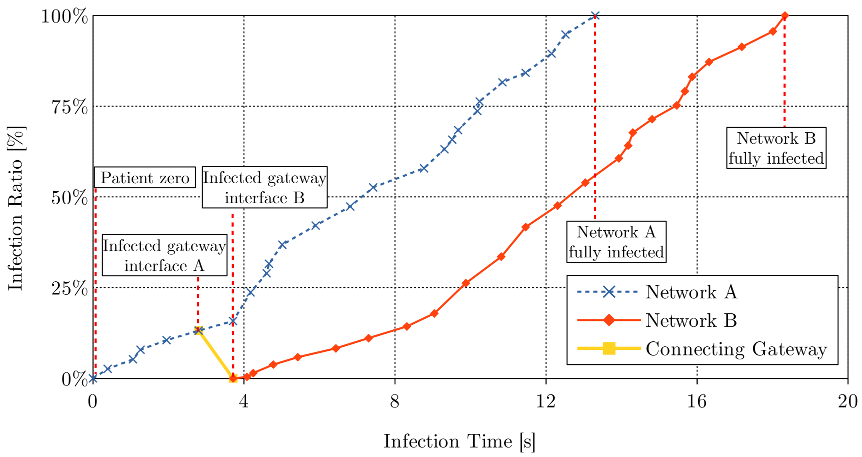

Fig. 4. Network segmentation and monocultures

- *Infected gateway interface A:* Up to this point, infected nodes in network A are scanning for the gateway, infecting several local nodes in the process. Once the gateway is infected lateral propagation into network B is possible if the corresponding network interface of the gateway can be utilized. Already 20% of the nodes in network A are infected, i.e., 10% of all nodes in both networks.
- *Infected gateway interface B:* The foothold in the target network B is established, i.e., the gateway is fully infected (interface A and B) and scanning in network B can commence.
- *Network A fully infected:* Full infection of network A is accomplished, i.e., 50% of all nodes in the whole setup. Assuming devices in network B to feature the same vulnerability, the malware already started infecting network B. In total (network A and B) 77% of all nodes are infected.
- *Network B fully infected:* Full infection of both networks (100% of all nodes) is achieved. All types of attacks (e.g., selective disruption or destruction, full disruption, etc.) can commence against both networks A and B.

In case the gateway is capable of preventing the lateral propagation of malware into network B, the potential damage would be reduced to a smaller set of nodes, in our case to 50% (all nodes of network A). Although evident in this scenario, we emphasize once more the importance of properly implementing defensive measures in such neuralgic nodes in the network setup.

Monocultures can produce challenging vulnerabilities for a large number of devices, as seen in several cases [11, 12]. Therefore, the protection of critical assets, such as the central gateway, is of utmost importance, serving as a last line of defense before malware can spread throughout the network or even to the control center (not simulated).

Moreover, adding to the aforementioned defensive measures, i.e., firewalls and virtual networks, which can mitigate such attacks, additional measures may be necessary, cf. [3]. Anomaly detection can in theory identify malware before it infects the gateway. In this case, aside from a large number of defensive measures that can be implemented, the gateway could preventively shut down the network interface, i.e., disconnect the network segment, operate in a fallback mode to protect the other network, or warn other LL-nodes from malicious intents and to operate in a fallback mode themselves. Although this may be impractical in critical infrastructures, it can present an emergency solution to protect a central asset by jettisoning a small part for protecting the whole. Furthermore, fully separating such infrastructure networks from any Internet facing services may prevent malware attacks, yet, cannot provide any guarantee because field devices can still be infected manually.

Depending on its criticality, an *importance factor* could be assigned to these infrastructures. E.g., a power grid outage having the most severe consequences in terms of impact, may be assigned the highest priority to the power grid. Networks with higher priorities should, additionally to increased protective measures, be operated from inside out [3]. This means that control of critical components, e.g., the firewall or IDS, must not be delegated to a central control entity that is located outside of the critical network such as a controller instance that is connected to the enterprise network. An attacker that manages to control this instance, also controls all connected, subjacent, critical networks.

In the following we elaborate on the differences of endemic malware to pandemic and contagion malware.

Pandemic malware has a more aggressive scanning strategy than endemic malware, thus, is more likely to be detected by intrusion or anomaly detection systems. Pandemic malware has less advanced features due to the low development effort and should be defeated by defensive measures that are already in place to protect from endemic malware.

Contagion malware on the other hand is using hidden communication inside legitimate network flows, does not scan the network actively, and implements advanced on-board features. Compared to endemic malware, contagion malware is more challenging to detect when infecting the gateway, which is why advanced host based defense measures are needed. However, all the key events in Fig. 4 apply to contagion malware, as well.

6. Conclusion

This paper shows that vendors and utilities have huge incentives to implement their control infrastructures as monocultures in terms of both, hardware and software. The same hardware is mainly used to lower development-, deployment-, and replacement costs. Using identical software decreases operational and maintenance costs. However, the huge number of identical nodes in such networks supports fast and efficient propagation of malware once a vulnerability is found.

When analyzing different malware types we conclude that the simple but aggressive pandemic malware is the least challenging adversary to defeat. Similar network based defense measures may also suffice to defeat the endemic malware type. However, endemic malware carries more advanced on-board features that include obfuscation techniques and can present a challenging problem for host based detection methods, should it manage to infect a host device. In this case, we argue that the gateway, being the node of central importance in the presented setup, must implement all of these features. Contagion malware uses hidden communication to propagate to its victims, which is why the gateway, being the only legitimate communication partner of all nodes, is infected first. The likelihood is extremely low that such a malicious payload can be detected in transit. Accordingly, host based detection must be much stronger, involving complex heuristics and advanced real-time anomaly detection in its attempt to protect the gateway.

Our detailed analysis confirms that monocultures support fast malware spreading, in particular if the communication networks are not configured and segmented properly. As this can have catastrophic consequences, critical networks should not be connected to shared network resources like enterprise networks. Advanced security measures like anomaly detection are recommended to be implemented on neuralgic nodes like gateways to detect and prevent malware from infecting neighboring networks. However, the downside of this strict separation between enterprise and control domain is a substantial increase in investments for planning, component replication and segmentation. Economic considerations being one of the main drivers of today's utility networks, we anticipate that major incidents must and will happen before utility companies' management will reevaluate the true costs of malware infection. We argue that this cost reevaluation is an essential prerequisite for balancing the true costs of potentially catastrophic grid failures against investments into security measures that the protection of critical infrastructures deserves.

Acknowledgements

Open access funding provided by TU Wien (TUW).

References

1. Bencsáth, B., Pék, G., Buttyán, L., Félegyházi, M. (2012): The Cousins of Stuxnet: Duqu, Flame, and Gauss. Future Internet, 4(4), 971–1003.
2. Christiner, G. (2013): Die Rolle der APG für die Stromversorgungssicherheit – Nationale und Internationale Herausforderungen. Tech. Rep. 20903, E-Control, Austria.
3. Eder-Neuhauser, P., Zseby, T. (2017): The art of defending critical infrastructures. In ISGT-Europe, IEEE conference, Turin, ITA.
4. Eder-Neuhauser, P., Zseby, T., Fabini, J. (2016): Resilience and security: a qualitativesurvey of urban smart grid architectures. IEEE Access, 4, 839–848.
5. Eder-Neuhauser, P., Zseby, T., Fabini, J. (2017): Malware propagation in Smart Grid networks: simulation and comparison of three malware types. J. Comput. Virol. Hacking Techn., in press.
6. Eder-Neuhauser, P., Zseby, T., Fabini, J., Vormayr, G. (2017): Cyber attack models for Smart Grid environments. Sustain. Energy Grids Netw., 12C, 10–29.
7. Federal office of civil protection and distaster assistance (2015): Kritis – sector: energy. White paper, Germany.
8. ISO/IEC Std 7498-1:1994 (1994): Information technology – open systems interconnection – basic reference model. International standard.
9. Kamluk, V., Gostev, A. (2016): Adwind – a cross plattform RAT. White paper V. 3.0 #Adwind, Kaspersky Labs.
10. Kaspersky Labs (2016): The Flame: questions and answers. [Online] Available: https://securelist.com/blog/incidents/34344/the-flame-questions-and-answers-51/.
11. Kocher, P., Genkin, D., Gruss, D., Haas, W., Hamburg, M. (2018): Spectre attacks: exploiting speculative execution. Preprint, arXiv:1801.01203.
12. Lipp, M., Schwarz, M., Gruss, D., Prescher, T., Haas, W. (2018): Meltdown. Preprint, arXiv:1801.01207.
13. Marinos, L. (2013): Smart Grid threat landscape and good practice guide. Tech. rep., European network and information security agency (ENISA).
14. Nativ, Y. (2018): theZoo: a repository of LIVE malwares for your own joy and pleasure. [Available] Online: https://github.com/ytisf/theZoo, original-date: 2014-01-09T18:55:35Z.
15. NS-3 Consortium (2016): NS-3 Consortium. [Online] Available: https://www.nsnam.org/.
16. olsr.org (2004): Open link state routing protocol – man page. [Online] Available: http://www.olsr.org/docs/olsrd.conf.5.html.
17. Wien Energie GmbH (2012): Smart metering und smart cities. [Online] Available: http://arge.ph-noe.ac.at/fileadmin/fwz/etech/Energiesysteme/2_smartmetering_smartcities.pdf.

Authors

Peter Eder-Neuhauser
received the M.Sc. degree in energy engineering from the University of Applied Sciences Technikum Wien, Austria. He is currently pursuing the doctoral degree with the TU Wien. He is working on smart grid security with a focus on malware containment in smart grid ICT.

Tanja Zseby
is a full professor of communication networks at the Faculty of Electrical Engineering and Information Technology at TU Wien. She received her diploma degree (Dipl.-Ing.) in electrical engineering and her doctoral degree (Dr.-Ing.) from TU Berlin, Germany. Before joining TU Wien she led the Competence Center for Network Research at the Fraunhofer Institute for Open Communication Systems (FOKUS) in Berlin and worked as visiting scientist at the University of California, San Diego, USA.

Joachim Fabini
received the Dipl.-Ing. degree in computer sciences and the Dr. techn. degree in electrical engineering from TU Wien. After five years of research with Ericsson Austria, he joined the Institute of Telecommunications, TU Wien, in 2003. He is a Senior Scientist with the Communication Networks Group with research focus on active measurement methodologies.