Steganography for Cyber-physical Systems

Steffen Wendzel[1,2], Wojciech Mazurczyk[3] and Georg Haas[1]

[1]*Worms University of Applied Sciences, Germany*
[2]*Fraunhofer FKIE, Germany*
[3]*Warsaw University of Technology, Poland*

Abstract

Cyber-physical Systems (CPS) have raised serious security concerns and thus have been subjected to intensive security research lately. Recent publications have shown that there is a potential to transfer hidden information through CPS environments. In comparison to these existing studies, we demonstrate that CPS cannot only be used to covertly transfer secret data but also to store secret data. Using an analogy to the biological concept of animal scatter hoarding behavior we exemplify CPS secret data storage using automated buildings.[1]

Keywords: Cyber-physical Systems (CPS), Internet of Things (IoT), Steganography, Covert Channels, Information Hiding, Smart Home, Smart Building, BACnet.

1 Introduction

Cyber-physical Systems (CPS) are defined as *integrations of computation with physical processes* [7] and notable examples include smart homes and buildings, industrial control systems (ICS), electronic health-care (e-Health) equipment, wearables and smart cars. These systems are of an increasing

[1]This publication is an extended version of [18].

Journal of Cyber Security, Vol. 6_2, 105–126.
doi: 10.13052/jcsm2245-1439.621

importance for today's Internet-connected world, influencing every area of modern living, interaction and automation.

However, being integrated in several sensitive areas, e.g. in a form of smart objects or smart buildings, many CPS have access to highly sensitive data. For example, presence sensors in buildings can reveal information about the location of people inside buildings [10] and a temperature sensor can reveal information about the building itself (room temperature, humidity, pressure in pipes, location of an elevator etc.). In other words, CPS data leakage can reveal information about the CPS *environment* as well as about the CPS *themselves*.

While few publications have already studied the technical details and risks related to the (steganographic) data *leakage* in CPS, no work is available that analyzes the other side of information hiding, namely steganographic data *storage*.

To better visualize what we understand under the term *steganographic data storage* and how it can be utilized, the following scenario is devised. Let us consider two spies who want to exchange classified information but they do not want to communicate or meet directly. In this scenario they establish a rather uncommon way to exchange data: every two weeks they travel by plane to different countries but they use the same airport as their intermediate stop (a transfer). The first of the spies has its transfer in the morning and the second in the evening so that they never meet in person. Both spies are looking just like ordinary business passengers traveling with their laptops or smartphones and using them while waiting for the transfer. If we assume that spies are able to exploit vulnerabilities in the airport's building automation system they can influence the sensors, actuators, controllers, etc. in such a way that these building components will become their *secret data storage*.

In advance, both spies decide which subset of the building automation devices they will use to store their secret data by scanning the automation devices during a first visit or by pre-defining a reproducable scheme (e.g. only utilizing certain devices and encoding a secret message based on ID numbers of devices).

Access to building automation devices can be achieved by wiretapping networked devices that are located in areas facing little monitoring, such as motion sensors in restrooms that are used to trigger the light, or by trying to access the wireless communication of the building automation equipment (e.g. EnOcean or ZigBee). The information about the devices to be used is called

the *steganographic key*. Such a key must be exchanged in advance between the spies.[2]

To store and transfer secret information, the first of the spies embeds covert data using CPS steganographic methods in the morning while the second spy extracts it in the evening. The time range between embedding and extracting is intentionally chosen within the same day so the secret data is not heavily disrupted.

It is worth noting that the CPS steganographic data storage scenario depicted above holds many analogies to *hoarding/caching* known from the Animal Kingdom which is a behavior to store food for a later use in locations hidden from the plain sight of both conspecifics and members of other species. Hoarding strategies are typically divided when considering spatial distribution into [13]: *larder hoarding* and *scatter hoarding*. The former relates to hiding (a large number of) food items in a small number of caches (called "larders") which are typically placed near the location where the animal lives. The latter relies on distributing food over many widely spaced caches with only few items in each (called "scatters"). In this case the caches are scattered throughout wider area.

Both hoarding approaches have pros and cons. Larder hoarding includes easy creation of caches and recovery of the food but they have to be actively defended against theft which costs time and energy. They may be also easier to detect by competitors due to the stronger odors present nearby. On the other hand, scatter hoarding involves higher energetic costs as the recovery of food items requires more traveling and better memory to remember cache locations. It also increases a risk on encountering the predator due to the longer exposure. It must be also noted that typically such caches are not defended individually.

Considering the biological concepts presented above, the "spies scenario" that we provided earlier, and also by taking into account the characteristics of smart buildings, we believe that the correct strategy to enable covert data storage in CPS is to apply "steganographic scatter hoarding". Scatter hoarding is especially suitable as CPS components do not provide larger storage potential which would be required for larder hoarding.

[2]In case a CPS would be used by only one steganographer, the key does not need to be exchanged with another participant. It also must be noted that especially wired communication in building automation and other CPS is usually lacking even basic security features, making it easy to wiretap the un-encrypted, un-authenticated communication.

Scatter hoarding for CPS means that only small modifications of the CPS will be allowed but they will be applied to numerous components of the system e.g. to actuators, controllers, sensors, etc. Additionally, CPS components should be carefully selected for the steganographic storage to avoid them being regularly modified, e.g. by a human user. This will potentially increase resiliency and undetectability compared to the situation in which a major change to the CPS would be performed. Moreover, the amount of secret data per single 'steganographic cache' should be limited so even when some caches are compromised (e.g. due to human interaction) it is still possible to successfully extract the secret data from the remaining caches. Potentially, redundant encoding can recover lost data in such a case.

In this perspective, we study whether CPS can be abused as a steganographic storage, i.e. whether hidden information can be stored there and successfully recovered. We use one of the most common CPS types, namely *building automation systems* (BAS), to perform our study. We analyze the amount of data that can be stored in a BAS under different conditions as well as we analyze the influences on the steganographic storage, i.e. the robustness of the applied hiding methods. To author's best knowledge, this is the first approach to evaluate data hiding in CPS for the covert data storage purposes.

The remainder of this paper is structured as follows. Section 2 covers related work. We introduce fundamentals of Information Hiding and Building Automation Systems and the general concept for data hiding in CPS in Section 3. Two approaches for covert data storage in CPS are introduced in Sections 4 and 5. Finally, Section 6 concludes and provides an outlook on future work.

2 Related Work

Several publications study privacy and surveillance aspects of smart homes, smart buildings and other types of CPS, e.g. Möllers and Sorge study whether user presence in homes can be detected using inter-arrival times of network data in home automation [10] and Mundt et al. show that users can be identified in smart buildings when sensor data is evaluated [11].

Few publications address Information Hiding in Cyber-physical Systems. In 2012, Wendzel et al. [17] identified the existence and scenarios for covert communication in BAS. In their scenario, covert channels (stealthy communication channels) can leak sensitive data and secretly transfer confidential information through BAS networks. The authors introduced

a *building-aware active warden*, which provides multi-level security (MLS) for a BAS environment to prevent read-ups and write-downs using the Bell-LaPadula model. In 2015, three relevant works were published. Two additional covert channels that rely on the modulation of transmission power and modulation of sensor data in pervasive computing were described by Tuptuk and Hailes [16]. Moreover, Howser presented work on data leakage in CPS, proposing an algorithm that protects against data leakage using MLS [3]. Recently, Tonejc et al. have shown that some types of covert channels in BAS networks can be detected using unsupervised machine learning methods [14]. Finally, an approach which applies image steganography to secure data that is transmitted within the IoT against eavesdropping has been proposed by Yin et al. in [19].

3 Fundamentals of Information Hiding and Building Automation

3.1 Information Hiding and Its Relationship to Nature

The term information hiding covers a wide range of data concealment techniques aiming at embedding a secret message in such a way that a third-party observer is unaware of its presence (see, e.g. [20] for a review of methods developed throughout the years). Steganography is a well-known form of information hiding. In this case, the covert data is placed inside carefully chosen and innocent-looking carriers [9].

It must be noted that the inspiration for information hiding techniques is strongly related to phenomena observable in nature. Evolution has proven long ago that abilities to disguise can serve as a protection and can significantly improve chances for survival, especially as resources that are needed for survival are very often limited and competed for. For instance, blending into the surrounding environment using camouflage techniques allows the exact location of the organism to remain ambiguous. Moreover, it must be noted that in nature as well as in digital environments two broad groups of information hiding mechanisms can be distinguished which enable [9]:

1. *secret data communication*: includes methods to exchange messages in a covert manner.
2. *secret data storage*: incorporates techniques to hide data in seemingly innocent ways, so that it is difficult for anyone besides the owner to locate or to retrieve the secret content.

Below, we present examples of both, covert data communication and storage, in nature and in digital environments.

In nature, for Philippine Tarsiers (*Tarsius syrichta*), which are small nocturnal primates, it was revealed that they have a high-frequency limit of auditory sensitivity (ca. 91 kHz) and are also able to vocalize with a high dominant frequency (ca. 70 kHz). Such an ultrasonic communication is utilized by them to communicate privately in a covert manner and this channel remains undetectable by predators, prey and potential competitors [12]. In current communication networks data hiding by modifying the content or the characteristics of network traffic is one of the most recent examples of such type of information hiding [9].

In nature, covert storage of important resources is performed by many animal species. As already mentioned in the 1, the caching/hoarding of food items at times of their high availability is a good strategy in order to rely on these reserves during periods of food scarcity. Examples of the organisms which rely on such behavior are Willow (*Parus montanus*) and Crested Tits (*P. cristatus*), which store seeds in a scattered distribution within their territory during the autumn [5]. In the digital environments, data hiding in digital media (e.g., images) is a most notable example of analogous technique.

From this perspective, CPS environments can be viewed as a promising next step in the evolution of the covert data storage scenario and it represents a case similar to scatter hoarding in nature. In this case, a *steganographer* places hidden information in the CPS environment by slightly modifying some of its components. The data is hidden in a way that a potential adversary interested in revealing it (called *steganalyst*) is unable to easily detect its presence.

3.2 Building Automation Systems and Information Hiding

Like in the case of other CPS, BAS comprise actuators, sensors, controllers, and monitoring equipment. In a typical BAS, sensors and actuators outnumber any other device type, making sensors and actuators an interesting target for information hiding. Therefore, we decided to determine ways to store data using these devices. The next two sections discuss two different approaches that we found to be suitable; each of them providing a different performance and robustness. These approaches are:

1. **utilization of unused registers** in CPS equipment, which appears as a robust approach as unused registers would not be used by other components of the CPS and stored data is unlikely to get overridden

(this approach can be considered trivial and is integrated for comparison with the second approach),

2. **modulation of actuator states**, which is not trivial due to the fact that actuator states change and influence the physical environment, i.e. steganographic operations may not be robust and be easily detectable and thus need a reasonable storage strategy.

For hiding in unused registers, we use a temperature sensor called DS18B20 that can be typically deployed not only in BAS but in basically all types of CPS. We will apply modifications to the register values of the temperature sensor in order to hide data in it.

To address the modulation of actuator states, we store data using the ISO standard 16484-5 [4], called *Building Automation and Control Networking Protocol* (BACnet). BACnet is a communication protocol standard for BAS developed by ASHRAE with more than 900 vendors world-wide [1].

For both approaches mentioned above, we will first introduce the concept in detail, explain the test-bed setup, and then present results of the experimental evaluation.

4 Utilization of Unused Registers

The first approach for covert storage in CPS is to write data to the CPS component's registers which are currently not used. Using a simple device, a temperature sensor (Maxim Integrated Products, Inc. 1-Wire DS18B20 [8]), we write and read steganographic data to/from registers with a reasonable performance. We achieve the hidden data embedding by utilizing two unused registers with a size of 2×8 bits. The utilization of such unused data areas, e.g. in network protocol headers, is a common strategy for steganographic methods but was not applied to unused registers. Such unused registers are potentially available in several CPS components. Some of these devices, such as temperature sensors, can be massively deployed in a single larger CPS environment, making it easy to access a larger number of devices with the same technique.

4.1 Testbed

The DS18B20 uses a 1-Wire connection for communication and power-supply (parasitic mode). Each of the temperature sensors comprises two 8-bit alarm registers which we used to store hidden values. These alarm registers contain a lower and an upper warning threshold. A bus with multiple sensors can

be queried for sensors that have exceeded the warning threshold. The alarm register can also be used as general-purpose memory ([8], p. 7). The default value of these two registers is 0x4b46.

However, if many sensors are present, a strategy is required to determine the correct order of sensors to store/read steganographic data. For this reason, we use the internal unique serial number of each sensor to sort hidden data during writing and reading operations. The serial number is hard-coded by the manufacturer and can be read using the 1-Wire connection.

4.2 Evaluation

4.2.1 Performance evaluation

To measure the performance for reading and writing operations in case of the temperature sensor, we used an on-board functionality of the Arduino: to measure the elapsed time in $ms/\mu s$ we used `millis()` and `micros()`. The precision of our μs measurements is approx. $4\mu s$ due to system constraints. We first measured the time it takes to read addresses and the registers of the temperature sensors, while the number of sensors for our experimental scenarios varied (1, 2 and 4 sensors). For each scenario, 100 measurements were performed (Table 1). The reading performance for 1, 2 and 4 temperature sensors is almost equal. However, as addresses only need to be fetched once per reading, the overhead slightly reduces if more than one sensor is read.

Similarly, we performed measurements to evaluate the writing performance (Table 1). Therefore, a temperature sensor was written 100 times with the alternating values 0x0000 and 0xffff. The average writing time was $71.827\mu s$, i.e. writing took significantly longer than reading.

The measured reading and writing performance is high in comparison to other forms of steganography, e.g. a network covert timing channel [9]. However, analogously to scatter hoarding, several sensors with a few utilizable register bits each are required to store a few hundred bits of data, e.g. a cryptographic key. For instance, if between 4 and 16 unused register bits

Table 1 Per sensor writing and reading performance

Scenario	Avg. Time [μs]	Min. Time [μs]	Max. Time [μs]
Reading 1 Sensor	12.841	12.800	12.844
Reading 2 Sensors	12.804	12.784	12.806
Reading 4 Sensors	12.802	12.788	12.804
Writing 1 Sensor	71.827	71.800	71.834

could be written per utilizable device, storing a 128 bit key would require 8 to 32 utilizable devices and storing a 256 bit key would require between 16 and 64 utilizable devices.

4.2.2 Robustness

The datasheet for the temperature sensor shows that two 8-bit registers can be used to store alarm values (minimum/maximum temperature). Both registers accept values between –55 and +125 degrees Celsius, which results in the fact that not all eight bits of these registers can be utilized. However, the stored values are robust and remain unchanged as long as they are not actively overwritten by the user.

We performed an experiment in which we alternatively wrote the data 0x0000 and 0xffff to the alarm register. After each writing operation, we afterwards read the stored data to verify whether the data was corrupted. The experiment was performed in a loop for 180,000 times to see whether bits are corrupted. Our small experiment has shown no error (all data where retrieved correctly). This indicates that data may not require error-tolerant encoding.

4.2.3 Detectability

Currently, no system exists in practice to check for modified register values of the temperature sensor, making the likelihood of a detection approach very low. However, it would be easy to poll the register values on a regular basis and compare them to the standard value (0x4b46) to determine the *potential,* but heavily unlikely, presence of hidden data.

5 Utilization of Actuator States

Actuators are devices that, in comparison to sensors, do not measure but alter the state of a component in the physical world. For instance, an actuator can be a device to open or close a window, a heater that changes the temperature in a room, an elevator that lifts up goods or humans, or a water pump. The state of an actuator (e.g. the heating level or the angle of an opened window) can be polled by other devices. If a device sends a command with a desired value to an actuator, the actuator changes its value.

With actuator states, a BAS can be used as a covert data storage (Figure 1). In this case the steganographer modifies the value of an actuator to a desired value. For instance, a heating value of 80% could represent a binary “0”, a heating level of 79% could represent a binary “1”. The higher the number of

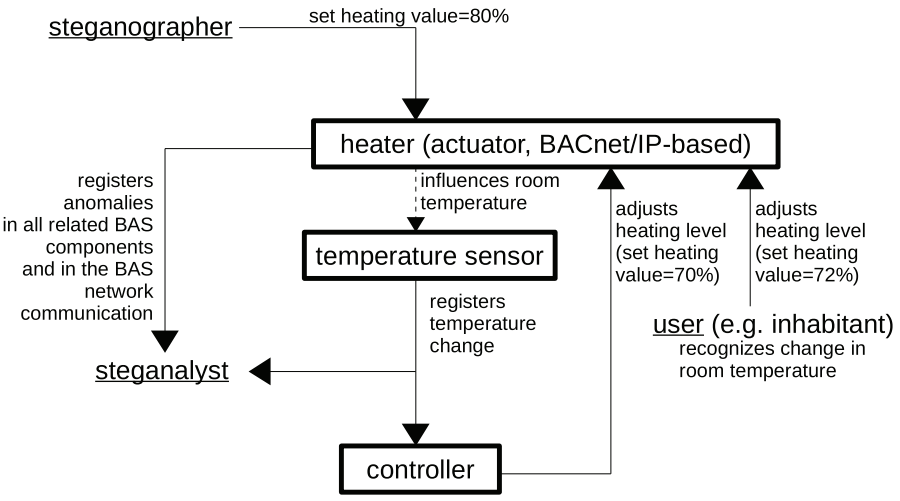

Figure 1 Hiding information in actuator values.

states an actuator can be set to, the larger the number of hidden data that can be stored.

However, each actuator influences the physical environment. Users (inhabitants, BAS operators) and sensors (e.g. temperature, humidity or motion sensors) can recognize changes in the physical environment. For instance, a temperature sensor may report a new value or an inhabitant will start to freeze. Determination of an actuator state change is independent of the cause of its change – be it because a user changed the heating level manually, be it because a controller adjusted the actuator due to a programmed logic, or be it because a steganographer set the status.

If the user's desired room temperature would not match the current room temperature, she may change the value manually with a user-interface. Alternatively, a controller may be set up to do this job for the user and adjusts the value automatically. In both cases, the steganographer's set actuator state would be overwritten and the hidden information would be lost. For this reason, the steganographer must apply a strategy to store hidden data in a robust manner. That is why we apply here scatter hoarding behavior known from nature.

Hiding information in states of actuators can be considered stealthy as it is unlikely that a steganalyst will search a BAS for such information. In this scenario, it is indifferent whether the steganographer is an internal or an external user, i.e. connected to the BAS via an in-house network or via

the Internet. The only requirement for the steganographer is to be able to monitor and alter states of actuators. This way, a limited number of bits can be stored per BAS, which may, in larger installations, be enough to store a cryptographic key.

A steganalyst's goal would be to correctly differentiate between an actuator change that was caused by a steganographer and by another reason, e.g. due to programmed logic or a user's interaction with the BAS.

5.1 Testbed and Storage Strategy

We verify our approach using the BACnet communication protocol. BACnet is integrated for the data exchange between sensors, actuators, direct digital controls (DDCs), monitoring and management devices, and gateways. Each BACnet *device* contains so-called *objects*, these represent the actual data points (e.g. analog input or outputs/binary inputs or outputs). Each object contains so-called *properties*, which define the attributes of the particular object such as the object's identifier, name, type or *present value*. The present value is the value that was currently measured (e.g. a temperature measured by a temperature sensor) or set (e.g. heating power of a heater). These properties can be read by other BACnet devices and – if writable – can be written (set) by other devices. BACnet can be transferred over IP (encapsulated in UDP, called BACnet/IP), MS/TP (on the basis of RS-485), ARCNET and several other protocols.

We will investigate how BACnet can be used to secretly store data using actuators. For this case, we will also make an analogy to the scattered hoarding found in nature.

We implemented a BACnet test-bed using the open source BACnet stack [6]. We have chosen the stack as it already supports several required BACnet features, including BACnet/IP transfer. We created several BACnet devices, each installed on a separate physical machine (each possessing an Intel Core 2 6600 CPU, 2GHz), running the GNU/Linux operating system (kernel 4.10). All machines were connected using a switched 100 Mbit/s Ethernet connection. For some of the experiments, virtual machines running GNU/Linux inside Virtualbox were virtually connected. For all experiment results, we indicate whether physical or virtual machines were used. However, physical machines were especially used to obtain results regarding the robustness of stored data under realistic conditions.

Selection and Storage Strategy: In a first step, the steganographer needs to learn which BACnet devices are available, and which of them can be used

as a steganographic storage. For this reason, the sender transfers a *Who-Is* broadcast message to which the available BACnet devices reply with an *I-am* message as shown in the following listing.

```
$ ./bacwi # broadcasts a BACnet Who-Is message
Received I-Am Request from 1001, MAC = 143.93.191.114.186.192
;Device    MAC (hex)           SNET  SADR (hex)          APDU
;--------- ------------------- ----- ------------------- ----
  1001      8F:5D:BF:72:BA:C0   0     00                  1476
;
; Total Devices: 1
```

The sender caches all addresses of the replying BACnet devices. In a following step, the sender iterates through all these devices in order to determine their BACnet objects, type of each object such as Analog Input, Analog Output, or Binary Input, their present value, and whether they are writable. A list of writable and potentially suitable objects, i.e. devices where slight actuator state modifications are not suspicious, is then cached on the sender. These objects can be found analogous to caches available for the animal for storing its food items.

In order to store data only on such devices which are not frequently changed and which are eventually unused, the cache is built up in a continuous process, including the repetitive scan of devices which were already scanned. This way, the steganographer can determine whether any values changed over a period of days. For instance, if the heating level of a heater is changed during a day or a window is opened and closed, a steganographic modification would not only be easy to detect but would also be overwritten soon. Based on the device types, time of the day, and day of the week, values can change at a different rate. Thus, the goal of the steganographer is to determine these values that were never changed (or were changed seldom) and which are most probably unused. This bears analogy to the hoarding behavior of the animals when they have to carefully select the location of a cache so it is not easily discoverable by any competitor.

The storing (embedding) of secret data into the actuators is performed using the *Write-Property* service of BACnet while the extraction of secret data is done using the *ReadProperty* service. To sort secret messages which were distributed over multiple devices for a later reconstruction, we write to BACnet devices with a lower number and lower object instance number first.

Storage Size: In our test-bed, we performed writing operations to the *present value* property of a BACnet sample object that is delivered with the BACnet

stack. We used several devices simultaneously. In each device, we stored 7 bits per property, using 1 property per device (i.e. 7 steganographic bits per device). BACnet devices typically allow to store between 1 (boolean) and 16 bits per present value property. To store an AES (Advanced Encryption Standard) key of 256-bit length, our test-bed would require 37 properties. These properties are usually distributed over several devices (e.g. one or two present values per devices, resulting in 19 to 37 devices per 256-bit key or 9–18 devices per 128-bit key). Large building installations can contain tens of thousands of properties (actuators, sensors and other BACnet devices combined). If a steganographer could utilize 5 to 10% of the actuators of a medium-size BACnet installation with 1,000 to 20,000 actuator properties, he could theoretically store approx. between 43 Bytes and 1.7 kBytes, given that he embeds 7 bits per property on average. This represents between 2 and 108 128-bit cryptographic keys.

5.2 Evaluation

5.2.1 Performance evaluation

We performed 10,000 writing and 10,000 reading operations to BACnet object properties from a steganographer process, located on a virtual machine, to the `bacserv` sample device of the BACnet stack, located on another virtual machine. Writing a property took on average $0.00556s$ per value, reading a property took on average $0.00563s$ per value. However, high reading and writing speed can only be considered realistic in the virtualized environment as real-world embedded BAS devices provide a slower performance, due to lower computing power and usually slower networking mediums, such as MS/TP. Also, if a time-consuming operation, e.g. opening a window, would need to be performed, the operations would slow-down significantly. Our measurements are thus only realistic for Binary Output devices with high computing power, where outputs are activated in a short period of time.

However, not necessarily all output devices of a building are actually in use, which means that unused outputs do not cause any effect to the physical BAS environment. Figure 2 shows such digital output of the devices which are integrated into a BACnet BAS but where the outputs of these devices remain unused at all times (no cables attached). For instance, such output/input devices could have been planned to be used for attached devices which were later not integrated, or, the attached devices were removed after being not needed anymore. If such devices are found by the steganographer, they represent optimal storage locations.

Figure 2 Unused BACnet digital output devices.

5.2.2 Robustness

The steganographic data in a building can be a subject to constant changes. The logic in a building and the manually performed changes of actuator states strongly influence the content of a hidden message if it is stored in actuator states. For this reason, we implemented two additional experiments:

Firstly, we performed writing and reading operations with an additional *spurious process* in our *virtual* test-bed. Spurious processes are known from database/system security [2] and disturb covert communication processes. In our case, the spurious process performs reading operations over the same link as the steganographer, reading the same BACnet device's object and property.

Our results have shown that all writing and reading operations of the steganographer were still performed successfully, independent of the spurious process and whether the spurious process read only one value/*s* or whether it was running in a burst mode (requesting as many BACnet values per second as possible). However, due to network overload in the burst mode, several UDP packets of the steganographer remained unacknowledged (BACnet provides a simple reliability feature) and had to be sent repeatedly, resulting in a delay of a factor 3.

Overall, this experiment simulated an active steganographer in a building that faces little traffic or burst traffic, i.e. demonstrating that our method can be applied under both conditions.

Secondly, in our *physical* test-bed, we caused the spurious process to write random data to the same BACnet device's object and property as used by the steganographer. The spurious process wrote data every T time-units while the desired storage time S of the secret data was 1 time-unit

(in our experiments 0.1sec). This reflects a typical animal hoarding situation, in which hoarding is usually successful if the storage time of the hoarded food items is less than the time needed by the competing animal to discover the cache and steal the resources within. If the choice of the storage location made by the steganographer was not fortunate, we assume that $T = S$ or even $S > T$. Figures 3 to 5 illustrate the results of our test-bed-based experimental evaluation, which was run under conditions reaching from being highly spurious ($T = S$) to those with few spurious intrusions ($T \ll S$). For all conditions, we counted the reading errors (in %) during 2,000 reads. We repeated each experiment 20 times (resulting in 40,000 measurements per condition) to calculate the mean value (drawn through line) and the standard deviation (dashed line) for each scenario.

The x-axis shows the loop interval in which the spurious process modifies a location (*T*) in which secret data are stored while the y-axis illustrates the steganographer's percentile reading errors of its own data (a reading error could only be caused by the spurious process, which represents an inhabitant or control loop that changes actuator states). We measured the percentage of reading errors depending on the interval of the spurious process' activation as follows:

- *bad selection strategy* (Figure 3), ranging from $S = T$ (spurious process was activated with the same frequency as data was stored) to $S = T/10$ (storage duration was $0.1 \cdot T$),

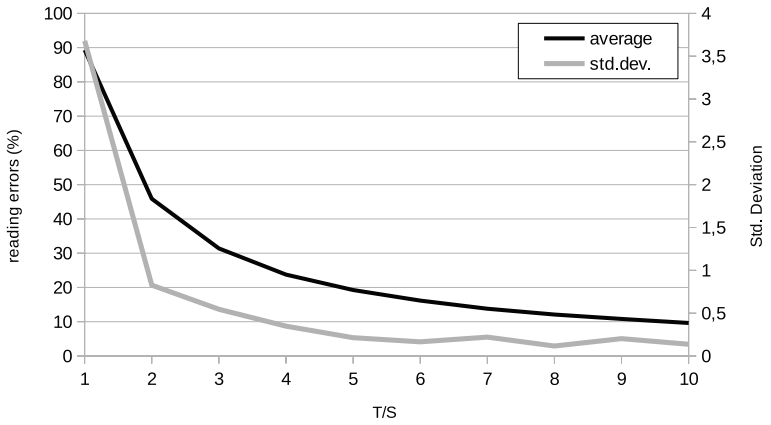

Figure 3 Percentage of spurious data corruption (in average and std. dev.) of steganographic data stored in BACnet properties for a bad selection strategy ($S = T$ to $S = T/10$) in our physical (non-virtualized) test-bed.

- *average selection strategy* (Figure 4) ranging from $S = T/10$ to $S = T/100$.
- *good selection strategy* (Figure 5) ranging from $S = T/100$ to $S = T/1000$.

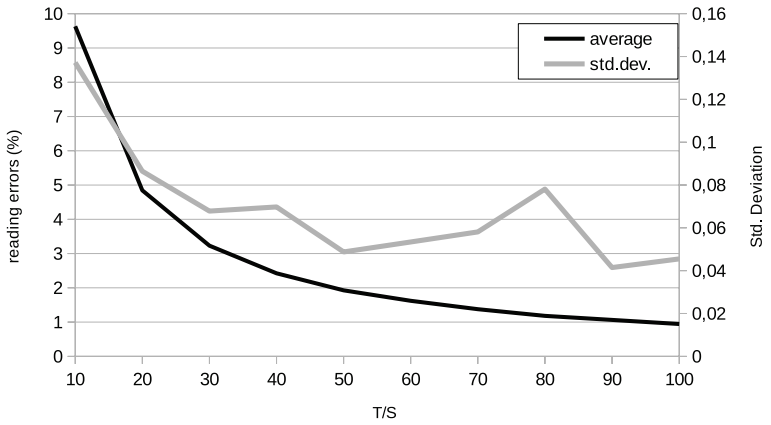

Figure 4 Percentage of spurious data corruption (in average and std. dev.) of steganographic data stored in BACnet properties for an average selection strategy ($S = T/10$ to $S = T/100$) in our physical test-bed.

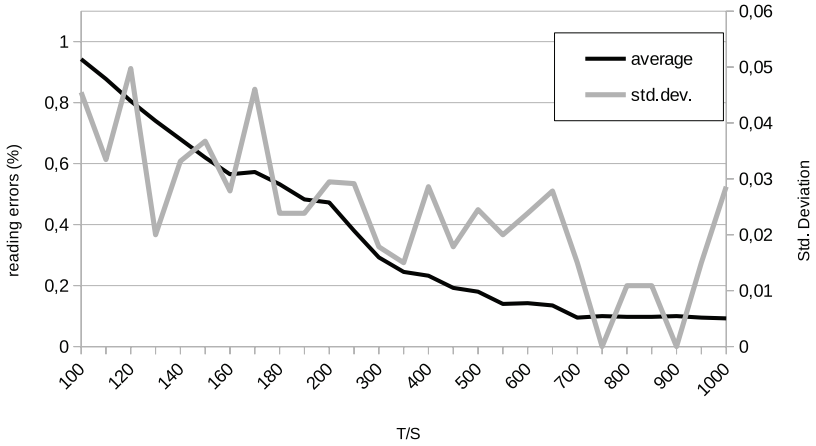

Figure 5 Percentage of spurious data corruption (in average and std. dev.) of steganographic data stored in BACnet properties for a good selection strategy ($S = T/100$ to $S = T/1000$) in our physical test-bed.

As shown, the stored value was lost in 89.32% ($\sigma = 3.68$) of cases if the storage time equals the interval time in which the spurious process overwrites the data ($T = S$), i.e. when the modification is almost guaranteed. This only happens when the device selection strategy was not applied or was unsuccessful. The error-rate drops to an average of 9.6325% ($\sigma = 0,14$) if $S = T/10$ and to 0.94% ($\sigma = 0.045$) for $S = T/100$ (0.093%, $\sigma = 0,029$ for $S = T/1000$).

We performed the same experiments within a virtualized environment. The difference between the results in the virtualized environment (same software) and the physical environment are shown in Figure 6 for the selected key values ($S = T,\ T/10,\ T/100,\ T/1000$). The light grey line shows the difference between virtual and physical results. The physical environment with more realistic conditions showed lower error rates. For $S = T$ the advantage for the physical environment was 10.18%, for the other conditions, the advantage/disadvantage of the physical environment was below 1% (ranging from 0.51% to –0.04%).

Given the fact that a perfect, i.e. error-free, storage strategy will only be available for entirely unused actuators, error-tolerant coding should be applied by the steganographer.

5.2.3 Detectability

The detectability by a steganalyst or user of a building as well as by operators can become trivial if the selection of devices and their object's properties is

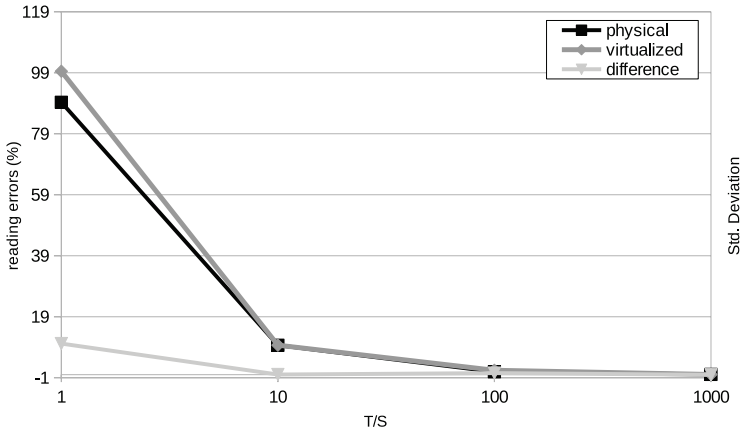

Figure 6 Comparison of the steganographic robustness for the physical and the virtual BACnet test-bed.

not optimal. For instance, if a window is opened (to store a hidden bit) that is usually closed at night, it could trigger an alarm. One detection approach could also be to use an entropy-based analysis of BAS events or machine learning methods [15]. However, such a detection method would require the constant recording of all actuator states over time, i.e. would result in a high number of value polling actions (negligible in most BAS environments) and could influence privacy of inhabitants since potentially sensitive sensor values must be stored and would allow the creation of user profiles.

Due to our described device selection strategy, however, only those devices are used which are unlikely to influence the actual physical environment and which are unlikely monitored, making them a suitable place for steganographic data. As a strategy to remove the most attractive caches, operators could deactive entirely unused actuators from the CPS. However, the number of these devices can be expected to be small in practice – at least in vacant buildings. Despite of deactivating unused devices, no countermeasures for this type of CPS steganography is available in practice.

6 Conclusion

In this paper we demonstrated that hidden messages can be stored in CPS environments in a reliable manner. At least two approaches exist for such a data storage: modification of device registers and modification of actuator states.

Due to the vitality of CPS, actuator state values are updated on a regular basis either by control logic loops or by human-introduced changes; we applied a device selection strategy to utilize devices which are used the least. Certain changes, such as opening or closing windows can be easily detected and should thus not be considered for the hidden data storage. To discover suitable CPS components, we used scatter hoarding as an analogy and strategy known from nature. Therefore, we distributed small parts of a steganographic message over a larger number of CPS devices and selected especially these devices for the embedding which are rarely used or not used at all.

Overall, it must be also noted that the amount of data that can be covertly stored in the CPS is small, i.e. it can contain a message in size of e.g. a cryptographic key within a single smart building, limiting the application scenarios for CPS-based information hiding in practice. On the other hand, CPS are currently not a subject to steganalysis research and can thus be considered stealthy storage objects.

In future work, we plan to store hidden data in the form of subscription relations for the BACnet protocol's *Change-of-Value* (COV) functionality. Moreover, different animal hoarding models will be considered in order to chose the optimal selection strategy which is most favorable for CPS-based information hiding scenario.

Acknowledgements

The authors would like to thank Jörg Keller for his valuable comments on this paper.

References

[1] BACnet.org. Vendor ID list. http://www.bacnet.org/VendorID/, 2016.

[2] Y. A. H. Fadlalla. *Approaches to Resolving Covert Storage Channels in Multilevel Secure Systems*. PhD thesis, Univ. of Brunswick, 1996.

[3] Gerry Howser. Using information flow methods to secure cyber-physical systems. In *Critical Infrastructure Protection IX*, volume 466 of *IFIP AICT*, pages 185–205. Springer, 2015.

[4] ISO. Standard 16484-5:2014: Building automation and control systems (BACS) – part 5: Data communication protocol, 2012.

[5] Simo Jokinen and Jukka Suhonen. Food caching by willow and crested tits: A test of scatterhoarding models. *Ecology*, 76(3):892–898, 1995.

[6] Steve Karg. BACnet Stack. http://bacnet.sourceforge.net/, 2015.

[7] Edward A. Lee. Cyber physical systems: Design challenges. In *2008 11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing (ISORC)*, pages 363–369, 2008.

[8] Maxim Integrated Products, Inc. Data sheet for the DS18B20 digital thermometer, 2015. http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf.

[9] Wojciech Mazurczyk, Steffen Wendzel, Sebastian Zander, et al. *Information hiding in communication networks: fundamentals, mechanisms, applications, and countermeasures*. IEEE Press series on information & communication networks security. Wiley, 2016.

[10] Frederik Möllers and Christoph Sorge. Deducing user presence from inter-message intervals in home automation systems. In Jaap-Henk Hoepman and Stefan Katzenbeisser, editors, *Proc. ICT Systems Security and Privacy Protection: 31st IFIP TC 11 International Conference, SEC 2016*, pages 369–383, Cham, 2016. Springer International Publishing.

[11] Thomas Mundt, Frank Krüger, and Till Wollenberg. Who refuses to wash hands? privacy issues in modern house installation networks. In *Int. Conf. Broadband, Wireless Computing, Communication and Applications*, pages 271–277. IEEE, 2012.

[12] Marissa A. Ramsier, Andrew J. Cunningham, et al. Primate communication in the pure ultrasound. *Biology Letters*, 8(4):508–511, 2012.

[13] Lennart Suselbeek, Vena M.A.P. Adamczyk, Frans Bongers, et al. Scatter hoarding and cache pilferage by superior competitors: an experiment with wild boar, sus scrofa. *Animal Behaviour*, 96:107–115, 2014.

[14] Jernej Tonejc, Sabrina Güttes, Alexandra Kobekova, and Jaspreet Kaur. Machine learning methods for anomaly detection in BACnet networks. *Journal of Universal Computer Science (J.UCS)*, 22(9):1203–1224, 2016.

[15] Jernej Tonejc, Jaspreet Kaur, Adrian Karsten, and Steffen Wendzel. Visualizing BACnet data to facilitate humans in building-security decision-making. In *Proc. Int. Conference Human on Aspects of Information Security, Privacy and Trust (HAS)*, volume 9190 of *LNCS*, pages 693–704. Springer, 2015.

[16] Nilufer Tuptuk and Stephen Hailes. Covert channel attacks in pervasive computing. In *Proc. 2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 236–242. IEEE, 2015.

[17] Steffen Wendzel, Benjamin Kahler, and Thomas Rist. Covert channels and their prevention in building automation protocols: A prototype exemplified using BACnet. In *Proc. IEEE CPSCom Workshop on Security of Systems and Software Resiliency (3SL)*, pages 731–736. IEEE, November 2012.

[18] Steffen Wendzel, Wojciech Mazurczyk, and Georg Haas. Don't you touch my nuts: Information hiding in cyber physical systems. In *Proc. IEEE Security & Privacy Workshops 2017*. IEEE, 2017. in press.

[19] Joanne Hwan Jie Yin, Gan May Fen, Fiza Mughal, and Vahab Iranmanesh. Internet of Things: securing data using image steganography. In *Proc. Third International Conference on Artificial Intelligence, Modelling and Simulation*, pages 310–314. IEEE, 2015.

[20] Elżbieta Zielińska, Wojciech Mazurczyk, and Krzysztof Szczypiorski. Trends in steganography. *Commun. ACM*, 57(3):86–95, March 2014.

Biographies

Steffen Wendzel received his Ph.D. degree in computer science from the University of Hagen in 2013, Germany. Between 2013 and 2016, he led a smart building security research team at Fraunhofer FKIE, Germany. He joined Worms University of Applied Sciences as a professor of information security and computer networks in 2016. Steffen wrote five books and his research focuses on information hiding and security in the Internet of Things. He is a member of the editorial board of the Journal of Universal Computer Science (J.UCS) and of the Journal of Cyber Security and Mobility (JCSM).

Wojciech Mazurczyk received the B.Sc., M.Sc., Ph.D. (Hons.), and D.Sc. (Habilitation) degrees in telecommunications from the Warsaw University of Technology (WUT), Warsaw, Poland, in 2003, 2004, 2009, and 2014, respectively. He is currently an Associate Professor with the Institute of Telecommunications, WUT, where he is the Head of the Bio-Inspired Security Research Group (bsrg.tele.pw.edu.pl). His research interests include bioinspired cybersecurity and networking, information hiding, and network security. He is involved in the technical program committee of many international conferences. He also serves as a reviewer for major international magazines and journals. Since 2013, he has been an Associate Technical Editor of the IEEE Communications Magazine (IEEE Comsoc).

Georg Haas is a bachelor's student of applied computer science and a student assistant at the Centre of Technology Transfer and Telecommunications (ZTT) at the University of Applied Sciences in Worms, Germany.