Decoy Systems with Low Energy Bluetooth Communication

Aaron Hunter and Ken Wong
BC Institute of Technology, Burnaby, Canada

Keywords: Decoy Systems, Intrusion Detection and Response, Low Energy Bluetooth.

Abstract: We propose an architecture for a decoy system that uses low energy Bluetooth devices for communication. We argue that these devices can be effective not only due to low power consumption, but also because an attacker can not detect the signal from a distance. As such, information sent from the decoy system to a monitoring system is unlikely to be noticed by an attacker. We describe a physical system that we have developed for testing and experimentation with this approach. The results so far are promising both in terms of the effectiveness of monitoring, and also with respect to the hidden communication. Moreover, while the decoy system is high-interaction, it does not lead to any system interruption on the main system. Our system is novel in that it is developed from scratch, using low-cost hardware in a manner that accurately captures the way communication would happen in a real system. We discuss the advantages and limitations of our framework, and discuss possible approaches to establishing formal proofs of security for this kind of physical system.

1 INTRODUCTION

Preventing break-ins and limiting the damage done by an attacker is a key problem for security professionals. While it would be preferable to completely block unauthorized access to systems, it is generally accepted that this is not a realistic goal. As such, a variety of decoy systems have been proposed to fool an attacker into entry for the purpose of information gathering. In order for these systems to be effective, they need to do two things. First, they need to closely resemble a real system. Second, they must be able to log and communicate the behaviour of the would-be attacker in a manner that can not be detected. In this paper, we argue that one effective way to design such systems is through the use of Bluetooth Low Energy (BLE) devices.

This is a preliminary paper in which we argue that BLE devices can be an important component of a decoy system. This paper makes several contributions to research on decoy systems. First, we describe the physical simulation that we have developed using BLE devices for system monitoring and show that it functions effectively. This demonstrates not only the value offered by BLE devices for monitoring, but it also provides a template for the development of low-cost simulations of physical systems. The notion of 'undetectability' of a BLE signal leads to a discussion of proofs of knowledge; we suggest that formal methods can be used to provide proofs of security of our system, under the assumption that the BLE signal can not be detected.

2 MOTIVATION

Many different techniques are used in practice to prevent break-ins. Standard preventative measures include the use of firewalls, along with cryptographic methods to protect important data. But it is well known that firewalls are often mismanaged (Khan and Gupta, 2013) and they are vulnerable to port-scanning techniques (Kaur and Gurjot, 2014; Ahanger, 2014). An intruder that gets past a firewall can often do a great deal of damage, and they can even gain access to protected data through vulnerabilities in cryptographic software (Lazar et al., 2013). Due to the limitations of purely preventative measures, the development of *decoy systems* has been established as another form of protection from attackers.

The simplest form of decoy system is a *honeypot*, which is just a fake system that contains data resembling a real system(Padda et al., 2016). The idea is to lure a would-be attacker into the honeypot and keep them busy looking at information while tracking their behaviour. One problem with this approach is that attackers can often detect when a system is fake. As such, it is more effective to develop a real system that

Hunter, A. and Wong, K.
Decoy Systems with Low Energy Bluetooth Communication.
DOI: 10.5220/0006646604040409
In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018)*, pages 404-409
ISBN: 978-989-758-282-0

has all of the expected functionality, but simply does not connect to real data or devices. The case for this approach is presented in (Rowe and Rrushi, 2016) for industrial control systems, but the basic idea applies broadly to any system designed to lure and fool attackers.

The second problem with a simple honeypot is that it must communicate with a monitoring system, but this communication can be detected by an attacker. The simplest solution is to implement a system with minimal communication, a so-called 'low-interaction' honeypot. However, it is well-known that such systems have limited use because they do not track the actions of the attacker sufficiently (Rutherford and White, 2016). In fact, if the system is not monitored adequately, a honeypot can actually increase the risk of theft of data from the real system (Brown and Andel, 2016).

We are left then with the following problem. We need a decoy system that communicates extensively with the real system, but is not detectable to an attacker. One idea is to avoid hard-wired connections with the real system by using wireless (Bluetooth) communication with the decoy (Fawaz et al., 2016). In principle, this should allow high-interaction with less risk of detection. There are two problems. First, Bluetooth is expensive in terms of energy use; this is a problem if the system is running for weeks or months at a time. The second problem is that a Bluetooth signal is reasonably strong, and it can be detected by an attacker with suitable hardware for monitoring wireless communication.

Both of the problems with Bluetooth communcation can, in principle, be solved by using BLE devices. These devices use much less energy than traditional Bluetooth, which allows them to be powered continuously for months or years; this makes it possible to place the devices in locations where Wifi access points would be difficult to power (Kriz et al., 2016). Moreover, it is known that the low-energy signal can not be detected at a distance (Gogic et al., 2016). But there are challenges in designing such a system. The physical limitation is that the decoy system needs to be within about 10 meters of the monitoring system. But even at a short distance, there are questions around the feasibility of communicating enough data quickly for monitoring using BLE devices, because the communication rate for BLEs is much slower than traditional Bluetooth. Moreover, in order to guarantee privacy, we still require a cryptographic handshake (Michalevsky et al., 2016).

Based on the situation outlined in this section, it seems that BLE devices offer a promising solution to developing effective, high-interaction decoy systems. But we need to test this idea in a practical system architecture in order to show that these advantages can actually be obtained in a physical demonstration setting. That is the main motivation for this paper.

3 THE PROPOSED INFRASTRUCTURE

3.1 Hardware

We set up a physical simulation using three Raspberry PI devices acting as servers. We used Raspberry PI version 3, as it was a low-cost option capable of running a variety of operating systems. For this simulation, we used Kali Linux.

- *The Decoy*: One of the devices represents the decoy system. It will contain data, and monitor the activities of those that connect to it.
- *The Attacker*: A second device represents the attacker. This is on the same physical network as the decoy system.
- *The Actual System*: This device is on a different network, and it represents the 'real' system that is being protected.

Each Linux server was configured with an internal firewall and user authentication privileges. The actual system was secured further with closed ports.

An Arduino electronic hardware device is used as an interpreter for communication between the servers; it is the key to align the BLE devices such that they can communicate with each other on each server. This interpreter was added to facilate data transfer, due to difficulties encountered with BLE stand-alone devices.

The interpreter device acts like the master controller that can control each of the Raspberry PI electronic boards. Each Raspberry PI electronic board acts like a peripheral or slave device. As such, the Arduino device activates and deactivates the BLEs when communication is required. The network design is shown in Figure 1. As noted in the previous section, since we are using BLEs, we need the devices to be relatively close; in testing we required less than 10 meters between devices.

3.2 System Details

We wrote several applications to detect the presence of an attack, and report activity to the monitoring system through the BLE devices. All monitoring applications are written in Python. The basic activities of the main monitoring application are the following:

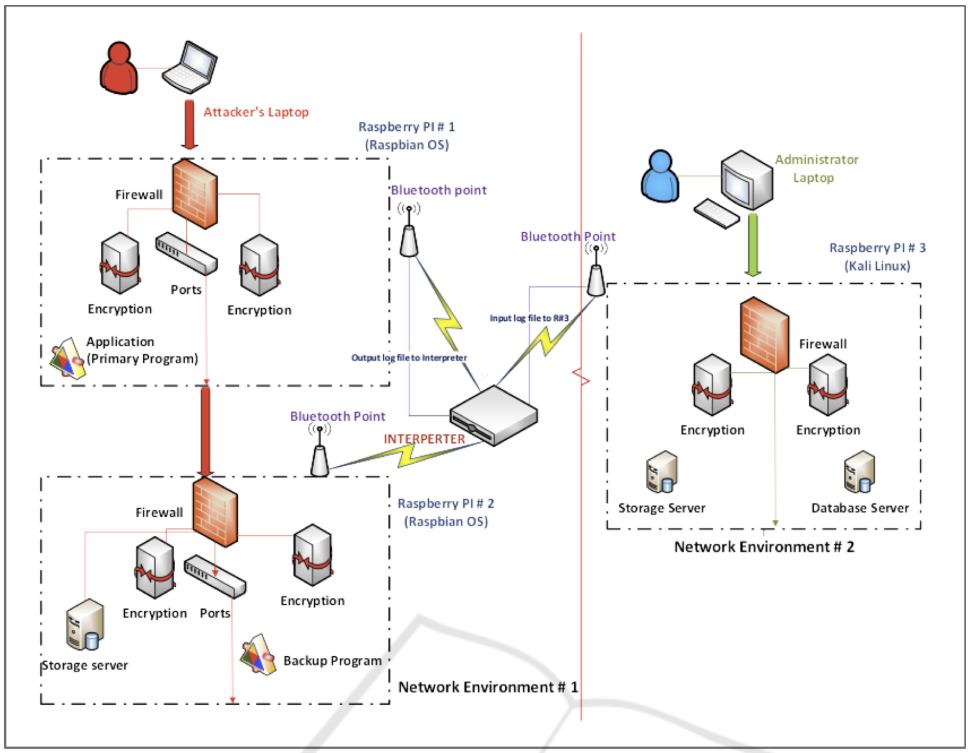

Figure 1: Network Configuration.

1. *Login detection*: Every login to the decoy machine is detected, and considered an attack.
2. *Information collection*: The user ID, passwords attempted, and session ID are collected.
3. *Information transmission*: The log file is sent to a program that communicates with the BLE devices.

All of these tasks are duplicated in the system, through two copies of the main monitoring program. This is a fail safe mechanism, in case the attacker identifies and disables the program running on the decoy system.

In addition to the main monitoring application, several other programs handle related tasks. These include scanning for BLE transmissions, establishing a communication channel, and receiving transmitted log files. The BLEs communicate with each other through these applications, relying on the interpreter to activate teh BLE devices and permit the transmission of data. When the Interpreter is disconnected, all communication between the servers stops. In this way, it secures the communication that is flowing through from the decoy system to the main system.

This methodology allows us to limit the attackers opportunity to compromise a system. It also allows us to investigate and gather precise information about the attackers position. There are limits to the methodology with respect to protecting the system. The method of installing two application programs can be interrupted and removed by the attackers. However, it would be difficult to remove both of the application programs at once, especially as they are installed on two separate servers and the backup application program will not be active until the primary application program fails. As noted previously, the BLEs provide a better solution than Classical Bluetooth devices because the range makes it harder for an attacker to detect the communication.

Testing of the system is performed as in Figure 2, with remote login on all devices simultaneously. This allows us to simulate attacker activity, and view the results of monitoring in real time.

4 RESULTS

4.1 Testing

There are several kinds of result that can be discussed from the development of this system. One entirely practical result is the fact that we were actually able to produce a working demonstration that uses BLEs for communication. This is worth emphasizing as a

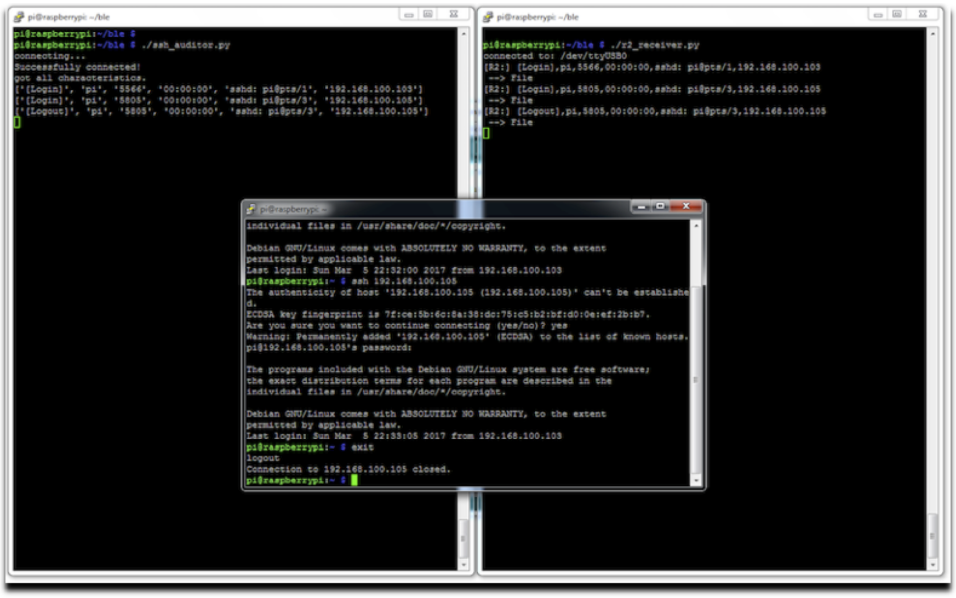

Figure 2: Remote Login.

result, because the physical configuration of the system was actually a technical challenge. In order to get it to work, we had to experiment with a variety of computing devices, a variety of BLE solutions, and a variety of physical configurations for the network. The result was a working system that includes the key features required for testing.

Once the system was setup, all three servers ran and communicated effectively. We were able to simulate attacks on the decoy system in which the attacker's activity was successfully logged, and the actual monitoring system successfully received the log file containing the attacker's IP address, session ID and password. Since the monitoring system is only connected to the decoy through the BLE interpreter, we can verify that the BLE communication was effective when the decoy system was within 10 meters of the monitoring system. However, when we moved the decoy system further than 10 meters from the BLE transmitter, then the log files were no longer transmitted successfully. We did not do detailed testing to find the exact distance required, as this is not significant for our purposes. But our testing did indicate that successful monitoring was dependent on proximity; when the devices were close enough, then the data was tranmitted successfully on every attempt.

4.2 Implications

We successfully built a system that will detect an attack, trace the attack and give an alert if there is an attack by using BLE devices to data transfer between servers. The results here show that the developed application can monitor and gather information about the attacker. However, one feature of the system that we were not truly able to verify is the undetectability of the BLE communication between the decoy and the main server. As noted, we were able to verify the limited range of communication; but this is not exactly the same as verifying undetectability. We know that the signal is weak, and it can not be directly detected physically at a distance. However, we need to consider other possibilities:

- The attacker may have access to a close physical sensor.
- The attacker may have some non-physical means to detect the communication.

The first point here is straightforward to address. In practical settings, we would physically prevent an attacker from having access within 10 meters of our system. The second point is harder. We can not state conclusively that the communication is undetectable, only that it is physically undetectable from a distance. We admit the possibility that an attacker could 'figure out' that communication is occuring, or that an attacker could have inside information.

We remark that the methodology here uses very low-cost hardware to simulate a physical system. The Raspberry PI devices running Linux offered a realistic simulation of real servers, and the BLE devices communicating through an Arduino board allowed us to use low-energy Bluetooth in the same way it would be used on a real system. Hence, while the actual data involved is very simple, our system accurately simulates the BLE-aspect of system monitoring that we

set out to study. Therefore, we suggest that the basic approach here can be seen as a model for low-cost prototyping of physical systems, particularly in a research or education environment.

5 DISCUSSION

5.1 Theoretical Verification

To this point, we have essentially established two things. First, we have shown that it is possible to set up a physical network that tracks an attacker's behaviour while reporting this behaviour through a BLE device. Second, we have reason to believe that the BLE communication will not be detected by an attacker, because the signal is so weak that it can not be detected outside very close physical proximity. However, we really have not established in a precise sense exactly what this means. To be more precise, we have not provided any *proof* of correctness or any *guarantee* that the system will work.

What we are actually interested in creating is a system where we have knowledge that the attacker does not. In particular, we would like the attacker to be ignorant with respect to several things:

1. The attacker should not know that the system they are viewing is fake.
2. The attacker should not know that we know what they are doing.
3. The attacker should also be unaware of the fact that their actions are being sent to a real system.

Of these points, the first two are standard features of any honeypot-like scenario. The key point that distinguishes our work here is the third one. We are interested in having some sort of guarantee that the attacker will not know what is happening. We have a physical *reason* to believe communication will not be detected, but we would like to work towards *proofs* of security for this kind of system. In this section, we briefly provide some theoretical grounds for the analysis and verification of the network architecture that we have introduced.

5.2 Logical Modelling

There is a long history of using logics to model knowledge and belief. The basic premise of much of this work is that the state of the world can be captured by an assignment of values to propositional variables. As such, we start with a set F of propositional variables, and we define a *state* to be any assignment of true/false values to these variables. A state represents one possible configuration of the world. We can construct *formulas* from F by using the normal symbols $\wedge$ (and), $\vee$ (or), and $\neg$ (not). We do not go further into the logic here, and refer the reader to a standard book such as (Enderton, 2001) for more details.

We also assume a set A of *agents*. The knowledge of each agent is defined through something called a *Kripke structure*, which associates with each agent a relation r_A on the set of states. Informally, if $r_A(s,t)$, then the agent A can not tell the difference between the states s and t. If the actual state of the world is s, we can therefore define the *beliefs* of A to be the set of states t such that $r_a(s,t)$. Typically, we have restrictions on r_A; for knowledge it is normally assumed to be an equivalence relation. We refer the reader to (Fagin et al., 2003) for a complete introduction to reasoning about knowledge in a multi-agent setting. There are many variations of this approach in the literature.

We are not interested in producing a detailed logical formulation of knowledge in this paper; it is sufficient to note that such a formalization is possible. Hence, the following can all be formally stated, where A stands for an attacker and S stands for a security administrator:

- $\neg Knows_A(fake)$
- $\neg Knows_A(Knows_S(action))$
- $\neg Knows_A(sent(sys(action)))$

Logics of knowledge in the tradition of BAN logic (Burrows et al., 1990) can be used to formalize the meaning of these statements. There are existing tools for using such formalizations to formally prove the security of systems (Cremers, 2008; Hunter et al., 2013).

The question is then: What use are logics of knowledge for the system currently under discussion? The fact is that our practical demonstration does not allow us to conclude anything about the knowledge of the attacker. We know that the attacker performs certain actions, and we communicate these actions through a channel that we assert is hidden. We have strong reason to believe that this communication is undetectable, based on the physical properties of the signal. But we really can not be precise about exactly what this allows us to conclude in terms of guarantees of system security.

By formalizing the communication in a logic of action, we can be precise about the initial knowledge of the attacker and the system administrator. We can also be precise about the actions the system performs, and whether or not they are visible to the attacker. This allows us to track the knowledge of the attacker accurately. At the same time, it allows us to go back

and reconsider, if it turns out that the attacker had some reason to know about the log files transfer. A clever attacker could actually deceive the system by performing actions that they want logged.

In any event, this formalization would allow us to state formal properties about the system, and then formally prove them under flexible assumptions about the attackers behaviour. This formal analysis will take our system from a practical demonstration to a provably secure architecture. We leave this formal aspect for future work.

6 CONCLUSION

The paper describes an approach to setting up a decoy system where the actions of an attacker are logged and transmitted to a monitoring system using BLE devices. The results showed that this could be done effectively, provided that the decoy system is in close physical proximity to the monitoring system. Significantly, the signal used to communicate could not be detected at a distannce beyond 10 meters. This is an advantage for physical systems, as it means an attacker is less likely to know that they are being monitored. While we acknowledge that an attacker might be able to use non-physical means to determine they are being monitored, the physical undetectability of our signal does make it less likely that our decoy system will be discovered. In the discussion, we briefly discussed how notions of undetectability and ignorance may be formalized to provide proofs of security at a theoretical level.

Based on the results of the present research, future work will enlist stronger filters such as adding additional authentication keys between each of the BLE devices that can strengthen security in the system. For instance, setting up an authentication key combination code that is required to be validated before entering the BLE master controller , which is between the Raspberry PI Linux servers. Not only will this strengthen the security of the system, but it will also prevent the attacker from being able to control the Arduino electronic board, which is the master Bluetooth key controller. Once the attacker compromise the Arduino electronic board, they can have full accessibility to control the Bluetooth Low Energy devices and that will allow the attacker to control the main system too.

REFERENCES

Ahanger, T. A. (2014). Port scan a security concern. *International Journal of Engineering and Innovation Technology*, 3(10):241–246.

Brown, A. and Andel, T. (2016). What's in your honeypot? In *Proceedings of the 11th International Conference on Cyber Warfare and Security*, pages 370–377.

Burrows, M., Abadi, M., and Needham, R. (1990). A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36.

Cremers, C. (2008). The Scyther Tool: Verification, falsification, and analysis of security protocols. In *Computer Aided Verification, 20th International Conference*, pages 1–30.

Enderton, H. (2001). *A mathematical introduction to logic*. Harcourt/Academic Press, 2nd ed edition.

Fagin, R., Halpern, J., Moses, Y., and Vardi, M. (2003). *Reasoning About Knowledge*. MIT Press.

Fawaz, K., Kim, K., and Shin, K. (2016). Protecting privacy of BLE device users. In *Proceedings of the 25th USENIX Security Symposium*, pages 1205–1221.

Gogic, A., Mujcic, A., Ibric, S., and Suljanovic, N. (2016). Performance analysis of bluetooth low energy mesh routing algorithms in case of distaster prediction. *World Academy of Science, Engineering and Technology*, 10(6):1–7.

Hunter, A., Delgrande, J., and McBride, R. (2013). Protocol verification in a theory of action. In *Proceedings of the Canadian Conference on AI*, pages 52–63.

Kaur, R. and Gurjot, S. (2014). Analysing port scanning tools and security techniques. *International Journal of Electrical Electronics and Computer Science Engineering*, 1(5):58–64.

Khan, S. and Gupta, R. (2013). Future aspects of firewall in internet security. *IEC International Journal of Technology and Management*, 1(1):30–36.

Kriz, P., Maly, F., and Kozel, T. (2016). Improving indoor localization using bluetooth low energy beacons. *Mobile Information Systems*, 62:1–11.

Lazar, D., Chen, H., Wang, X., and Zeldovich, N. (2013). Why does cryptographic software fail? A case study and open problems. In *Proceedings of the 5th Asia-Pacific Workshop on Systems*, pages 1–7.

Michalevsky, Y., Nath, S., and Liu, J. (2016). Mashable: Mobile application of secret handshakes over bluetooth. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking*, pages 1–14.

Padda, S., Gupta, S., Apoorva, G., Lofty, S., and Kaur, A. (2016). Honeypot: A security tool in intrusion detection. *International Journal of Advanced Engineering, Management and Science*, 2(5):311–316.

Rowe, N. and Rrushi, J. (2016). *Introduction to Cyber Deception*. Springer, 1st ed edition.

Rutherford, J. and White, G. (2016). Using as improved cybersecurity kill chain to develop an improved honey community. *IEEE Computer Society*, pages 2624–2632.