Ransomware in Resource-Constrained Industrial IoT Networks: There Actually is a Threat

Yuxiang Huang*, Calvin Brierley‡, Adel ElZemity‡, James Pope†, Jiteng Ma*, Antonio Di Buono§, Budi Arief‡, George Oikonomou*
*School of Electrical, Electronic and Mechanical Engineering, University of Bristol, UK
†School of Engineering Mathematics and Technology, University of Bristol, UK
‡School of Computing, University of Kent, Canterbury, UK
§National Nuclear Laboratory, Central Laboratory, Sellafield Seascale, UK

***Abstract*—The threat of ransomware attacks against Industrial Internet of Things (IIoT) networks, particularly networks of resource-constrained devices, is starting to become a reality. In this paper, we contend that the threat of ransomware infection of an IIoT environment is not only plausible, but that it also exhibits different properties compared to ransomware attacks against traditional desktop systems, necessitating a new and more appropriate approach to deal with this threat. In particular, we articulate the unique characteristics of ransomware behaviour in IIoT networks considering the distinctive characteristics, such as computationally-constrained devices and low-power wireless communication protocols. Furthermore, we outline the necessary attributes for ransomware to effectively compromise and propagate within IIoT networks. To back our argument, we present a proof-of-concept (PoC) IIoT ransomware prototype. To highlight the generality of our work, we have developed the prototype for two different hardware platforms, powered by two different open source embedded operating systems: Contiki-NG and Zephyr. The results underscore the feasibility of ransomware attacks against networks of resource-constrained IIoT devices, providing evidence of the real threat posed by such attacks in these environments. Finally, our PoC prototype can serve as a foundation for future research focused on securing these potentially vulnerable networks.**

***Index Terms*—Industrial Internet of Things, security, ransomware, prototype, test bed, constrained device.**

I. Introduction

The Industrial Internet of Things (IIoT) has emerged as a pivotal technological innovation in industrial environments by enabling real-time monitoring and seamless data exchange across interconnected sensor networks. This connectivity facilitates continuous data collection, analysis, and decision-making processes, enhancing the efficiency and automation of critical operations. However, such extensive connections also introduce significant security vulnerabilities. The computation- and communication-constrained nature of IIoT networks exacerbates security risks, as the devices forming those networks typically lack the capacity to support traditional countermeasures, creating a security gap that attackers can exploit [1].

One of the ways that attackers can use to compromise interconnected devices is through deploying ransomware. Ransomware is a type of malicious software (malware) that blackmails its victims, typically by preventing victims from accessing their resources (for example through encrypting their data), unless a ransom demand is paid [2]. In the past few years, ransomware attacks have grown significantly and can cause widespread disruption, for instance as demonstrated by the Colonial Pipeline ransomware attack in 2021 [3]. One of the worrying trends is that these days, ransomware operators tend to target big companies and industrial sectors (rather than individual victims) through "big game hunting" facilitated by *Ransomware-as-a-Service (RaaS)* [4].

When it comes to ransomware in IIoT systems, the landscape is still largely unexplored. In the context of IIoT systems, ransomware can exploit communication channels, sensor systems, and machine-to-machine (M2M) interfaces to spread and disrupt operations, making IIoT networks – especially those deployed in critical infrastructures such as manufacturing, energy, and transportation – particularly attractive targets for ransomware. Such an attack could prove to be very disruptive, costly, or outright dangerous.

Therefore, we believe that the threat of ransomware against resource-constrained IIoT networks is real and cannot be dismissed. Our work makes the following contributions:

- We highlight the unique characteristics and properties of ransomware against IIoT networks. Unlike traditional ransomware that affects desktop systems, this work emphasises how the low-power, resource-constrained nature of IIoT devices and their wireless communication protocols can make them particularly vulnerable.
- We analyse how these devices (with their limited processing power and wireless communication protocols) can be effectively targeted. In doing so, we provide evidence that ransomware in IIoT environments is not just theoretical but a real and plausible threat.
- We present a proof-of-concept (PoC) ransomware prototype that has been implemented for two different contemporary hardware/software platform combinations. This prototype shows the feasibility of ransomware infection and propagation using low-power wireless communication technologies prevalent in IIoT networks, highlighting the urgency for further research in developing countermeasures.

The rest of the paper is organised as follows. Section II presents related work in ransomware and its context in IoT, microcontroller and IIoT. Section III outlines our assumptions

TABLE I
RANSOMWARE CHARACTERISTICS ACROSS DIFFERENT PLATFORMS

Category	Characteristics	Platform	Attack Technique	Typical Detection Approach
Desktop Oriented	Encrypts critical data. Threatens data leaks to coerce ransom payment	Desktop OS (Windows, Linux, etc.)	Cloud / Network-based	Signature-based detection, behavioural analysis, network traffic monitoring
IIoT Endpoint Device	Locks device functions. Encrypts stored data	Consumer IoT (smart TVs, NAS)	Device-specific	Lightweight opcode sequence analysis, cloud-assisted detection
IIoT Gateway	Encrypts critical gateway data. Disrupts network operations	Microprocessors (Raspberry Pi)	Resource-intensive	Traffic analysis, signature-based detection
IIoT Node	Tampering sensor data. Disrupting device behavior	Microcontrollers (Cortex-M)	Resource-constrained	Relies on gateway/cloud analysis

and threat model. Section IV explains the set-up of our PoC ransomware prototype, followed by the two PoC scenarios in Sections V and VI. Section VII concludes our paper and suggests areas for future work.

II. RELATED WORK

Ransomware has evolved beyond traditional desktop environments to target diverse Industrial Internet of Things (IIoT) platforms. Each category, including IIoT endpoint devices, gateways, and resource-constrained nodes, exhibits distinct characteristics and attack techniques, as summarised in Table I.

A. Ransomware on Endpoint

Ransomware is typically defined as a form of malware that, upon infecting a target device, will restrict its victim's access to the device's services and/or data (for instance, by encrypting the stored data), forcing the victim to pay the ransom demanded by the attacker in order to restore access [5]. On top of this attack on availability, newer strains of ransomware tend to follow a "double extortion" approach, in which the attacker would compromise the confidentiality of the victim's data by threatening to divulge the victim's sensitive, embarrassing or compromising information, unless they paid the ransom demand [2]. While there have been numerous examples of ransomware impacting traditional desktop operating systems [4] and businesses [6], ransom attacks against IoT devices are less common. Those that have been performed in the wild typically targeted consumer devices, such as smart TVs [7], and NAS drives [8].

Previous research has attempted to create PoC ransomware for IoT devices, including attacks on their availability [9], as well as confidentiality [10]. These studies have also identified several challenges that attackers must address due to inherent limitations in IoT devices. Such challenges include the relatively low value or easily replaceable nature of IoT hardware, difficulties in achieving persistence, and the absence of standardized communication channels between attacker and victim. As a result, bespoke ransomware had to be developed specifically for IoT targets. Some solutions were found to be partially generalisable by making use of the features offered by operating systems implemented by the target device [11]. However, for devices with stricter hardware limitations, these issues may prove harder to solve, especially if the target does not implement an operating system that the attacker could leverage for abstraction.

B. Ransomware on Gateway

One of the primary challenges in securing IIoT devices is their limited computational and energy capacities [12], which restrict the use of conventional security techniques such as anti-malware tools and encryption. Microprocessors, like Raspberry Pi, serving as IIoT gateways, offer greater computational power and can implement more comprehensive security solutions, such as malware detection and encryption. However, they still face limitations when compared to traditional computing systems [13].

Researchers have focused on leveraging real-time traffic classification methods, as well as Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) specifically designed for IIoT edge gateways to enhance security without overburdening the device [14]. Despite these efforts, ensuring comprehensive security across the IIoT ecosystem remains a challenge, particularly in maintaining the balance between security and resource consumption in constrained environments.

C. Ransomware on Microcontroller

Ransomware detection on microcontrollers, particularly in the context of IIoT environments, has gained increasing attention due to the limited computational resources and security vulnerabilities of these devices. Recent studies have explored various approaches to tackle this challenge. A prominent method involves analysing opcode sequences to identify malicious patterns within the firmware [15], to differentiate between benign firmware and ransomware. Similarly, opcode sequence analysis has been applied to the detection of cryptographic algorithms within ransomware, utilising a dynamic recurrent neural network (DRNN) for malware detection [16]. However, this approach faces challenges when transitioning to microcontrollers due to its computational cost.

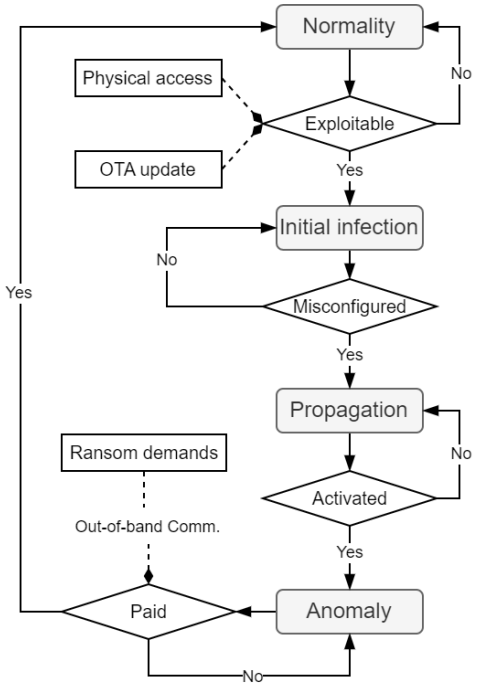

Fig. 1. Threat model lifecycle against the device

While studies have explored heuristic-based approaches to identify abnormal behaviours such as unusual memory access or processing spikes in ransomware attacks, these methods often suffer from high false-positive rates in resource-constrained environments. The cloud-based anti-malware system CloudEyes [17] implements a lightweight client-side scanning agent and a novel server-side signature detection mechanism, ensuring data privacy and low-cost communication. However, it relies on cooperation from the cloud server, and its detection rate and accuracy could be further improved to optimise storage and matching efficiency.

III. Assumptions and Threat Model

In this work, we assume that constrained devices are organised in a star or mesh network topology, and that they communicate using a low-power wireless communication protocol such as IEEE 802.15.4, Bluetooth Low Energy (BLE), or LoRa. We assume that this network is connected with the rest of the infrastructure through a more powerful gateway.

A. Initial Infection

We assume that initial infection does not take place over the network by traversing the gateway, as it would be feasible for the defender to deploy traditional countermeasures (e.g., IDS/IPS). Instead, we assume that the malicious actor has gained physical access to one of the constrained devices that form the network, as can be seen at the top of Fig. 1, which depicts the lifecycle of our threat model. Through this access, the actor can exploit unpatched or outdated firmware, making the device susceptible to specific vulnerabilities. The assumption that the malicious actor has physical access is a reasonable assumption, for example, this could be a disgruntled employee, or an individual who gained access to the network as a contractor. Having gained physical access to this device, the attacker is able to infect it by reprogramming it.

Attackers can also inject malicious code during the Over-the-Air (OTA) update process. As firewalls are often configured to only protect the network boundary, rather than local wireless connections, attackers might exploit vulnerabilities in nearby wireless protocols, such as the BLE stack, to inject the malicious code.

B. Infection Propagation

After infecting the initial device, we assume that the ransomware will try to propagate wirelessly across the network, by copying itself to other devices. We are assuming that devices are capable of receiving firmware updates over the network. This is considered a good practice (for example, to ensure that all devices can be patched easily and regularly), but also opens up a possible attack vector. Incorrect configurations of networked embedded systems is very often a cause of security vulnerabilities that expose the device to unauthorised access or exploitation [18]. Such misconfigurations could, for example, result in a failure to authenticate the source of a firmware update, which can be exploited by the malicious actor in order to spread the infection.

It is worth pointing out that the ransomware will likely remain dormant during this phase – i.e. not interfering with the normal operations of the infected devices – in order to minimise the risk of being detected too early. The goal is to infect a significant portion of the network before activation, so that it can cause the maximum damage, and to make it harder for the victim to contain the spread within their network.

C. Anomaly Activation

As stated earlier, to maximise disruption, we assume that the ransomware will delay its manifestation until it has infected a significant number of devices, then triggering simultaneous activations across all of them. The activation criteria are predetermined by the malicious actor and can involve various methods. For instance, activation may be initiated via a command over the network, or the ransomware may self-activate once the number of infections reaches a certain threshold, thereby compromising a large percentage of the networked devices.

Unlike traditional ransomware, which typically encrypts files to deny access, we assume that these constrained devices do not store sensitive or highly valuable files locally. Instead, the attacker's goal is to use the compromised devices as a platform for other malicious activities, such as interfering with sensing or actuation within the infrastructure.

D. Ransom Demands

Given that the constrained devices in question may not have a graphical interface to interact with the end-user, we assume that the attacker will demand ransom through out-of-band communication methods, such as via email or phone, or using

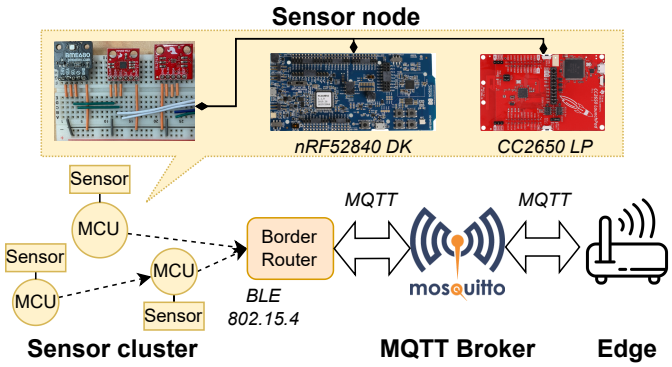

Fig. 2. Sensing network architecture

some communication hijacking techniques such as those outlined in [9], [10]. To convince the victim of the ransomware's presence, the attacker is capable of remotely demonstrating the attack, for instance, by temporarily disrupting one or more devices as a proof of the infection. Lastly, if the victim has paid the ransom, the attacker is capable of remotely deactivating the ransomware and disinfecting the targeted devices.

E. Desirable Traits for Effective Ransomware

In addition to the baseline assumptions and threat model outlined above, we also identify several "desirable" traits that, if implemented, would likely increase the likelihood of success and the impact of the attack:

i *Viable ransom methods*: The attack will attempt to lock devices to prevent their usage, or force them produce malicious output. If the device can be easily recovered by victims, they would be unlikely to pay.

ii *Worm-like behaviour*: After infecting a device, the ransomware should automatically attempt to spread to other devices that it encounters, maximising its impact, and reducing the need for manual intervention from the attacker.

iii *Generalisable*: Ideally, the ransomware should be designed with more than one device in mind, as this would drastically reduce the cost when attempting to implement it on any additional targets.

iv *Adaptable*: Different targets may present unique challenges that cannot be generalised, such as new peripherals or methods of communication. The attacker should be able to adapt the malware in these scenarios to maximise its impact on each target.

IV. Ransomware PoC Prototype

A. Hardware and Software Platforms

To provide evidence that the threat of ransomware against the networks in question is not theoretical, we present a PoC prototype implementation. To highlight the generality of the argument presented in this paper, we have implemented our prototype for two different hardware/network/software combinations:

- *Texas Instruments CC2650 LaunchPad*. This device is powered by a Cortex-M3 MCU and is equipped with an IEEE 802.15.4 radio transceiver. The PoC runs on the Contiki-NG open source operating system (https://www.contiki-ng.org/) [19]. Devices are organised in a 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) multi-hop mesh.
- *Nordic Semiconductor nRF52840 Development Kit*. This device is powered by a Cortex-M4 MCU and equipped with a dual IEEE 802.15.4/BLE radio. For our PoC implementation on this platform, we have used BLE. The PoC runs on the Zephyr RTOS (https://www.zephyrproject.org/).

B. Network Architecture

Our PoC deployments consist of constrained devices that are either equipped with an IEEE 802.15.4 radio, or with BLE. The architecture of our PoC is diagrammatically presented in Fig. 2.

In the case of IEEE 802.15.4, devices are organised in a 6LoWPAN multi-hop mesh. Each individual resource-constrained device performs sensing and/or actuation, while at the same time acting as a router capable of forwarding data from a different device on the path towards the destination node. At the edge of the network, a border router acts as a gateway between the constrained network and the remainder of the infrastructure. In a typical setup, the edge device is mains-powered without resource limitations.

BLE devices are also present and they communicate with the gateway directly within a single wireless network hop.

V. PoC Scenario 1: CC2650/6LoWPAN/Contiki-NG

This scenario has been implemented using Contiki-NG for the TI CC2650 Launchpad hardware platform. As outlined earlier in Section III, initial infection might take place as a result of a malicious actor gaining physical access to a device. They can install malicious code by reprogramming the device over a physical interface, such as a Joint Test Action Group (JTAG) or Serial Peripheral Interface (SPI).

A. Malicious Function Injection via RAM

The ransomware prototype takes advantage of the limited protection mechanisms in IIoT devices to execute a malicious payload by exploiting memory management vulnerabilities.

The attack begins by copying the Contiki-NG process scheduler to a different address. A malicious function replaces the original process scheduler, effectively hijacking control of the device. The method operates by copying the malicious function into a designated buffer in the RAM, which is then transmitted to another device within the network, as depicted in Fig. 3.

On the target device, a function pointer is created locally, and the buffer address containing the malicious function is cast to this pointer. By calling the function pointer during the program execution, the malicious function is invoked seamlessly, allowing its payload to execute without detection.

Algorithm 1 shows a method to dynamically inject and execute malicious code by copying the machine instructions

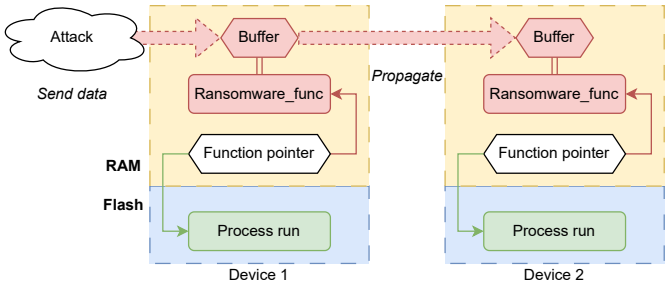

Fig. 3. Malicious function copy in RAM

Algorithm 1 Copy Malware Code to Buffer

procedure CopyMalwareCodeToBuffer
  typedef int $(*func\_ptr)$(void) ▷ Declare a function pointer
  $pRAM \leftarrow$ address of func_buffer
  $pCode \leftarrow$ address of mal_process_run
    AND NOT 1 ▷ Align to Thumb mode
  for $i \leftarrow 0$ to length of func_buffer $- 1$ **do**
    $pRAM[i] \leftarrow pCode[i]$
  end for
  $malware\_func \leftarrow$ cast func_buffer to *func_ptr* type ▷ Cast the buffer address to the function pointer type
end procedure

Algorithm 2 Malware Execute Process

procedure MalwareExecuteProcess
  typedef int $(*func\_ptr)$(void) ▷ Define a function pointer type
  Cast *process_ptr* to *func_ptr* type
  $process\_ptr \leftarrow$ normal_process ▷ Point to normal process
  if $is\_compromised$ **then**
    $process\_ptr \leftarrow$ malicious_process ▷ Redirect to malicious process
  end if
  $process\_ptr()$
end procedure

of a predefined function (`mal_process_run`) into a buffer located in RAM. A function pointer is then used to invoke the copied code during runtime. Initially, the machine code is referenced using a pointer (`pCode`), which points to the source address of the malicious function, ensuring alignment through bitwise manipulation. The machine instructions are then sequentially copied into a buffer (`func_buffer`) in RAM via a loop. Once the copying process is complete, a function pointer (`func_ptr`) is cast to point to the buffer address, enabling the execution of the malicious function without the need for recompiling the firmware or rebooting the system. The process function pointer (`process_ptr`) is redirected to the malicious function during runtime execution, bypassing the normal process once the attack is initiated, as shown in Algorithm 2.

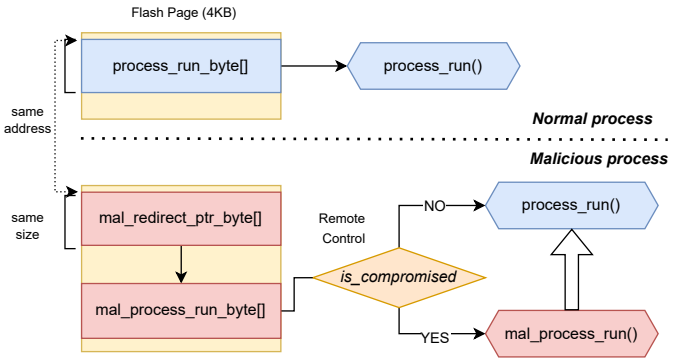

Fig. 4. Malicious function copy in flash

B. Malicious Function Embedding in Flash

To gain permanent control, the ransomware overwrites the device's flash memory with the same malicious function and replaces the normal process function at the same address. This malicious rewrite of the flash ensures that the ransomware persists through reboots and continues to control the system even after the device is restarted. The success of this attack is often facilitated by the lack of comprehensive runtime security features, such as memory protection or robust privilege enforcement, in many IIoT devices.

The ransomware targets the normal process function in the program while keeping other systemic functions intact. Once the malicious buffer is received in RAM, it is written to flash memory at the same address as the legitimate process function. This allows the ransomware to be invoked using the same function name as the normal process, effectively hijacking the system.

To avoid an overflow caused by differing function sizes, a jump table is used to redirect the execution to the malicious function, which can be stored in an area of the flash memory with sufficient space. To ensure that it is the same size as the original function, we insert no-operation instructions (NOPs) to pad the empty space before the next function. The normal process is replaced with a malicious function pointer that redirects the program to the malicious function, which is allocated at the last section of the flash page. Additionally, a global variable is reserved to create a backdoor for subsequent remote activation, facilitating the remote switching between normal and ransomware program execution, as seen in Fig. 4.

C. Stealthy Ransomware Propagation

When the resource-constrained device is in the sensor data reporting network like destination oriented directed acyclic graph (DODAG), the proposed ransomware prototype can achieve spontaneous machine-to-machine propagation, thereby infecting neighbouring nodes. Once the ransomware is activated, the infected nodes will immediately become compromised, and carry out abnormal activities defined by the attacker to achieve ransom. The three stages of the ransomware infection and propagation process are illustrated in Fig. 5.

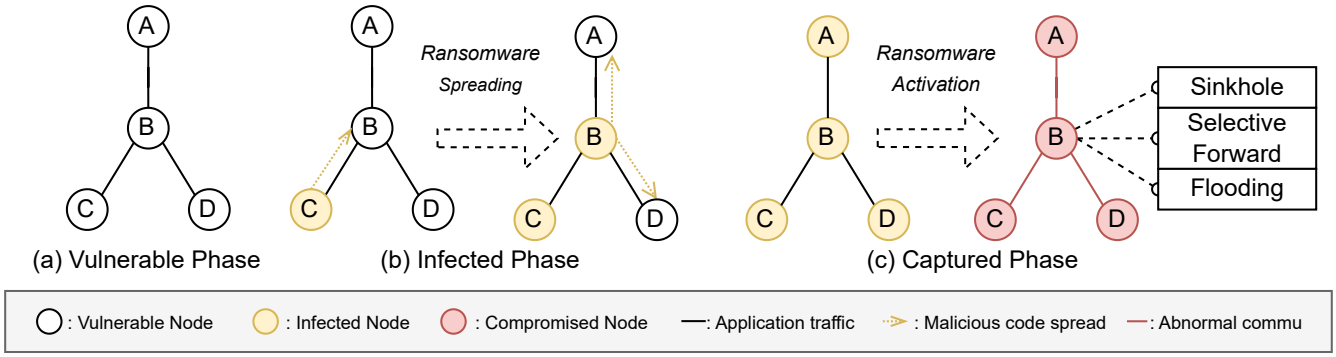

Fig. 5. Ransomware infection and spreading on the target device

In the initial state, nodes pass normal application data to each other, but due to the lack of protection measures, they are vulnerable to attacks. This phase is called the Vulnerable phase. The proposed ransomware model can achieve the spontaneous spread of malicious code to adjacent nodes to cause infection, but in this phase, defined as the infected phase, even if the infected nodes report sensor data normally, they will not show abnormalities as before, making this spreading behavior difficult to detect.

When the number of infected nodes in a network reaches a threshold, the ransomware attack mode can be activated. In this captured stage, the infected device will unconsciously force the attacker to execute the attack mode pre-loaded, such as classic sinkhole, selective forward and flooding attacks. Even at this stage, the ransomware attack becomes explicitly captured, and restarting the infected node cannot restore it to normal. Although reflashing the firmware can remove the ransomware, applying such solution across a large number of IIoT devices incurs significant time and resource costs, which can be catastrophic in critical infrastructures.

D. Implementation and Results

In the experiment, the target device is configured to receive data from a downstream end device during its vulnerable phase. It then forwards the received data together with its own sensor data to the upstream device. After completing the sending task, the device enters the normal idle state, and switches to the sleep mode to conserve energy until the next data exchange cycle. During the experiment. Charge consumption (in milliCoulombs) and number of packet transmissions are recorded to evaluate the activity of the target device in the IoT network.

When the proposed prototype model infects the target device, the malicious code, being small in size and broadcast at a very low frequency, spreads to neighboring nodes. After confirming infection, the device stops broadcasting to avoid noticeable changes in the network traffic and reduce the chance to be detected.

When the number of infections in the network reaches a defined threshold, the ransomware will be activated. In this experiment, the attacker is assumed to use three typical attack methods. As shown in the Fig. 6, under different attack

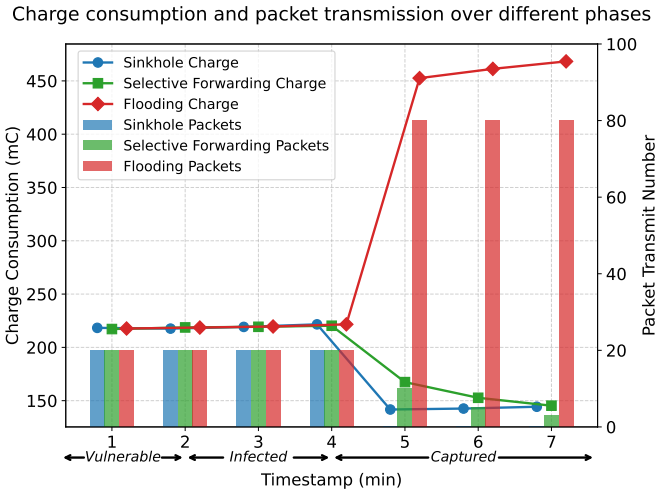

Fig. 6. Charging consumption and packet transmission on device

scenarios, the target device exhibits significant changes in both charge consumption and the number of packet transmissions. These observations confirm that the proposed prototype effectively achieves the intended attack behavior and fulfills the objective of ransomware deployment.

The experimental results demonstrate that the proposed self-propagating prototype successfully enables machine-to-machine transmission to neighboring nodes without triggering alerts from standard monitoring mechanisms. A key characteristic of the prototype is its ability to remain dormant in a quiescent state on infected devices. As a result, infections through the proposed ransomware prototype are largely undetectable by conventional monitoring tools. Once activated, infected devices begin to display abnormal behavior when it is already too late to implement effective countermeasures.

VI. PoC Scenario 2: nRF52840DK/BLE/Zephyr

We have investigated the applicability of a similar approach for other hardware/software targets, and their possible susceptibility to ransomware using alternative communication methods. During this exploration, we created firmware for an nRF52840 based on the Zephyr operating system, which allowed users to communicate with the device via BLE and modify various characteristics.

On characteristic allows users to upload data to the device 20 bytes at a time, while the other allows them to change the device's name. However, the code handling modifications of the "name" characteristic is intentionally vulnerable by design, and can be exploited remotely by an attacker. By using crafted bluetooth packets, we are able to overflow the "name" variable, redirect execution to previously uploaded shellcode, and gain control of the device. Using these characteristics, we simulated an attack. First we generated ARM shellcode on an attacking machine, then uploaded it to the device's RAM. We then exploited the buffer overflow vulnerability in the "name change" function to overwrite the program counter with the address of our uploaded shellcode.

Typically, it would not be possible to run the shellcode stored in RAM, as it is stored an area marked as data, which cannot be executed. However, the nRF52840 has another memory partition, named "Code RAM", which *is* executable, and is mapped to the same physical RAM. By instead redirecting the program counter to reference the stored shellcode via the Code RAM mapping, we were able to take control of the device, and remotely execute custom code.

Using this exploit, we were able to run various shellcode that called functions provided by the original firmware, searched the memory space for a predetermined value, ran an infinite loop, or forced the device to crash. This demonstrates that gaining control of IoT devices via other communication mediums is possible, and could lead to similar forms of ransomware, such as those shown in this work, being implemented on other devices. By forcing infected BLE devices to switch between the central and peripheral roles, we aim to explore ransomware spread across the network, propagating malicious code autonomously through the network via nearby adjacent devices.

Based on the functional and effective prototype described above, the proposed attack targets both RAM and flash memory and exhibits distinct characteristics compared to traditional methods of exploiting microcontrollers. As summarised in Table. II, such attack does not require rebooting active programs or re-flashing the original firmware to deploy the ransomware. Moreover, its wireless self-propagation relies on malicious code composed of only a few bytes, allowing it to evade detection during the propagation phase.

TABLE II
COMPARISON OF INTRUSION METHODS IN IOT SYSTEMS

Exploitation Method	Intrusion Mode	No Restart	No Re-flash	Spread Pkt. Size	Self-Propagate
Firmware Update (OTA)	Remote	×	×	Large	✓
Debug Port (JTAG/SWD)	Physical	×	×	N/A	×
Side-Channel Injection	Physical	✓	✓	N/A	×
Memory Vulnerabilities	Remote	×	✓	Medium	✓
Our Work	Remote	✓	✓	Small	✓

VII. Conclusions and Future Work

This paper has demonstrated the feasibility and distinctive characteristics of ransomware attacks against IIoT networks of severely resource-constrained devices. Through a PoC ransomware prototype, we produce evidence that such attacks are not merely theoretical, but present a real and significant threat. The PoC prototype successfully compromises two hardware platforms (Texas Instruments CC2650 LaunchPad and Nordic Semiconductor nRF52840) running on Contiki-NG and Zephyr embedded operating systems, respectively.

Specifically, in the case of IEEE 802.15.4 networks, we have also shown how ransomware can spread over low-power wireless communication channels. The findings of our study underscore the urgent need for the development of countermeasures, and lay essential groundwork for future research, providing a stepping stone towards more secure and resilient IIoT infrastructures.

Contemporary embedded operating systems do very little in terms of collecting auditing information (such as network interface access by processes, file access, process CPU/RAM utilisation, interrupts, and pre-emptions). As part of our current ongoing work, we are developing novel lightweight system auditing and logging capabilities that can be turned into (and adopted as) novel prevention and detection countermeasures. Development of recovery and immunisation techniques is also within our immediate plans.

We are also currently in the process of generating a dataset collected from a simulated, large-scale IEEE 802.15.4 and 6LoWPAN formed of Contiki-NG devices. The dataset will comprise: i) network packet captures; ii) device auditing logs. This dataset will include periods of normal network operation, as well as periods during which the network is infected and ransomware is spreading. The respective normal/infected periods will be annotated. We believe such a dataset can serve as a valuable resource for understanding, designing and implementing countermeasures against IIoT ransomware. The dataset will be made openly available, in order to support future collaborative research efforts in this area.

Acknowledgment

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC), Grant Numbers EP/X036871/1 and EP/X036707/1.

References

[1] A. Alrawais, A. Alhothaily, C. Hu, and X. Cheng, "Fog computing for the Internet of Things: Security and privacy issues," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 34–42, Mar. 2017.

[2] G. Hull, H. John, and B. Arief, "Ransomware deployment methods and analysis: views from a predictive model and human responses," *Crime Science*, vol. 8, no. 1, pp. 1–22, 2019.

[3] V. Stracqualursi, G. Sands, and A. Saenz, "Cyberattack forces major US fuel pipeline to shut down," 2021, https://edition.cnn.com/2021/05/08/politics/colonial-pipeline-cybersecurity-attack/index.html.

[4] S. Razaulla, C. Fachkha, C. Markarian *et al.*, "The age of ransomware: A survey on the evolution, taxonomy, and research directions," *IEEE Access*, vol. 11, pp. 40 698–40 723, Apr. 2023.

[5] P. O'Kane, S. Sezer, and D. Carlin, "Evolution of ransomware," *IET Networks*, vol. 7, no. 5, pp. 321–327, Sept. 2018.

[6] K. Makortoff, "World's biggest meat producer JBS pays $11m cybercrime ransom," 2021, https://www.theguardian.com/business/2021/jun/10/worlds-biggest-meat-producer-jbs-pays-11m-cybercrime-ransom.

[7] C. Cimpanu, "Android ransomware infects LG smart TV," 2016, https://www.bleepingcomputer.com/news/security/android-ransomware-infects-lg-smart-tv/.

[8] J. Bajera and M. Glet, "Ransomware attack on the QNAP device: A case study," in *IBIMA Conference on Artificial intelligence and Machine Learning*. Springer, 2023, pp. 402–406.

[9] C. Brierley, J. Pont, B. Arief, D. J. Barnes, and J. Hernandez-Castro, "Paperw8: An IoT bricking ransomware proof of concept," in *Procs. 15th Int'l Conf. on Avail., Rel. and Sec. (ARES)*, 2020, pp. 1–10.

[10] C. Brierley, B. Arief, D. Barnes, and J. Hernandez-Castro, "Industrialising blackmail: Privacy invasion based iot ransomware," in *Nordic Conference on Secure IT Systems*. Springer, 2021, pp. 72–92.

[11] C. Brierley, J. Pont, B. Arief, D. J. Barnes, and J. Hernandez-Castro, "Persistence in linux-based iot malware," in *Secure IT Systems: Procs. 25th Nordic Conference (NordSec 2020)*. Springer, 2021, pp. 3–19.

[12] A. Tewari and B. Gupta, "Security, privacy and trust of different layers in Internet-of-Things (IoTs) framework," *Future Generation Computer Systems*, vol. 108, pp. 909–920, 2020.

[13] A. Shalaginov and M. A. Azad, "Securing resource-constrained IoT nodes: Towards intelligent microcontroller-based attack detection in distributed smart applications," *Future Internet*, vol. 13, no. 11, p. 272, 2021. [Online]. Available: https://www.mdpi.com/1999-5903/13/11/272

[14] X. Kong, Y. Zhou, Y. Xiao, X. Ye, H. Qi, and X. Liu, "idetector: A novel real-time intrusion detection solution for IoT networks," *IEEE Internet Things J.*, vol. 11, no. 19, pp. 31 153–31 166, June 2024.

[15] H. Kim, J. Park, H. Kwon, K. Jang, and H. Seo, "Convolutional neural network-based cryptography ransomware detection for low-end embedded processor," *Mathematics*, vol. 9, no. 7, p. 705, Mar. 2021. [Online]. Available: https://www.mdpi.com/2227-7390/9/7/705

[16] J. Smith and J. Doe, "Malware-detection method with a convolutional recurrent neural network using opcode sequences," in *Procs. Int'l Conf. on Embedded Systems and IoT Security*. IEEE, 2020, pp. 120–130.

[17] H. Sun, X. Wang, R. Buyya, and J. Su, "Cloudeyes: Cloud-based malware detection with reversible sketch for resource-constrained Internet of Things (IoT) devices," *Software: Practice and Experience*, vol. 47, no. 3, pp. 421–441, Jun. 2017.

[18] D. Papp, Z. Ma, and L. Buttyan, "Embedded systems security: Threats, vulnerabilities, and attack taxonomy," in *13th Annual Conference on PST*, Sep 2015, pp. 145–152.

[19] G. Oikonomou, S. Duquennoy, A. Elsts, J. Eriksson, Y. Tanaka, and N. Tsiftes, "The Contiki-NG Open Source Operating System for Next Generation IoT Devices," *SoftwareX*, vol. 18, p. 101089, 2022.