Article

Enabling Secure Guest Access for Command-and-Control of Internet of Things Devices

Andrew John Poulter *　and Simon J. Cox

Faculty of Engineering & Physical Sciences, University of Southampton, Southampton SO16 7QF, UK; s.j.cox@soton.ac.uk
* Correspondence: a.j.poulter@soton.ac.uk

Abstract: Internet of Things (IoT) devices are becoming ubiquitous, and may be arranged to form formal or *ad hoc* Command and Control (C2) networks. Such networks typically do not have a mechanism to facilitate the sharing of either data or control inputs. This paper examines this problem in the context of IoT devices operating within C2 systems which do not have a trusted relationship with each other. We propose a solution which we call syndication, to provide a controlled mechanism to share data between C2 systems of devices without a fully trusted relationship. This paper builds upon previous work which established a lightweight protocol for secure C2 operations within the IoT. Using the proposed approach enables not only sharing of data but also permits the external controller to submit moderated requests for actions to be performed. The paper concludes by examining how this approach could also be adopted to provide secure guest access to connected systems in a domestic or commercial context.

Keywords: C2; data sharing; guest access; Internet of Things; MQTT; smart city

1. Background

The growth in the IoT over the last five years has been substantial and has lead to a growing awareness of the security and privacy concerns that apply to the IoT [1].

Our previous work introduced [2] and described [3] the Secure Remote Update Protocol (SRUP), which established a mechanism for secure C2 oriented messaging for the IoT, built on top of the widely used Message Queuing Telemetry Transport (MQTT) protocol [4]. IoT devices can exploit the publish/subscribe nature of MQTT, with devices and control systems running lightweight MQTT clients [5], connecting to a secured broker over Transport Layer Security (TLS) [6].

SRUP provides secure authentication of messages by requiring that each message is signed by its sender, thus ensuring that it both originates from a valid source and that it has not been changed or corrupted (maliciously or otherwise) in transit. The protocol also mitigates a number of other attack vectors, such as replay attacks and message interception, and uses an identity confirmation technique to ensure that physical devices correctly correspond to logical identities within the C2 system [7].

Most previous work in the area of sharing IoT data focuses on the use of cloud-based sharing of data as an output of the C2 structures [8], or controlling access at the MQTT level [9,10], rather than the sharing of data within a C2 framework which we propose here.

This paper introduces a new concept to SRUP which we call *syndication*; a technique which provides a mechanism to permit the moderated sharing of data and federated C2 operations between systems which do not have a trusted relationship with each other.

Although the requirement to connect IoT devices together in an C2 is not new [11], much previous work relating to integrating systems has focused on integrating disparate sources into a unified C2 network [12]. The approach of centralization of control in a C2 is in contrast to swarm robotics [13] where the intended approach is to create a decentralized and self-organizing system of control. Although such an approach is highly beneficial in

some scenarios, the inherent centralization of control within a construct such as a smart-city means that a C2-oriented approach is required.

Our approach is also in contrast to other work, which has explored mechanisms for privacy-preserving data sharing, such as identity-based encryption [14], and social-relationships based discovery of IoT services [15].

The approach described in this paper is in contrast to these approaches—since it examines a different part of the problem-space. Specifically we explore the idea of sharing data at at C2 level, making links between active C2 systems, and presenting an interface for devices on one system to be presented alongside systems on another C2 network.

The remainder of this paper examines the problem in the context of the background and implementation of this approach. In Section 2, we examine the motivation for sharing C2 data beyond the trust boundary of a given system, and in Section 3 we describe a realistic scenario in which such an approach might be required. Section 4 explains how the syndication approach works to address these requirements, and Section 5 examines in detail the messages required to facilitate the syndication process. In Section 6, we discuss how the syndication approach could be used to address the scenario described in Section 3, and Section 7 describes the experimental implementation of this approach within the Secure Remote Update Protocol. Section 8 examines a related problem (enabling guest access to C2 systems), and describes how the syndication approach described previously could be used to provide a solution to this requirement. Finally Section 9 draws some conclusions and makes suggestions for follow-on work.

2. Motivation for This Work

Although the concept of C2 for the IoT is a powerful one, it may require additional elements to align with real-world use-cases. For example, in a scenario where there is a deployed IoT system consisting of a number of fixed sensor devices, the devices could belong to a controlling C2 system. Some examples of different C2 systems that may be operating within the same environment are shown in Figure 1. In the event of an emergency scenario, the devices could be augmented by combining them with a second set of mobile devices; previous work on SRUP has described how these devices may be given commands requesting that they transfer their registration to another C2 server operating using the same backend system [3].
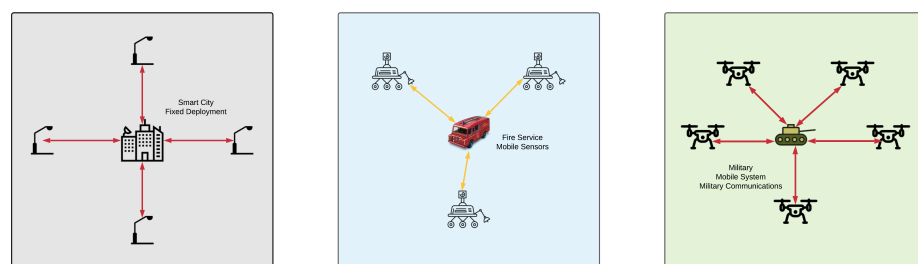

Figure 1. An example of three IoT-based sensor systems, which may be required to operate together in an emergency scenario.

For the purposes of this paper, we consider a hypothetical network of fixed sensors, installed to measure air quality and to provide real-time alerting to citizens if the air quality deteriorates below a certain threshold. This sensor network is owned and operated by a metropolitan region, such as a city, and incorporated into part of a wider smart city concept [16].

In the event of a major fire or other disaster, it may be desirable for this system to be augmented with a number of additional mobile sensors, such as may be deployed by the city's fire service. An example of this type of combined C2 is depicted in Figure 2.
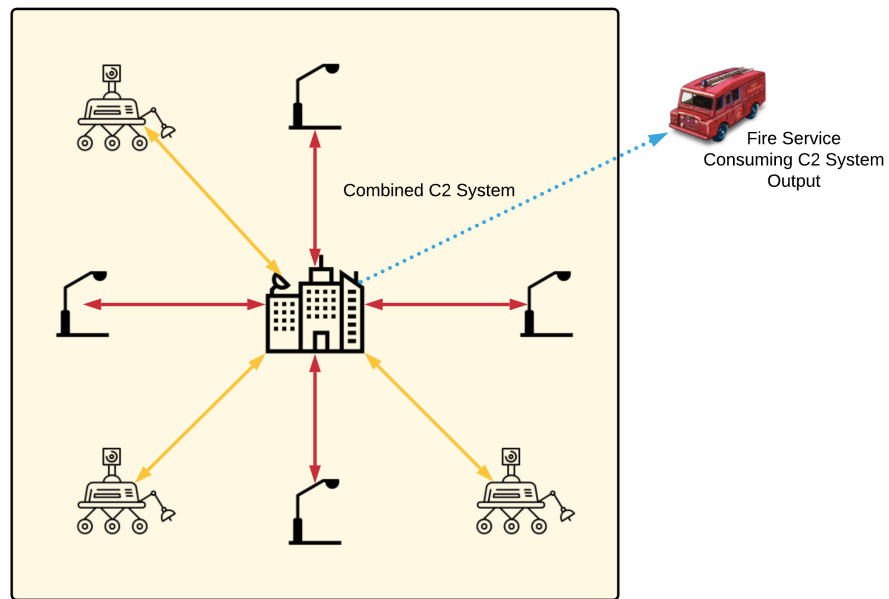
Figure 2. An example of a mobile system joining its sensors to a fixed system during an emergency scenario.

This augmentation of the system can be accomplished by joining the fire service's mobile sensors to the city's fixed sensor network, and adding the fire service to the city's C2 system as a user. For this to be possible, we assert that it requires the following three conditions to be true:

1. The systems must use (or be compatible with and be able to use) the same backend system. In the context of SRUP this means they must use the same key service and MQTT broker;
2. The owner of the sensors which are to be joined to the existing C2 system accepts (temporarily) losing ownership of the sensors to the operator of the fixed system;
3. The owner of the fixed system accepts the new sensors, and takes temporary responsibility for them.

Conditions 2 and 3 are important; since the devices in question will become full members of the new C2 network, both parties must be willing for the systems to be joined, implying a degree of trust. For example, the side hosting the C2 network needs to trust that the devices joining are not compromised with malware, and the side providing the devices needs to trust that the other party will be diligent to ensure that they do not become so.

Depending on the requirements of the city, it may also be possible to simply provide them with access to the fire service's C2 system for their sensor devices or otherwise provide a simple data export. However, in order to best exploit existing onward connections from the city's C2 system (such as real-time alerting), it may be desirable to have live access to the sensor network and to incorporate the fire service's mobile sensors into the city's wider C2 picture.

There are scenarios, however, where such a join is not possible to achieve because one or more of the preconditions are not, and cannot, be satisfied.

3. Conceptual Scenario

To explore this further, let us imagine that the emergency scenario described in Section 2 is sufficiently serious that, in addition to resources from civilian fire and rescue services, other specialist resources are brought in to provide assistance to the regular civil authorities. For example, a situation such as may occur in the event of a natural disaster. This additional

assistance may be provided by military units, non-governmental organizations (NGOs), or even specialist commercial providers.

In this hypothetical scenario, we shall assume that neither condition 1 nor condition 2 hold true; the sensors in question use their own backend system (and potentially a different cryptographic standard too, especially in the case of a military system) and the provider of the sensors wishes to retain full control over their devices. Additionally, in this scenario the provider is not willing to share all of the data with the city, but they do wish to provide a live feed of some of their data to the city's C2 system. Reluctance to share a totality of the data may be due to commercial, security, or other reasons. Similarly other data may not be relevant to the situation for which the systems are cooperating, and so may not be shared to avoid overloading or confusing operators with unnecessary information.

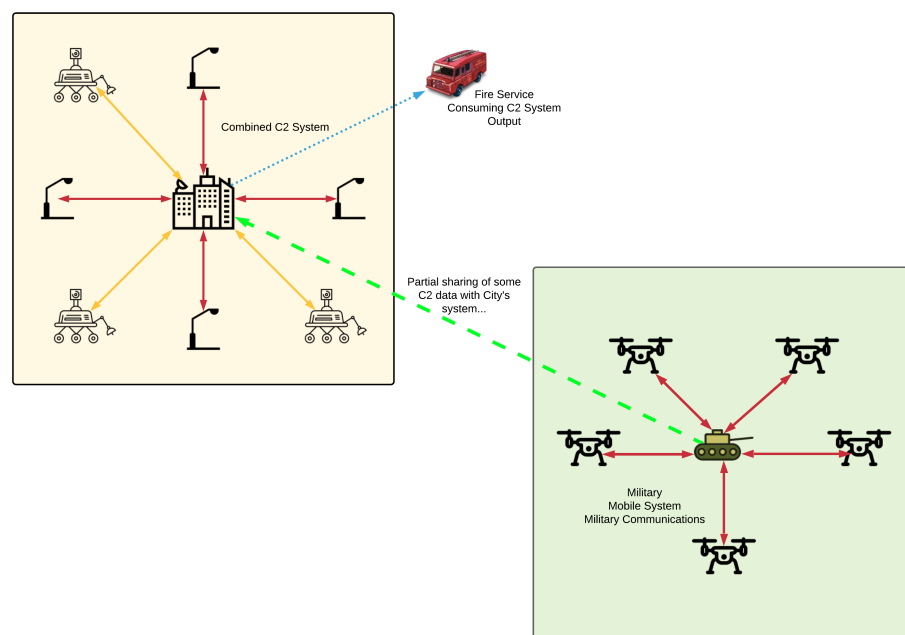This conceptual scenario is illustrated in Figure 3.

Figure 3. A depiction of the idealized data sharing relationship between a smart city C2 system and the sensors provided as a part of military assistance to the civil authorities.

It should be noted that in some scenarios there may be a requirement for bidirectional, rather than unidirectional, transfer of data; for example, sharing of the city's data with the provider's system. Depending on the specifics of how the systems are being used, there may be additional concerns about the trustworthiness of the data. For simplicity, in this paper, we consider this as a unidirectional data flow only.

4. Syndication

To address the requirements described in the previous Section, we have devised a concept which we call *Syndication* (a term coined by analogy to newspapers, where syndicated content may be reprinted in other publications where a pre-existing relationship exists to the original publisher), which provides a mechanism for two C2 systems—which do not share a common backend system—to collaboratively share data and commands between them.

Syndication not only permits sharing of data without ownership of devices changing, it also enables the C2 server corresponding to the source of the information to moderate the information flow, thereby controlling which data types (and data from which platforms) are shared.

Although syndication could also be used to link C2 networks that share a common backend, it is primarily intended to link networks which operate in different SRUP uni-

verses. In this context, a universe refers to the combination of the SRUP backend (the key exchange service and secured MQTT broker) and the underlying cryptographic system being used for that specific instance of SRUP traffic.

By the addition of a specialized syndication device to act as a bridge between the two universes, it is possible to facilitate communications between C2 systems that do not have a cryptographic algorithm in common.

Although the implementation of SRUP described in previous work [3] used 2048-bit RSA (Rivest-Shamir-Adleman) [17] keys for the message signing and encryption, this is a matter of implementation, and the choice of cryptographic algorithm is independent of the underlying protocol. Noting, of course, that both parties involved in secure communication need to be using the same cryptosystem.

5. Messages

In this section, we explore the process of syndication, and examine the various SRUP message types that are required by the syndication process.

5.1. Setup

In each of the following sections, the setup is assumed to be as follows:

- We assume that there are two backend universes, which we call Universe A and Universe B.
- There is a SRUP C2 server which controls one or more devices and which is willing to share data with another C2 server. We call this the syndicated server and assert that it is a member of Universe A.
- There is a second C2 server, which is a part of Universe B. This server is the syndicating server; the server which wishes to receive data from the syndicated server.
- There is also a syndication device. This is a native member of Universe B and is initially subordinate to the syndicating server. In this context the term native member here refers to a device that has been designed to operate wholly within a given universe. Here it resides within Universe B, and makes a connection to Universe A only when it initializes the syndication. During syndication operations, it forms a bridge between the two universes.

A sequence diagram illustrating each of the data flows can be seen in Figure 4.

5.2. Syndication Initialization

To begin syndication operations, it is first necessary to send a syndication initialization message from the syndicating C2 server to the syndication device in order to initialize the join. This is sent using the Universe B backend and cryptographic protocols, via the syndication device's MQTT topic. This message contains the Universal Resource Locator (URL) for the Universe A key service registration, and the pre-shared secret to be used to authenticate the syndication.

On receipt of this, the syndication device should attempt to connect to the provided URL and perform the initial registration process with a new device identity to be used within Universe A. It should perform the subsequent MQTT operations using a second MQTT client connection (and using the necessary keys and certificates received as a part of this join).

5.3. Syndication Request

Having successfully joined (likely requiring a human or machine moderated join), the syndication device sends a syndication request.

The syndication request message is sent by the syndication device to the server to be syndicated. It is sent using the syndication device's Universe A identity for the MQTT topic, and includes the pre-shared secret that the syndication device received as a part of the syndication initialization message.
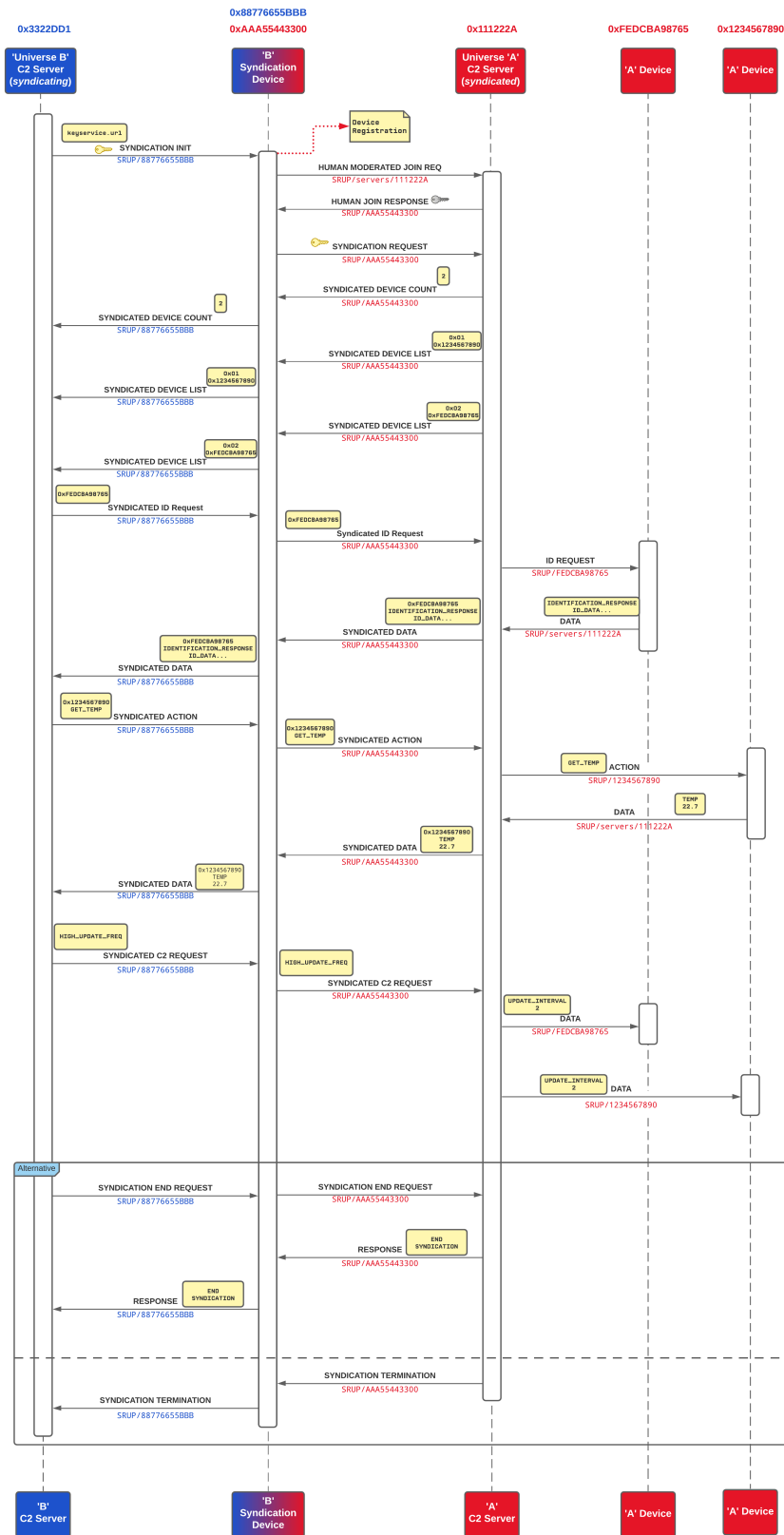
Figure 4. A sequence diagram showing an example data exchange between the devices described in Section 5.1, during SRUP syndication operations. The text beneath the message-type line denotes the MQTT topic used by SRUP to carry the message, as specified within the SRUP protocol.

5.4. Syndicated Device Count and Syndicated Device List

The process for determining the validity of the syndication request is application-specific, and is not enforced by the protocol. However, having received a valid syndication request message, and having determined that the request is acceptable, a C2 server which is to become syndicated must send a syndicated device count message to the syndicating C2 server, stating the number of subordinate devices that it intends to syndicate. This is initially sent to the syndication device (using its Universe A identity), and then relayed on to the syndicating server, using the syndication device's Universe B identity and topic. This approach of message relaying by the syndication device—using its two identities—is adopted similarly for all other syndication message types.

After a suitable message propagation delay, the syndicated server begins sending syndicated device list messages. These consist of two fields: the identity of one of the devices to be syndicated; and an integer index value to indicate which of the devices in the list of n devices that identity pertains to (where n is defined as the value sent in the syndicated device count message). In all cases, the identity that is sent is the Universe A identity of the end device, since devices that are syndicated do not have any identity within Universe B.

5.5. Syndicated ID Request

After the receipt of the syndicated device list messages, the syndicating server may optionally send a syndicated ID request message to the syndicated server, asking that the ID request string corresponding to the device in question is sent to it. The ID data may either be sent directly by the syndicated server, via a syndicated data message containing the ID request data it has previously received from the device, or may be passed on to the end device (in the form of a regular ID request message) to enable the device to update this information.

5.6. Syndicated Data

Syndicated data messages are used by a syndicated C2 server to send data on behalf of one of its subordinate devices. These may be, as described in Section 5.5, as a part of an ID Request, or as a part of sending operational data from a subordinate device. In either case, the message consists of three elements, over and above the standard SRUP message:

- The source ID—containing the ID of the device from which the data originates;
- The data ID—used identically to the non-syndicated SRUP data message to indicate the meaning of the data;
- The data value itself.

Syndicated data messages always originate from a syndicated C2 server, and never from an end device, meaning that the syndicated C2 server always has the ability to moderate what is sent. In practice, this means that a syndicated C2 server can easily be configured to share only certain types of data (e.g., temperature and humidity, but not barometric pressure), to share data only from certain devices, or to downgrade the precision of data before it is sent; for example, a military operator may decide to share a reduced-precision form of geolocation data with civilian authorities.

5.7. Syndicated Action

In addition to receiving data from a syndicated C2 server, a syndicating server may make requests for actions to be performed by the syndicated server's devices. As with the data messages described in Section 5.6, there is no direct connection between the syndicating server and the syndicated devices, so all requests are moderated by the syndicated C2 server. In practice this means that the syndicated C2 can be configured to permit or deny different action types, for different devices, or simply accept any requests and pass them on to the corresponding target device.

The syndicated action message consists of the target device's identity, as well as the 8-bit integer value corresponding to the action type that is being requested.

5.8. Syndicated C2 Request

The syndicated C2 request message allows for the syndicating C2 server to send a request (and any associated data) to the syndicated C2 server. This message consists of an 8-bit integer C2 request ID (corresponding to one of up to 256 defined C2 request types), along with a byte-stream containing any data that may be required to support this request type.

For example, a C2 request could take the form of a request to change a parameter value (such as the update frequency of the device's sensors), which would not necessarily require any supporting data. Conversely, this may take the form of a more complex request, such as requesting that the syndicated server applies a software update to its devices, the details for which are supplied within the data field.

In common with all of the other syndication messages, the syndicated server must determine whether or not to to accept the request.

5.9. Syndication Termination and Syndication End

Either party within the syndication can end the syndication session at any time. There are two ways that this can be achieved:

- The syndicating server may send a syndication end request message. This is used to inform the syndicated server that it should stop sending syndication messages, and to instruct the syndication device to stop accepting messages from the syndicated server once it has received a response message with a status of END SYNDICATION.
- The syndicated server may send a syndication termination message, which is used to inform the syndicating server that the syndicated server will no longer send syndication messages.

These represent a situation analogous to either party hanging up during a telephone conversation, either the caller signalling that they have finished the conversation, or the recipient of the call electing to end it.

6. Syndication in Action

Figure 5 shows an example of how syndication might work, in the context of the scenario described in Section 3. In this example, the military units provide a syndication device, which is compatible with both their own Universe and the Universe in which the civilian systems operate. The syndication device, can then be joined to the civilian C2 system, and be used to initialize syndication with the military C2 system. The military C2 system can then provide a suitably moderated version of the data available to it, which can be used by users of the civilian authority's C2 system.

In the scenario described in Section 3, where there is a requirement for mutual, bidirectional, data exchange between two SRUP Universes, the syndication approach described here could easily be adopted by both sides. Such a mutual syndication approach would enable each to be both syndicating and syndicated, each with their own set of rules about data sharing, processing action requests, and so on.
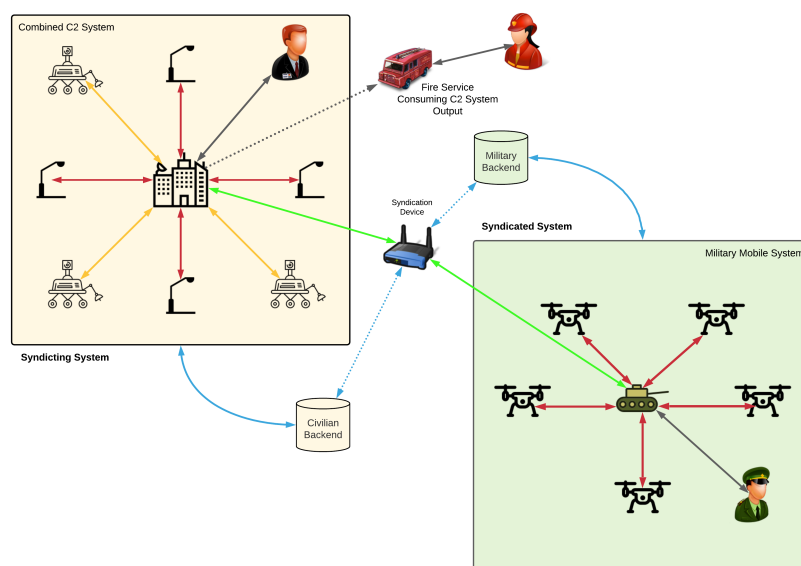
Figure 5. An example of syndication in action—showing how output from a restricted-access system can be syndicated with existing fixed, and mobile sensors.

7. Experimental Implementation

An implementation of the syndication approach was created for the Secure Remote Update Protocol, building upon the previously released C++ and Python libraries. This implementation was then utilized to conduct a small-scale table-top experiment. Both syndicating and syndicated C2 networks of simple IoT devices were built, and syndication operations were demonstrated. This adopted a web-based C2 interface, and enabled a subset of the data from devices joined to the syndicated network to be viewed in the C2 system for the syndicating network.

An example of the web-based user interface for the two C2 systems, showing syndication in progress, can be seen in Figure 6, and the hardware used for this is shown in Figure 7.
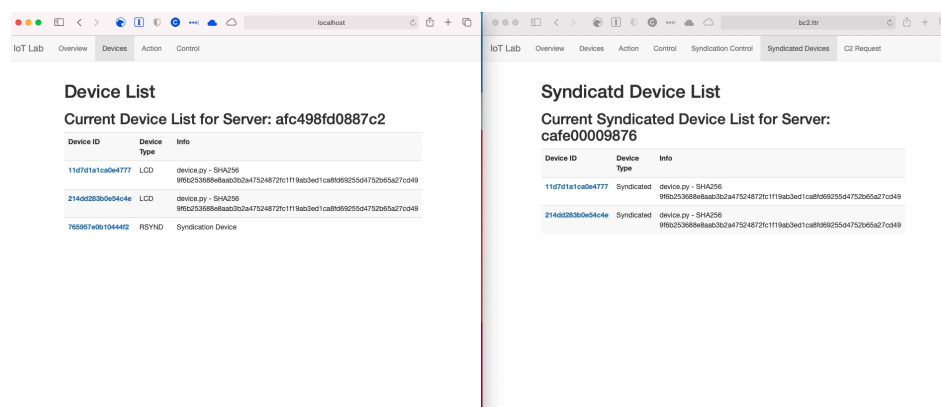

Figure 6. A screenshot from the example implementation of the syndication in action.

A video depicting the software implementation, and which illustrates the syndication concept in action, can be found at: https://youtu.be/F0_qlqh0Oiw (accessed 29 April 2021).

A full version of the software implementation for this functionality is available at: https://doi.org/10.5281/zenodo.4575539 (accessed 29 April 2021) [18] .

Figure 7. The Raspberry Pi based hardware used in the experimental assessment of the research.

8. Guest User

Although the concept of syndication was conceived in relation to wanting to share data between extant C2 systems, the concept can also be used to solve the problem of how to provide guests with limited or partial access to smart devices in an area they are visiting.

The problem of providing guest access is not new, but as the prevalence of IoT devices increases [19], the degree to which they become tightly integrated with their C2 systems is likely to grow. As a consequence of increased complexity, the full range of a device's functionality may not be accessible without a user also having access to the full C2 system's user interface via, for example, an app. Although simply giving a guest access to a building's C2 system is simple, the challenge is to be able to securely revoke that access once the guest is no longer permitted access.

By adopting SRUP syndication, the guest user could add access to the permitted devices to their existing C2 instance, integrating them alongside any other devices that they either own or have been granted access to. Guest access can be securely revoked by the C2 system after a designated period of time has elapsed, or the device's owner manually revokes access, without effecting other user's access. The use of syndication would also make it possible to dynamically moderate the data shared, based on criteria such as location, time of day, or the state of other devices within the system. For example, with this solution access could be limited to daytime access, or only when the user is at the location in question. Similarly, 'the state of other devices within the system' may relate to, for example, smoke detector activation also unlocking windows or doors.

An example of the SRUP syndication solution can be seen in Figure 8. Here the guest user is given time-bounded access to a subset of the devices on the owner's C2 system, in order to facilitate access to only a selection of available devices.

Although the initial implementation of syndication within SRUP C2 servers was built to permit just one syndication session at any time, the underlying SRUP protocol has no such restrictions and a C2 server could support simultaneous bidirectional, syndication to multiple other systems.

Although this approach flies somewhat in the face of the current walled garden approach to IoT device control, where each new device typically requires its own bespoke app to permit the user to interact with it, it aligns well with the growing hub-based approach used to permit devices (such as digital assistants) to control multiple systems of devices from a single interface. Although currently this latter approach does not use a

unified C2 model, and instead relies on Application Program Interface (API) hooks into Representational State Transfer (REST) interfaces [20], the adoption of such an approach as described here and previously [3] would enable superior integration, whilst still enabling more advanced functionality to be controlled from a dedicated app.

The adoption of open standards for IoT C2, alongside the provision of freely available Open Source Software (OSS) implementations of backend services, would also help to eliminate the problems caused when hardware vendors remove support for older products. If manufacturers adopted open standards, then instead of leaving customers with unusable devices when vendors cease provision of backend services, those customers would be able to provision their own backend services (either locally or via cloud-based servers) or purchase third party provision from a service provider.
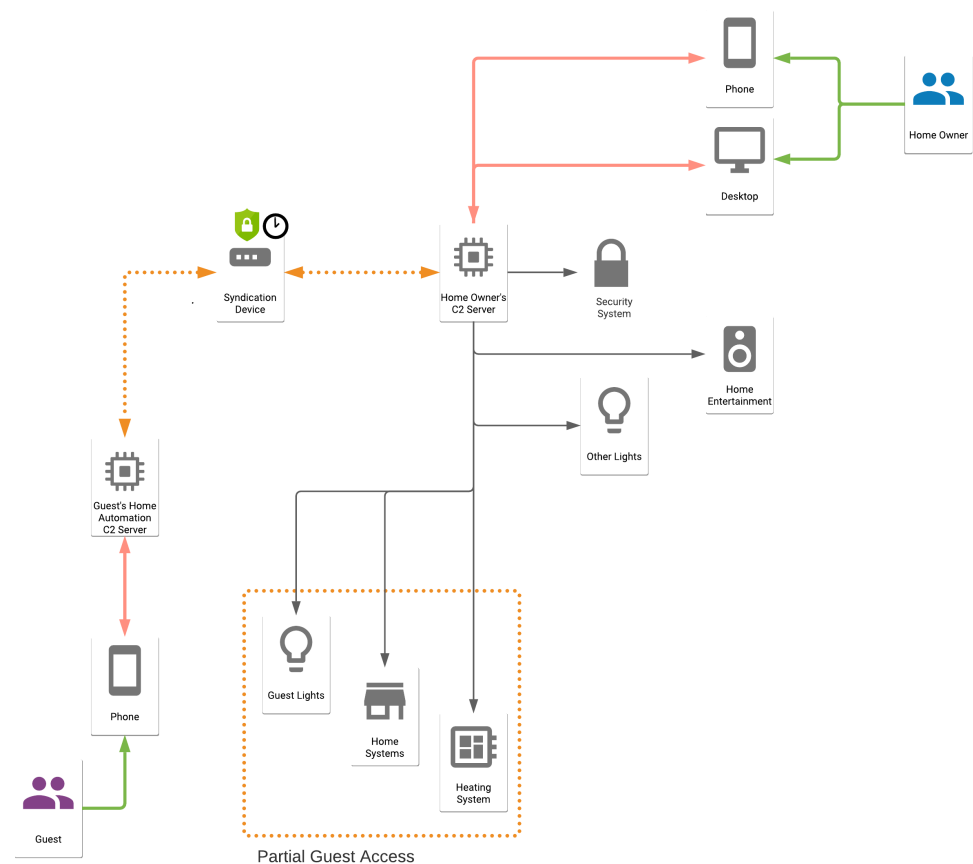

Figure 8. An illustration of the integration of guest access to a C2 system, utilizing syndication to provide moderated access to parts of the system.

9. Conclusions

Given the increase in utilization of IoT devices, the need to facilitate sharing of data between discrete C2 networks is important in order to provide dynamic cooperation between the operators of deployed services, especially in the context of future smart cities where standing networks of sensors or other devices may need augmentation with additional capabilities provided by third-parties during times of emergency or crisis. This paper has outlined the concept of syndication for C2 messaging for the IoT, and we have shown how this can provide moderated access to both data and services from IoT devices across different organizational boundaries (such as civilian and military authorities). Using these techniques, we have seen how data and device control can be shared from one C2 system to another, and how the owning C2 server is able to constrain that data sharing according to static or dynamic rules.

We have also demonstrated how the SRUP protocol can be expanded to include syndication-related messages, and how this can be used across different backend and cryptographic systems.

Additionally, we have examined the problem of provision of time-limited access to systems of IoT devices by guest users, and the need for revocation of their access at the end of their use. We have shown how syndication may be applied to solve this, and how it may also be used to constrain access to specific device data or features.

Further research in this area would be beneficial in order to explore the use of syndication in support of time- or activity-bounded guest access to location-specific devices, and around the necessary architectures and infrastructure required to enable the utilization of the SRUP protocol in the context of mass-market devices.

Author Contributions: Conceptualization, methodology, software, validation, and investigation, A.J.P.; writing—original draft preparation, A.J.P.; writing—review and editing, A.J.P. and S.J.C.; supervision, S.J.C. All authors have read and agreed to the published version of the manuscript.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Strous, L.; von Solms, S.; Zúquete, A. Security and Privacy of the Internet of Things. *Comput. Secur.* **2021**, *102*, 102148. [CrossRef]
2. Poulter, A.J.; Johnston, S.J.; Cox, S.J. SRUP: The Secure Remote Update Protocol. In Proceedings of the 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), Reston, VA, USA, 12–14 December 2016; pp. 42–47. [CrossRef]
3. Poulter, A.J.; Johnston, S.J.; Cox, S.J. Extensions and Enhancements to "the Secure Remote Update Protocol". *Future Internet* **2017**, *9*, 59. [CrossRef]
4. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; Ayyash, M. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 2347–2376. [CrossRef]
5. Heđi, I.; Špeh, I.; Šarabok, A. IoT network protocols comparison for the purpose of IoT constrained networks. In Proceedings of the 2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 22–26 May 2017; pp. 501–505. [CrossRef]
6. Rescorla, E. The Transport Layer Security (TLS) Protocol Version 1.3. Internet Engineering Task Force. RFC 8446. 2018. Available online: https://tools.ietf.org/html/rfc8446 (accessed on 29 April 2021).
7. Poulter, A.J.; Ossont, S.J.; Cox, S.J. Enabling the Secure Use of Dynamic Identity for the Internet of Things—Using the Secure Remote Update Protocol (SRUP). *Future Internet* **2020**, *12*, 138. [CrossRef]
8. Benazzouz, Y.; Munilla, C.; Günalp, O.; Gallissot, M.; Gürgen, L. Sharing User IoT Devices in the Cloud. In Proceedings of the 2014 IEEE World Forum on Internet of Things (WF-IoT), Seoul, Korea, 6–8 March 2014; pp. 373–374. [CrossRef]
9. Colombo, P.; Ferrari, E.; Tümer, E.D. Regulating Data Sharing across MQTT Environments. *J. Netw. Comput. Appl.* **2021**, *174*, 102907. [CrossRef]
10. Ravidas, S.; Lekidis, A.; Paci, F.; Zannone, N. Access control in Internet-of-Things: A survey. *J. Netw. Comput. Appl.* **2019**, *144*, 79–101. [CrossRef]
11. Koo, J.; Oh, S.R.; Lee, S.H.; Kim, Y.G. Security Architecture for Cloud-Based Command and Control System in IoT Environment. *Appl. Sci.* **2020**, *10*, 1035. [CrossRef]
12. Raglin, A.; Metu, S.; Russell, S.; Budulas, P. Implementing Internet of Things in a military command and control environment. In Proceedings of the Next-Generation Analyst V, Anaheim, CA, USA, 10–11 April 2017; Hanratty, T.P., Llinas, J., Eds.; International Society for Optics and Photonics: Bellingham, WA, USA, 2017; Volume 10207, pp. 71–81. [CrossRef]
13. Wei, H.; Cai, Y.; Li, H.; Li, D.; Wang, T. Sambot: A self-assembly modular robot for swarm robot. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AZ, USA, 3–7 May 2010; pp. 66–71. [CrossRef]
14. Deng, H.; Qin, Z.; Sha, L.; Yin, H. A Flexible Privacy-Preserving Data Sharing Scheme in Cloud-Assisted IoT. *IEEE Internet Things J.* **2020**, *7*, 11601–11611. [CrossRef]
15. Khelloufi, A.; Ning, H.; Dhelim, S.; Qiu, T.; Ma, J.; Huang, R.; Atzori, L. A Social-Relationships-Based Service Recommendation System for SIoT Devices. *IEEE Internet Things J.* **2021**, *8*, 1859–1870. [CrossRef]

16. Eremia, M.; Toma, L.; Sanduleac, M. The Smart City Concept in the 21st Century. *Procedia Eng.* **2017**, *181*, 12–19. [CrossRef]
17. Rivest, R.L.; Shamir, A.; Adleman, L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
18. Poulter, A.J. *Secure Remote Update Protocol*; Version 5.0; Open Source Software: Southampton, UK, 2021. [CrossRef]
19. Gartner Predicts the Future of Cloud and Edge Infrastructure. Gartner. 2021. Available online: http://www.gartner.com/smarterwithgartner/gartner-predicts-the-future-of-cloud-and-edge-infrastructure/ (accessed on 29 April 2021).
20. Fielding, R.T.; Taylor, R.N. Principled Design of the Modern Web Architecture. In Proceedings of the 22nd International Conference on Software Engineering, Limerick, Ireland, 4–11 June 2000; Association for Computing Machinery: New York, NY, USA, 2000; pp. 407–416. [CrossRef]