How to Prevent Misuse of IoTAG?

Bernhard Weber <bernhard1.weber@st.othr.de>[1]
Lukas Hinterberger[2] Sebastian Fischer[1]
Prof. Dr. Rudolf Hackenberg[1]

[1]Ostbayerische Technische Hochschule Regensburg

[2]Freie Universität Berlin

20.04.2021

- **Name**: Bernhard Weber
- **Education**:
  > 2014 – 2018: OTH Regensburg (B.Sc. Technical Computer Science)
  > 2019 - Current: OTH Regensburg (M.Sc. Applied Research)
- **Research Area**: IoT Security
- **Previous Publications**:
  S. Fischer, K. Neubauer, L. Hinterberger, B. Weber, and R. Hackenberg, "**IoTAG: An Open Standard for IoT Device IdentificAtion and RecoGnition**," in SECURWARE 2019, Thirteenth International Conference on Emerging Security Information, Systems and Technologies, 2019, pp. 107-113.

  L. Hinterberger, B. Weber, S. Fischer, K. Neubauer, and R. Hackenberg, "**IoT Device IdentificAtion and RecoGnition (IoTAG)**," CLOUD COMPUTING, 2020, pp. 17-23, 2020.

  L. Hinterberger, B. Weber, S. Fischer, K. Neubauer, and R. Hackenberg, "**Extended Definition of the Proposed Open Standard for IoT Device IdentificAtion and RecoGnition (IoTAG)**," International Journal on Advances in Internet Technology, Vol. 13, 2020, pp. 110-121, 2020.

Content

- Open Standard for **IoT** Device Identific**A**tion and Reco**G**nition [1]
- Ability for IoT devices to provision security-related information
- Well-defined communication channels
- Encrypted and signed data transmission
- Based on existing standards

- Iot devices are a potential security threat for a network
- Wish for an automated risk evaluation
  > Information about devices needed
  > Existing network scanning methods can be manipulated
  → Devices have to provide information themselves

- Current IoTAG version has no access restriction
- The transmitted metadata in IoTAG could provide useful information for an attacker

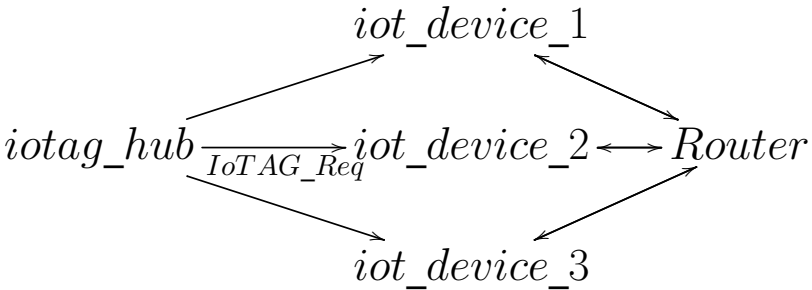


Figure: Example network using IoTAG.

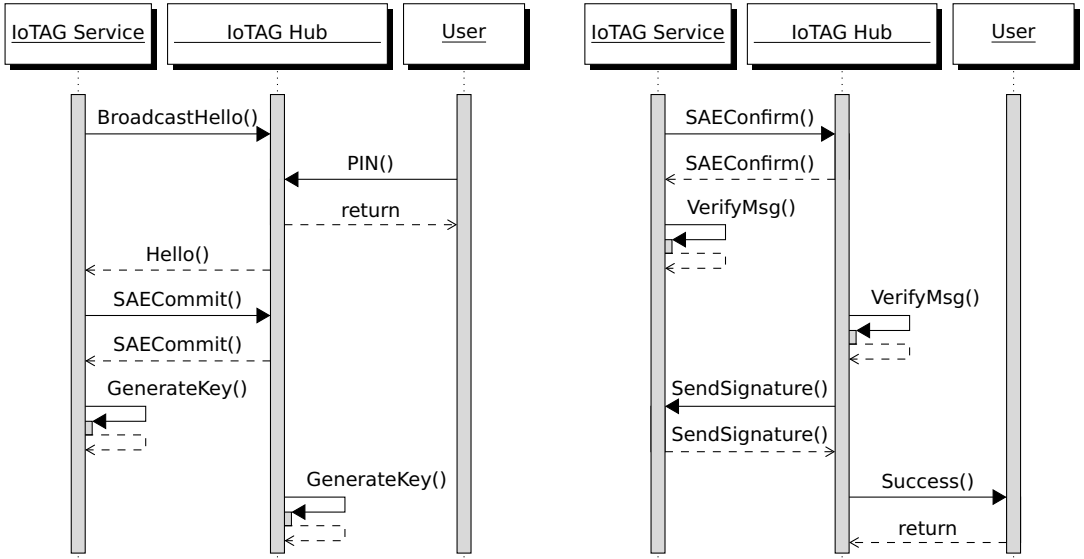
- **Target**: A third party is able to receive an IoTAG without the user's permission
- Attacker has gained access to the network (e.g. weak WiFi, compromised device)
- Attacker has the ability to capture all traffic of the network
- Attacker could also be a legit device, which wants to collect user data

- IoTAG uses HTTP over TLS for the communication between the hub and the devices [2]
- HTTP offers basic authentication methods, which rely on a secure channel to be secure [3]
- TLS includes the ability for server and client authentication, which is based on verifying each others signature and confirming the access to a private key [4][5]
- Secure Channel needs to be between the hub and a device, without anything in between
  → Signatures need to be verified anyway, so HTTP authentication is not needed

- The pairing process needs to exchange both signatures
- It needs to verify that the hub and the device are legit and not only pretending to be the correct device
- It has to make sure that the user has authorized the pairing
- Listening to the messages sent by both devices must not reveal any information or authentication data

- When no hub is paired, IoTAG is disabled
- Each device has a secret (PIN), which is at least decimal and four digits long
- Pairing is only available for a limited time (between 1 and 10 minutes) after each restart or pairing button (optional) press and is limited to three tries per time frame
- During this time the device is broadcasting a "Hello"-message on the network
- The simultaneous authentication of equals algorithm (SAE) is used for the key exchange [6]
- AES-256 with Cipher Block Chaining (CBC) is used to encrypt the signature [7][8]

Sequence


IoTAG Service
IoTAG Hub
User
BroadcastHello()
PIN()
return
Hello()
SAECommit()
SAECommit()
GenerateKey()
GenerateKey()
IoTAG Service
IoTAG Hub
User
SAEConfirm()
SAEConfirm()
VerifyMsg()
VerifyMsg()
SendSignature()
SendSignature()
Success()
return

Conclusion

- Using authentication helps limiting access to the IoTAG
- The use of a PIN ensures that the pairing is initiated by the user
- SAE provides a secure way to get a high-quality key using a low-entropy shared secret and an implementation should already be present as it is part of the latest WiFi standard
- The verification of TLS certificate signatures is a secure and lightweight solution for authentication, as it is also already implemented by most IoT devices

References

[1] L. Hinterberger, B. Weber, S. Fischer, K. Neubauer, and R. Hackenberg, ”IoT Device IdentificAtion and RecoGnition (IoTAG),“ CLOUD COMPUTING, 2020, p. 17, 2020.

[2] E. Rescorla, ”HTTP Over TLS,“ RFC 2818, RFC Editor, May 2000, doi:10.17487/RFC2818.

[3] J. Franks et al., ”HTTP Authentication: Basic and Digest Access Authentication,“ RFC 2617, RFC Editor, June 1999, doi:10.17487/RFC2617.

[4] T. Dierks and E. Rescorla, ”The Transport Layer Security (TLS) Protocol Version 1.2,“ RFC 5246, RFC Editor, August 2008, doi:10.17487/RFC5246.

[5] D. Cooper et al., ”Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile,“ RFC 5280, RFC Editor, May 2008, doi:10.17487/RFC5280.

[6] I. . W. Group, ”IEEE Standard for Information Technology–Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks–Specific Requirements - Part 11: Wireless LAN Medium Access Control(MAC) and Physical Layer (PHY) Specifications,“ IEEE Std802.11-2020 (Revision of IEEE Std 802.11-2016), pp. 1–4379, 2021, doi:10.1109/IEEESTD.2021.9363693.

[7] V. Rijmen and J. Daemen, ”Advanced encryption standard,“ Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology, pp. 19–22, 2001.

[8] M. J. Dworkin, ”Recommendation for Block Cipher Modes of Operation: Methods and Techniques,“ National Institute of Standards and Technology, 2001.