Analyzing Time Based Security in IoT Devices

Sammy Gill, Miranda Go, Kavya Ramamoorthy, Meredith Zhou, Barbara Vargas

Advisor: Deian Stefan

ERSP UC San Diego Early Research Scholars Program

UCSDCSE Computer Science and Engineering

Motivation

Internet of Things (IoT) devices are being adopted at a rapid rate in homes and businesses. However, there is currently no systematic way to implement proper security within such devices, leading to many issues. To avoid this, we needed a way to streamline security by efficiently implementing proper security checks. We thus focused on time based security policies to help determine how we can streamline security.

Background

Our time based security policy research was inspired by the work Hails[1], a framework that implements access control by labeling every piece of user data in web applications. These labels define who can read or write to the data. However, they do not involve any notion of time. Thus, it is difficult to apply Hails directly to create an IoT device with time based security policies.

Problem

Through some initial research, we realized that there are currently no data structures that can store time-based security information efficiently. To fix this, we focused on IoT systems that involve time based interactions, since many IoT devices emphasize time based access.

Approach

- Created a web application and physical lock device to implement security policies needed for time-based IoT devices.
- Created a security policy that placed restrictions on users directly
- Allowed us to easily incorporate time into read and write accesses for each user individually
- Developed a Node.js module that abstracted security code for server side
- Creating a Domain Specific Language(DSL) to further generalize time-based security for IoT devices

Current Data Structure

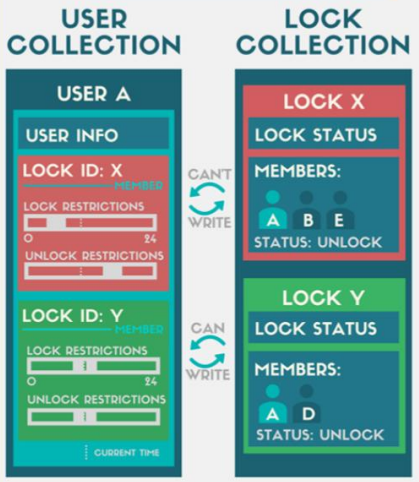


Figure 1: There are currently no data structures that store a relationship between users and hardware efficiently, which resulted in our complex data model.

Modified Structure

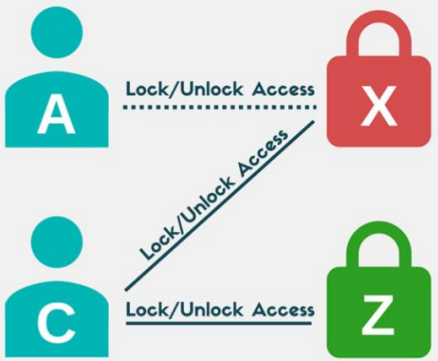


Figure 2: The relationships between users and locks are generalized into a simpler structure, much like a bipartite graph. We believe that incorporating this idea into a DSL would result in more efficient security policy enforcement.

State Machine for the Lock

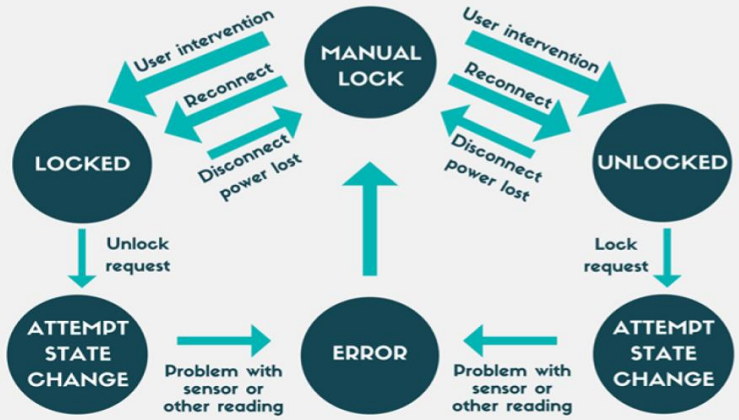
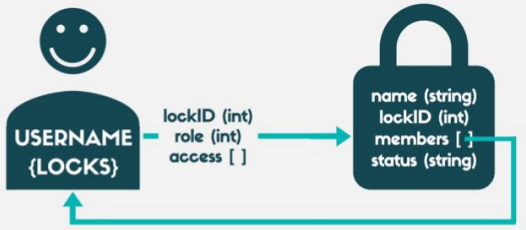


Figure 3: This diagram represents the actions of the locks in different situation, as well as its ability to analyze the validity integrity of data being collected from hardware.

Security Policy Enforcement



Example of the security policies:

- For lock L, user A can lock/unlock L at time T if:
  isOwner(A, L) ∧ isAdmin(A, L) ∧ (isMember(A, L) ∨ withinBound(A, L, T))
- For lock L, user A can add user B as an admin of lock L if:
  isOwner(A, L) ∧ isMember(B, L)

Future Work

Based off of the work mentioned in our contribution, we will be proposing solutions which may include:

- A DSL with a labeling based approach to develop security policies
- A database system that allows easy development of data models that include relationships between users and hardware

Acknowledgements

Many thanks to Deian Stefan, Christine Alvarado, Aditi Mavalankar, Gary Soeller for all their guidance and support.
This material is based upon work supported by the National Science Foundation under Grant No. CNS-1339335

References

1. D.B. Griffin, A. Levy, D. Stefan, D. Terei, D. Mazières, J.C. Mitchell. Hails: Protecting Data Privacy in Untrusted Web Applications. In *Proc. of the 10th OSDI*, pages 47-60.USENIX,2012.