RESEARCH REVIEW 2019

KalKi: High Assurance Software-Defined IoT Security

The term "KalKi" is of Sanskrit origin, and is the name of an avatar of the god Vishnu, who is the destroyer of filth and malice and usherer of purity, truth and trust.

Problem

Despite the DoD's current use of Internet of Things (IoT) devices in supervisory control and data acquisition (SCADA) systems, and its interest in using such devices in tactical systems, adoption of IoT has been slow mainly due to security concerns (e.g., reported vulnerabilities, untrusted supply chains).

At the same time, the DoD recognizes the rapid pace at which the IoT commercial marketplace is evolving, and its urgency to embrace commodity technologies to match its adversaries.

Solution

Move part of security enforcement to the network to enable the integration of IoT devices into DoD systems, even if the IoT devices are not fully trusted or configurable, by creating an IoT security platform that is provably resilient to a collection of prescribed threats.

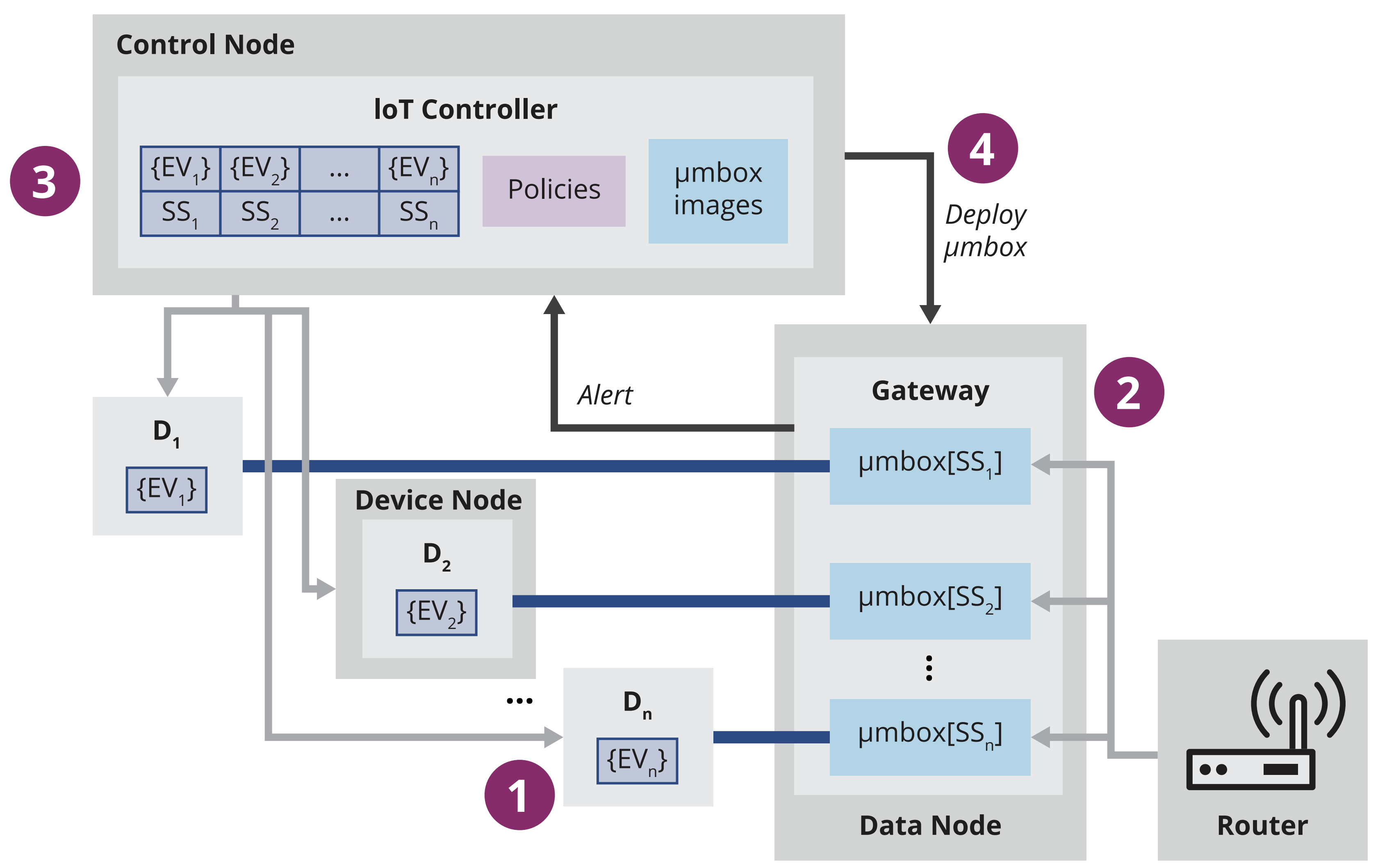
The "Software-Defined" Aspect

Use software-defined networking (SDN) and network function virtualization (NFV) to create a highly dynamic IoT security platform.

The "High Assurance" Aspect

Use überSpark (a framework for building secure software stacks) to incrementally develop and verify security properties of elements of the software-defined IoT security platform.

The KalKi IoT Security Platform **enables the integration of IoT devices** into DoD systems, even if the IoT devices are **not fully trusted** or configurable.



1 Each IoT Device D senses/controls a set of environment variables EV

2 Network traffic to/from each device is tunneled through μmboxes that implement the desired network defense for the current system state
D_1 μmbox[SS_1] = Firewall
D_2 μmbox[SS_2] = IPS, ...

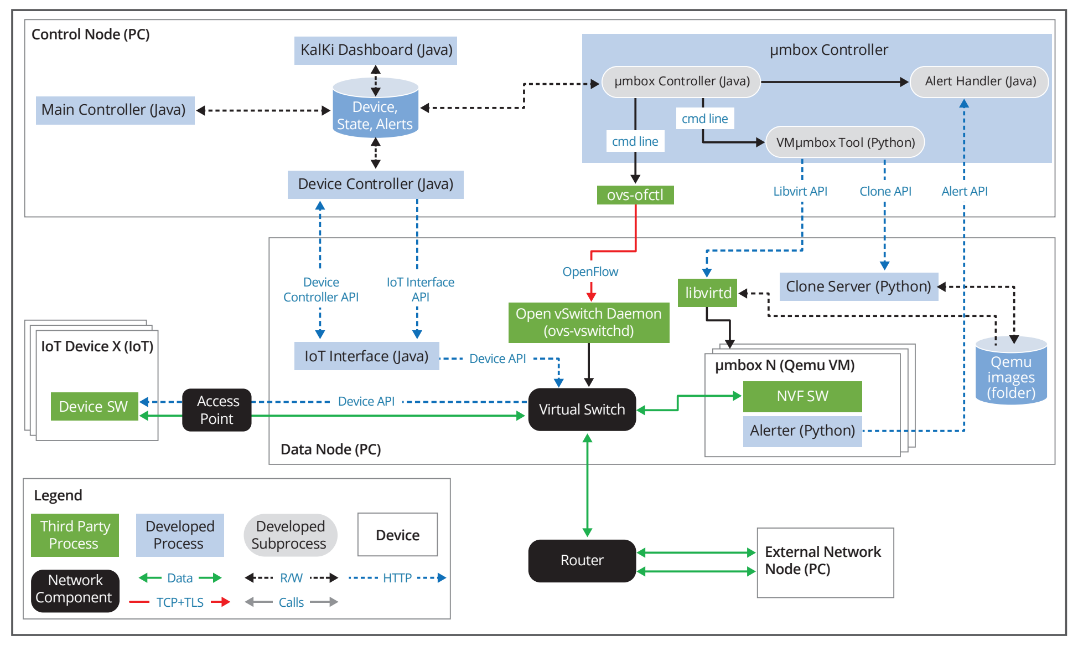
3 IoT Controller maintains a shared statespace composed of {EV} and security state (SS) for each device
SS= {Normal, Suspicious, Attack}

4 Changes in the shared statespace are evaluated by policies and may result in the deployment of new μmbox(es)

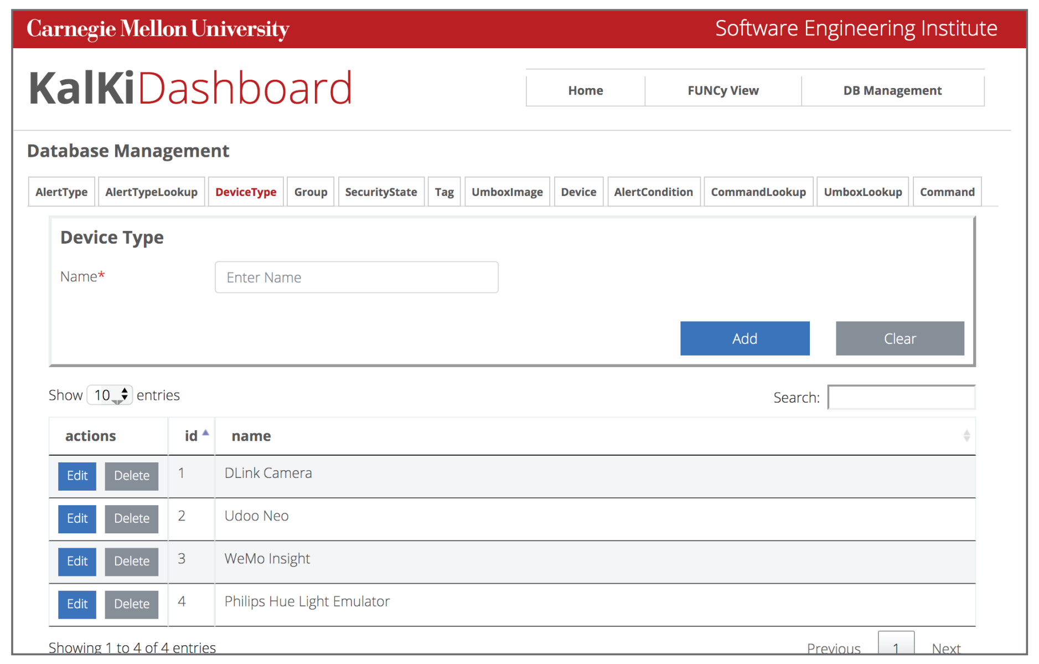
Sensitive areas of the system are protected via FUNCY Views – a novel, performant, isolated execution environment extension to überXMHF/überSpark

- **State machine** that controls the security state transitions for each IoT device in the Control Node
- **Routing tables** and other sensitive data structures used by Open vSwitch (OVS) in the Data Node

Year 2 Highlights



End-to-end prototype of IoT Security Platform implemented and tested with different attack scenarios. Policies and μmboxes implemented for four representative devices.



Database Management

Device Type

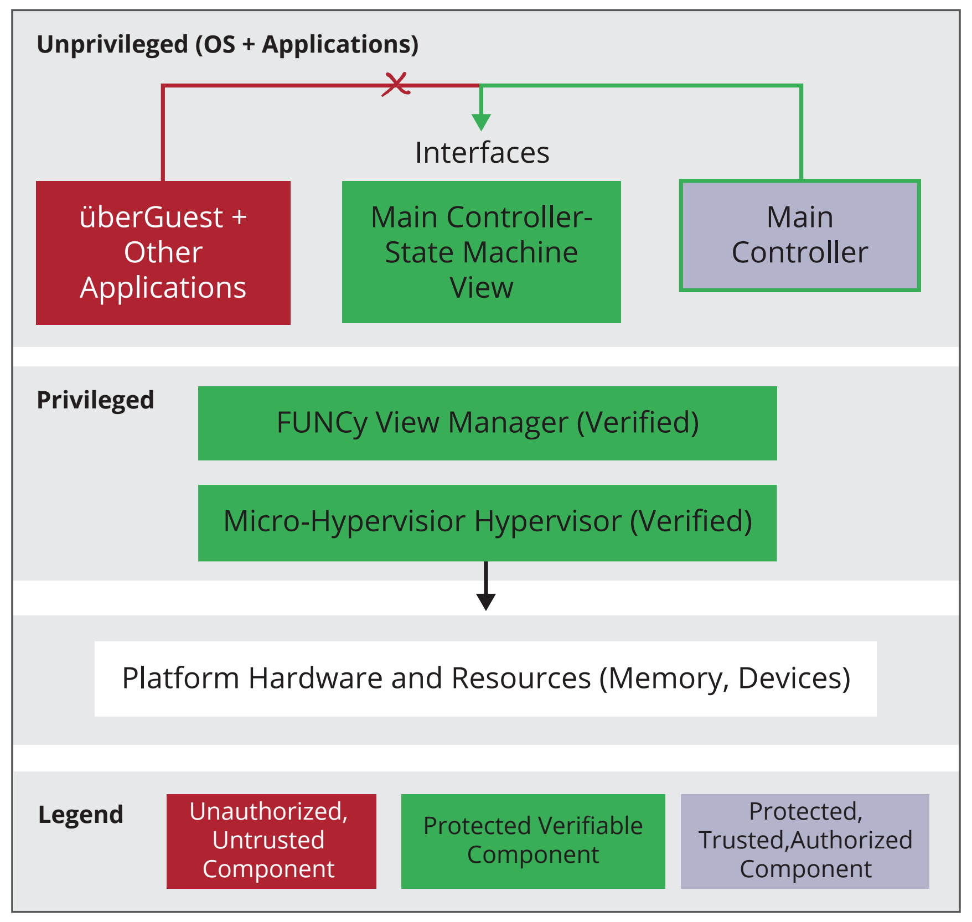
Name* Enter Name

Show 10 entries Search:

actions	id	name
Edit Delete	1	DLink Camera
Edit Delete	2	Udoo Neo
Edit Delete	3	WeMo Insight
Edit Delete	4	Philips Hue Light Emulator

Showing 1 to 4 of 4 entries

Updated version of the KalKi Dashboard, which allows to fully configure and monitor the system.


Unprivileged (OS + Applications)
Interfaces
überGuest + Other Applications
Main Controller-State Machine View
Main Controller
Privileged
FUNCy View Manager (Verified)
Micro-Hypervisior Hypervisor (Verified)
Platform Hardware and Resources (Memory, Devices)
Legend
Unauthorized, Untrusted Component
Protected Verifiable Component
Protected, Trusted,Authorized Component

überXMHF/überSpark extended to include überobject protections for sensitive areas of the Control node and Data node.

Carnegie Mellon University
Software Engineering Institute

Sebastián Echeverría | secheverria@sei.cmu.edu
Chris Grabowski, Dr. Grace Lewis, Craig Mazzotta, Matthew McCormack, Marc Novakouski, Kyle O'Meara, Dr. Vyas Sekar, Dr. Amit Vasudevan

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.
The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

DM19-1028