**IoT Security Development Framework for Building Trustworthy Smart Car Service notes**

Jonh Cavalieri, Anirban Mookherjee, Nicholas Conlon

1. **INTRODUCTION**

The rapid development of Internet of Things has dramatically increased the vulnerability of the interdependent systems. In this paper, the authors focus on one emerging IoT service associated with smart and autonomous cars that will have major security problems. Past research has shown that when a single control unit is compromised, a capable attacker may gain access to other vehicle units via internal communication buses such as controller area network (CAN), and attack critical subsystems. The authors of this paper introduce an IoT hierarchical architecture that is then extended to the IoT Security Development Framework (ISDF). The objective the ISDF is to enable developers to address security issues in a systematic way while designing and developing each IoT layer rather than considering security as an afterthought and in ad-hoc manner. In our approach, IoT hierarchy consists of four layers: Application, Service, Communications and End-Devices layers. By insuring for each layer that all existing vulnerabilities and threats can be identified and mitigation solutions will be applied, the ISDF will provide the architectural support to deliver trustworthy IoT services that can: 1) Protect IoT services against epidemic attacks; 2) Ensure that critical IoT systems can survive faults and destructive attacks; and 3) Ensure IoT security and privacy. Development of a trustworthy Vehicle Information and Management Portal (VIMP) services to support smart car applications is also discussed.

1. **BACKGROUND**

CANCar Area Network is a distributed protocol of short messages (signals and measurement values). Major faults with CAN:

* + Node stuck-at-bit fault - faulty node sends constant bit value preventing other nodes from communicating
  + Medium break-up fault - CAN bus split due to damage
  + Babbling idiot fault - node floods CAN bus with high priority messages

**Smart Vehicles Cyber Security**

Smart Vehicle (SV) networks distributed frameworks leave it open to attack. Types of attacks:

* + indirect physical attack
  + short-range wireless access
  + long range wireless access

Autonomous Car (AC) is a special category of SV. An AC has five basic functions:

* + Perception - sense surrounding environment
  + Localization - find position
  + Planning - determine future motion
  + Control - guidance following plan
  + System Management - supervise the overall system

**IoT Cyber Security**

Traditional IT security solutions not applicable to IoT because:

* + IoT extends internet to all devices
  + Smaller devices may not support complex security protocols
  + Multiple entry points and vulnerabilities when all "things" communicate
  + Device IoT services could be shared

**Intrusion Detection System (IDS) and threat model**

Current cyber security solutions are not effective. Two basic intrusion detections techniques:

* Signature based - builds a database of known attack signatures
* Anomaly based – detect new attacks by defining a baseline model of normal behavior.

A threat model is used during design to mitigate potential threats. Steps to create a threat model:

* + Identify attackers
  + Rank threats
  + Choose mitigation strategies
  + Build mitigation solutions

1. **IoT ARCHITECTURE FOR SMART CAR SERVICES**

IoT SV services use hierarchical architecture:

* + Layer 1 - physical devices and statuses
  + Layer 2 - communication layer between low level devices
  + Layer 3 - services layer provides common middleware and functions
  + Layer 4 - application layer provides custom apps for user needs

1. **IoT SECURITY DEVELOPMENT FRAMEWORK (ISDF)**

**Secure and Trustworthy Services**

Trustworthy service - one which can self-protect against cyber attack and continue to operate normally despite faults

IoT security is maintained in all layers by using:

* + Authentication - secure identification
  + Authorization - limiting access
  + Integrity - detect whether an object was modified illegally
  + Non-repudiation - prove messages have been sent and received

**IDSF Architecture**

2-D architecture with four layers, each layer has four planes. Descriptions below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Attack Surface** | **Target** | **Impact** | **Mitigation Mechanism** |
| **End-Devices Layer** | | | |
| Controllers | Control, information | Control, human life, safety, time | IDS, behavioral analysis |
| Sensors | Information, access to system | Control, human life, safety, money, energy | Lightweight encryption, IDS, behavior analysis, sensor authentication |
| Actuators | Control | Control, human life, safety, time, money | Lightweight encryption, IDS, ABA, anti-jamming |
| Entertainment | Access to the system | Time, energy, money | Encryption, moving target defense, ABA |
| **Communications Layer** | | | |
| Protocols | Access, information, control | Control, human life, safety, time, money, energy | Authentication, access control, IDS, ABA, anti-jamming |
| Firewalls | Access to system | Time, money, reputation | IDS, behavior analysis, authentication |
| Routers | Access, information | Control, human life, safety, time | IDS, ABA, anti-jamming |
| Comms Bus | Information, control | Privacy, money, human life, safety, time | Encryption, IDS, ABA, moving target defense |
| **Services Layer** | | | |
| Cloud storage | Personal and confidential information | Information, money, time , safety | Encryption, IDS, moving target defense, analysis |
| Web services | Control, monitor | Control, human life, safety, money | Authentication, IDS, behavior analysis |
| **Applications Layer** | | | |
| Mobile devices | Information control | Human life, safety, loss of personal info and money | Authentication, access control, IDS, behavior analysis |
| Programs and apps | Access to the system, control, information | Time, money, safety, reputation | IDS, behavior analysis, authentication |

1. **VEHICLE INFORMATION AND MANAGEMENT PORTAL (VIMP) SERVICES**

The VIMP will provide 24/7 visibility into the smart car operational states (e.g., engine conditions, entertainments, etc.) and also provide the mechanisms to secure and protect the smart car operations and functions. Figure below shows the main components of the VIMP fields that can be maintained at the portal and the deployment approach for VIMP services.



1. **SMART VEHICLE TESTBED AND ABA-IDS METHODOLOGY**

* Authors designed IOT security development framework and tested on smart-vehicle testbed.
* The SV testbed contains all the characteristics and functionalities of actual smart-vehicles: sensors, actuators, automation controls, communication channels.
* User can monitor all SV variables and control all SV components using variety of protocols Bluetooth, WiFi, I2C
* SV variables include temp, distance, motion, illumination
* SV controllable components include lights on/off, movement guidance (Left/Right/Forward/Back), brakes and speed
* Monitoring and controlling SV testbed can be accessed locally through secure gateway or remotely via VIMP (vehicle information mgmt portal)
* VIMP is web-server that monitors and analyzes in real-time the SV IOT hierarchical framework to detect and respond to attacks and system faults.
* For example—an attacker may target a temperature sensor at the End Devices layer of the SV
  + temperature sensor has little computational power → can’t use encryption
  + instead, the VIMP will monitor sensor output for anomalies to detect attacker
* Anomalous Behavior Analysis implemented via a module stack
  + modules: continuous monitoring, structure of data, ABA, sensor classification, recovery actions
* Continuous Monitoring Module:
* analyze behavior of End Device layer components to characterize normal operations—source,destination, packet contents are extracted from protocol and forwarded to data structure analysis module for sensor ID and characterization of runtime profile (S-DNA) using discrete wavelet transform coefficients of the signal
* Data Structure Analysis Module:
* this module computes runtime profile based on DWT coefficients of signal so that it can be compared against reference profiles in ABA module and uses that runtime profile to determine ID of End Device layer component ( what sensor sent the signal, what actuator sent the signal etc)
  + runtime profile generated by DWT coefficients where original signal is decomposed into an approx coefficient and detail coefficient by applying high pass and low pass filters respectively—after decomposing signal into coefficients those coefficients are aggregated into vector that represents runtime profile (S-DNA)
  + The ID of the End Device layer component that is sending signal is determined using the runtime profile generated in previous step and computing Euclid distance between that vector and pre-computed reference vectors obtained in off-line training phase—so if runtime profile vector is closest to temperature sensor vector → then sensor is ID’ed as temperature sensor
* ABA module:
* Euclid distance is computed between runtime profile and reference profile for that particular sensor stored in ABA module (computed in off-line training phase). Five vector used to determine control limits for normal operation and euclid distance computed between all 5 reference vectors and current runtime vector. Those distances used to define Upper Control Limit and Lower Control Limit—these limits are passed down to classification module.
* Classification Module:
  + Once ABA module determine anomalous behavior, this module classifies the type of anomaly using Euclid dist e.g. for DOS attacks, euclid distance show sudden spikes above UCL.
* Recovery Module:
* if attack cannot be classified the data is rejected; else the user can be alerts, an re/authentication mechanism can be used on the component or the network config can be changed or the data can be discarded.

1. **EXPERIMENTS AND RESULTS**

* In off-line training phase, the matrix of normal operating limit is computed—this is the matrix used in ABA module to determine whether anomalies exist within component operations
* After training system under normal operating conditions, attacks launched against SV testbed to examine behavior of components under attack all in off-line phase.
* Window of 7 Euclid distances used to determine behavior trends
* false positives: when behavior not considered during training, after attack when sensor needs to move to its steady state

1. *Offline training phase*

* build the matrix for sensor’s classification
* launch attacks against that sensor to learn its behavior

Table below summarizes normal operation limits.



Figure below shows behavior for trained attacks



1. *Online testing phase*

Some values are noticeably above the UCL for the abnormal behavior while other values are in the normal behavior area. This happens because the attacks were performed several times during the experiment. Table below summarizes the detection and classification accuracy for each type of attack.



Pulse DoS and noise injection are two new attacks that were not considered in the offline phase. Here the system detects these attacks and classify them as “new attack”. Two cases that triggered false positives: 1) when the behavior is not considered in the training phase (e.g. environmental manipulation); and 2) When the sensor needs to reach its steady state after an attack. Experiments showed that at most 4.2% of these situations produced false positives alerts. Once the attack is detected and classified, the recovery actions to solve the problem include: 1) reject sensor’s data, 2) launch an alert, and 3) deauthenticate the sensor.