INF523: Assurance in Cyberspace Applied to Information Security

Student Case Studies (Part C)

Prof. Clifford Neuman

Lecture 13C
20 November 2020

Friday November 20

Operating Systems

- Linux Applications - Aditya Goindi
- Linux - Tejas Pandey
- Chrome OS - Malavika Prabhakar

Infrastructure and Vehicle Control Systems

- US Voting Infrastructure - Anthony Cassar

Vehicle Control Systems

- Autonomous Vehicles - Chris Samayoa
- Autonomous Vehicles - Amarbir Singh
- Connected and Automated Vehicles - Abhishek Tatti
- Tesla - Dwayne Robinson
- Avionics - Pratyush Prakhar

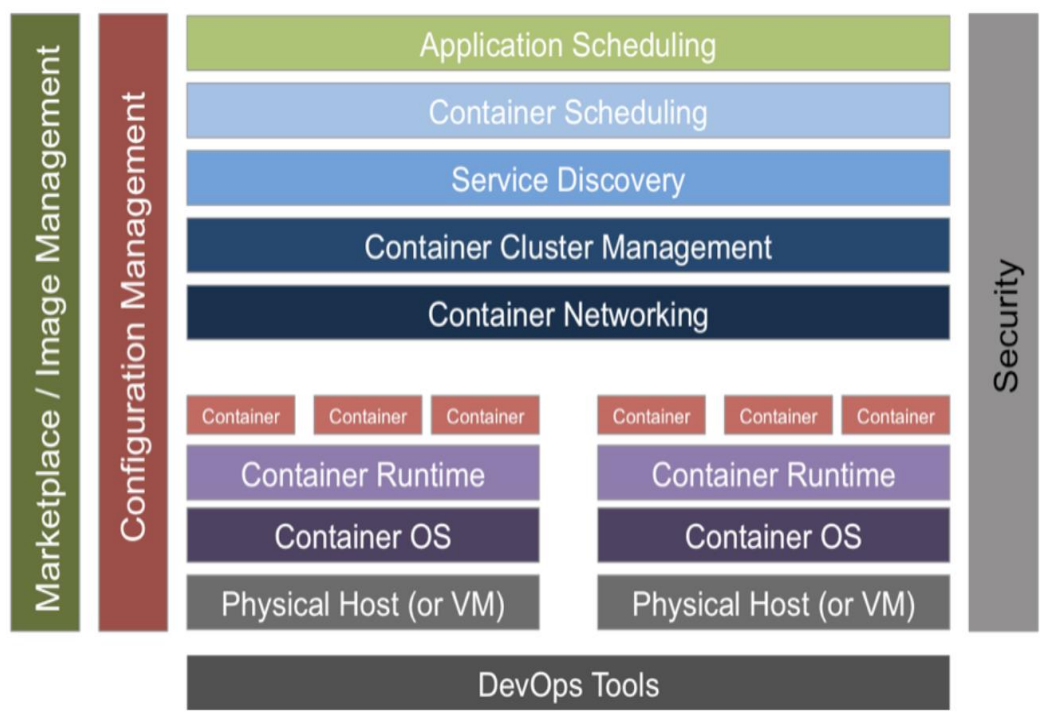
Assurance Requirements for Linux Application Container Deployments

Aditya Goindi

Introduction

- A **container** is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another.
- Application Containers are slowly finding adoption in enterprise IT infrastructures.
- Security countermeasures for & through six entities including
    - Hardware
    - Host Operating System (OS)
    - Container Runtime
    - Image
    - Registry
    - Orchestration
- Focus of this presentation is on security assurance requirements for security solutions for containers hosted on Linux

Container Technology Stack



Container Execution & Attack Surface

- First, the application is run within a container as a single operating system process.
- The container has a copy of the application code itself as well as the software stack (consisting of binaries and libraries)
- This stack can be assembled using library system, avoiding the need for the developer to build and configure the stack from scratch.
- The deployment model in a container architecture may involve running copies of the same application in parallel within separate containers, even spread across different container hosts.
- The infrastructure may have a mechanism to distribute incoming requests across all instances of the same application using some form of load balancer.
- Attack surface: Vulnerability in the application code of the container or its faulty configuration has been exploited by an attacker.
- This would allow the attacker to take control of and compromise the privilege code in container runtime and host OS kernel.

Security Solutions for Linux Application Container Stack

- Kernel Based
- There are two types of interfaces:
  - Kernel interfaces
    - Namespaces
    - cgroups
    - Capabilities
  - Kernel Loadable Module interfaces
    - SELinux
    - AppArmor
    - Seccomp

- Hardware Based
  - vTPM in host OS kernel
  - vTPM in a Dedicated Container

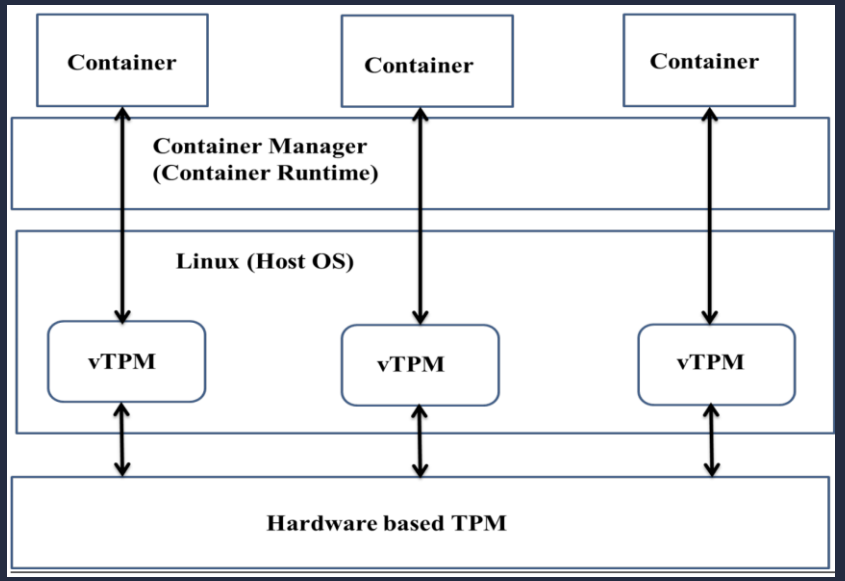
- Namespaces
  - Partitions filesystems, processes, users, network stacks, Inter-process communication (IPC) objects, host names, and other components into separate pieces.
  - Each filesystem namespace has its own root directory and mount table.
  - Assurance depends on:
    - methods used to enforce namespace isolation.
    - kind of metadata associated with each namespace that implements the appropriate access control.
- cgroups
  - Control Groups (Cgroups) are a kernel mechanism for specifying and enforcing hardware resource limits and access controls to a process or a group of processes.
  - Goal is to prevent a process from hogging all available resources and starving other processes and containers on the host.
  - Controlled resources include Central Processing Unit (CPU) shares, Random Access Memory (RAM), network bandwidth, and disk I/O
  - Security provided by cgroups:
    - Preventing Denial-of-Service Attacks
    - Device Integrity Protection
- Capabilities
  - Helps to partition the extensive set of privileges available to root.
  - Prior to the introduction of the Capabilities feature, a process that needs to open network sockets must be run as a root to perform this single function.
  - Security protection:
    - The security consequence of this is that the potential attackers would gain significantly fewer privileges.

Kernel Loadable Modules

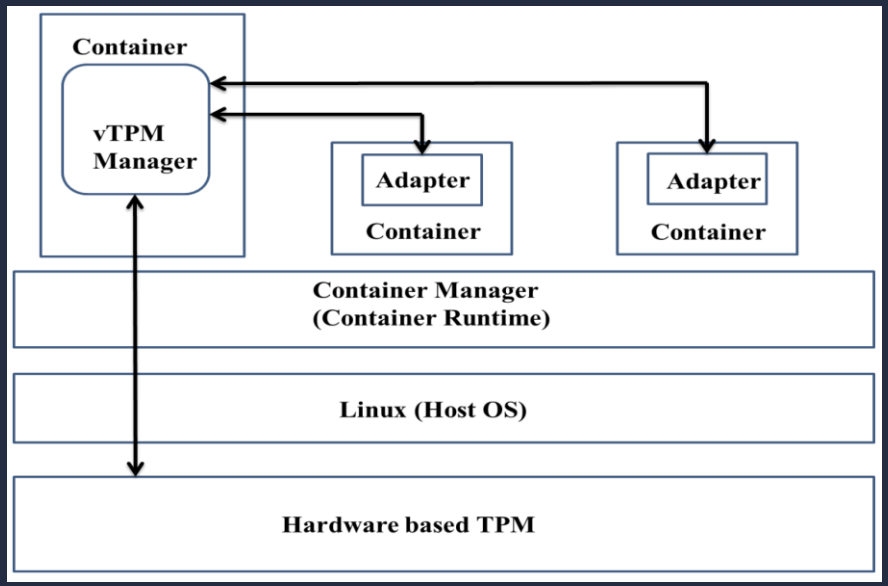
- SELinux
  - Security Enhanced Linux, can be used to assign categories to processes and objects (e.g., files, sockets) and specify access restrictions based on certain combinations of categories.
  - A specific SELinux label can be applied to a container to enforce a security policy (e.g., a container hosting a Webserver can only open ports 80 or 443)
- AppArmor
  - Helps enforce mandatory access control (MAC) policies by applying profiles to processes that enable restriction of privileges they have at the level of Linux capabilities and file access.
  - Finer level of granularity compared to SELinux
- Seccomp
  - SECure COMPuting (Seccomp) is a module that can define and enforce an access control method that enables specification of the number of system calls available for an application within a container to interface with the kernel.
  - Limiting system calls provides a restricted execution environment and thus reduces the kernel attack surface.

Hardware based Solutions

- vTPM in host OS kernel
  - To make TPM available to several containers, it needs to be virtualized.
  - This functionality can be implemented by having the container runtime (or container manager) ask the host OS kernel to create a new vTPM and assign the virtual device to a container.
  - vTPMs are linked to the physical host TPM that hosts the container stack.
  - Assurance Requirements scenarios:
    - The host OS is completely trusted: assurance that containers cannot modify the host kernel by exploiting vulnerabilities in the kernel.
    - The host OS is not completely trusted, and independent trust is needed on vTPM: In case of physical TPM, the hardware platform provider signs an endorsement key (EK) stating that the TPM is trustworthy. This is extended by giving each vTPM instance its own endorsement key and signing it.



- vTPM in a Dedicated Container
  - The software-based vTPM with the same functionality described in the previous slide is built and hosted in a dedicated container (referred to as vTPM management container).
  - Main functions:
    - Access to hardware-based (physical) TPM
    - Exposes the vTPM interface to other containers through a communication channel using "adapter".
  - Assurance:
    - Same as the one provided by the host OS in the container stack.
    - The security of this implementation is jeopardized only in the event of a container escape attack.
    - Less secure then the first approach as the kernel is more reliable in limiting the kind of access it exposes to the User space.



Assurance Requirements for host OS Protection

- Generic Host OS Protection Requirements:
  - Prevent manipulation of program execution by modifying memory (e.g., buffer overflow attacks)
  - Prevent attempts to reroute code to existing procedures (e.g., system calls in common libraries)
- Requirements for Host OS Protection for Container Escape:
  - SELinux, AppArmor, and Seccomp, all of which utilize kernel loadable modules
  - A user authorized to run applications in the container should not be allowed access to the kernel-loadable modules.
  - If using Seccomp, both a syscall whitelist (a list of allowable calls) and a syscall blacklist (a list of prohibited calls) should be generated.
  - If using Seccomp, the sandboxes created by seccomp filters must not allow the use of the *ptrace* command. If *ptrace* is allowed, the tracer can modify the process's system call to bypass the filter and therefore call blocked or restricted system calls.
  - Linux Kernel Modules should have features to create a security profile for the administrators of container runtime using a combination of the above features.

Assurance Requirements for Container Runtime Configuration

- Secure Connection: TLS connection involves the encryption and authentication of both sides (container runtime module as well as the client tool used for remote administration) of the connection before establishing the TLS session.
- Isolation-based Configurations: Process, filesystem, network, IPC, User & Group level isolation.
- Resource Limiting Solutions (cgroups): should not expose container host information, local disk access, even within user namespaces and mount restricted namespaces
- Least Privilege Configuration (capabilities): Capabilities that provide the broad set of privileges almost equal to that of root should not be enabled, disable capability which allows for the loading and unloading of kernel modules.
- Container Launching Options: Containers should always be launched with a specific memory limit to prevent DoS, by specifying the number of CPU shares, only with “required” capabilities by initially dropping all capabilities and then adding only the required ones.

Assurance Requirement for Image Registry Protection

- The number of accounts accessing the registry must be limited since the common threat in some environments is account hijacking when a diverse set of clients has access to a container registry. One such environment is the registry maintained by cloud service providers who offer container services.
- The permission to create container image registries and add or remove content to registries must be cryptographically protected.

Assurance requirements for Orchestration Function

- The use of an Orchestration platform (consisting of a suite of tools) in a containerized infrastructure is intended to perform the following functions:
  - Enable the definition of a cluster
  - Enable automated deployment of containers in various clusters/hosts (container scheduling)
  - Provisioning, or defining new container hosts and attaching them to existing clusters.
- Clusters should have capabilities for logging and monitoring the resource consumption patterns of individual containers to avoid unanticipated spikes in resource usage leading to non-availability of critical resources.
- The Orchestration platform must be usable on containerized infrastructures with more than one host OS. In other words, the orchestration tools used must be container-host OS-neutral. Using different tools for different container host OS platforms increases the probability of denial-of-service attacks in those environments since the enterprise is not able to obtain a global picture of resource usage for all running containers in the entire containerized infrastructure of the enterprise.

Conclusion & References

- [1] NIST Special Publication (SP) 800-190, Application Container Security Guide, National Institute of Standards and Technology, Gaithersburg, Maryland, September 2017. https://doi.org/10.6028/NIST.SP.800-190.
- [2] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, An Updated Performance Comparison of Virtual Machines and Linux Containers, IBM Research Report, RC25482 (AUS1407-001), July 21, 2014. https://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257 D2300681E7B/$File/rc25482.pdf.
- [3] Cloud Standards Customer Council, Practical Guide to Platform-as-a-Service, Version 1.0, September 2015. http://www.cloud-council.org/deliverables/CSCC-Practical-Guideto-PaaS.pdf.

Security Assurance in Linux

Tejas Pandey

USC Viterbi
School of Engineering

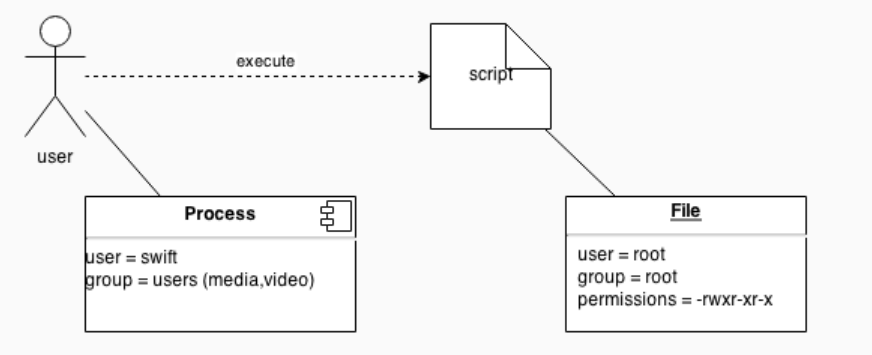
University of Southern California

Operating System Security

- Applications can be subverted.
- Security mechanisms can be circumvented.
- Secure end-to-end transactions require secure end systems
- Human the weakest link (deliberate/unintentional).

Linux Access Control

- User wants to read/write/execute file,
  - Linux DAC checks process user is running (shell).
  - Compares with that of file to make decision.



Discretionary Access Control

- Permissions identified by “OGO” (owner, group, other).
- Is existing control mechanism among all Linux flavours.
- Limited in its scope, based on identity and ownership that defines access to files, processes, sockets etc.
- Only differentiates between admin and regular user.
- Subject to admin/user’s whim; subverted by malware.
- Additional security features (ACL) still DAC in nature.

Case for SELinux

- Research project by NSA.
  - Based on learnings from prior projects (Trusted Unix, TRUSIX) towards standardizing MAC and DAC policies in Unix.
  - Demonstrate the need and implementation Mandatory access controls in Linux.
  - Initially just set of patches to Linux kernel, integrated into main branch since kernel v2.6 in August 2003.
  - Provides support for MLS, RBAC, and Type Enforcement.
  - Shipped with RHEL, CentOS, OpenSUSE, Fedora.

SELinux in a Nutshell

- Reinforce user-based DAC with MAC policy. MAC checks happen after DAC.
- Strives for separation of different security domains based on confidentiality and integrity requirements.
  - Service may listen on a port and write to syslog, but does it need to access /home?
- Services access resources through domains that limit damage by compromised applications.

Attributes of DAC and MAC

DAC	MAC
Object owner has full capability	Object owner has some capability
Complete trust in users	Trust relies only in administrators
Access decisions are based on user-id and object ownership	Objects and tasks have IDs
Impossible to control data flow	Data flow control is possible

SELinux Policy Engine

- Implements a combination of:
  - Type Enforcement
  - Role-based Access Control
  - Multi-Level Security

Type Enforcement Access Control

- Access specified between:
  - Subject type: process (invoked by user) or a domain.
  - Object type: file, dir, socket etc.
- Policy restricts subjects from inheriting access permission of caller (user).
- Four elements that define allowed access:
  - Source type: domains
  - Target type: objects to which access is allowed.
  - Object Classes: classes to which access applies.
  - Permissions: type of access allowed.

Object Classes and Permissions

- SELinux defines ~50 object classes, each with their own permissions:

Object Class	Description
blk_file	Block file
file	Regular file
sock_file	Unix sockets
dir	Regular directories
tcp_socket	TCP socket

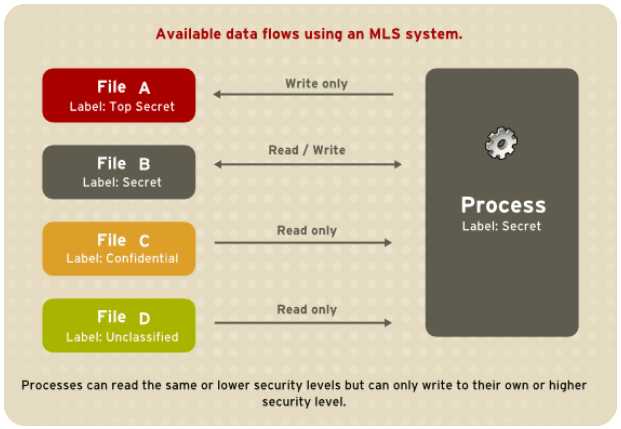
Permissions on “file” object class
create, append, rename, read, execute, write setattr etc

Role-based Access Control

- Users are assigned roles.
- Role is granted permission to access.
- SELinux maps Linux users to SELinux users.
- Further maps SELinux users to SELinux roles.
- Roles can be switched if policy allows
  - Unprivileged user is assigned role “user_r”, administrator is assigned “staff_r” for regular operations, and “sysadm_r” for system administration tasks.
  - Other roles include developer, dba etc.

Multi-Level Security

- Enforces the Bell-La Padula Mandatory Access Model (no read up, no write down).
- Defines levels of clearance and allowed data flows.



Comparison with standard Linux DAC

- Access defined by the administrator, user cannot propagate.
- Clean separation of policy from mechanism (processes run their own domains, resource access).
- RBACs enforce privilege separation.
- Lower vulnerability to privilege escalation attacks.
  - attacker access limited to process and data in a particular domain (no concept of *root*).
- Type enforcement provides integrity.
- MLS provides confidentiality.

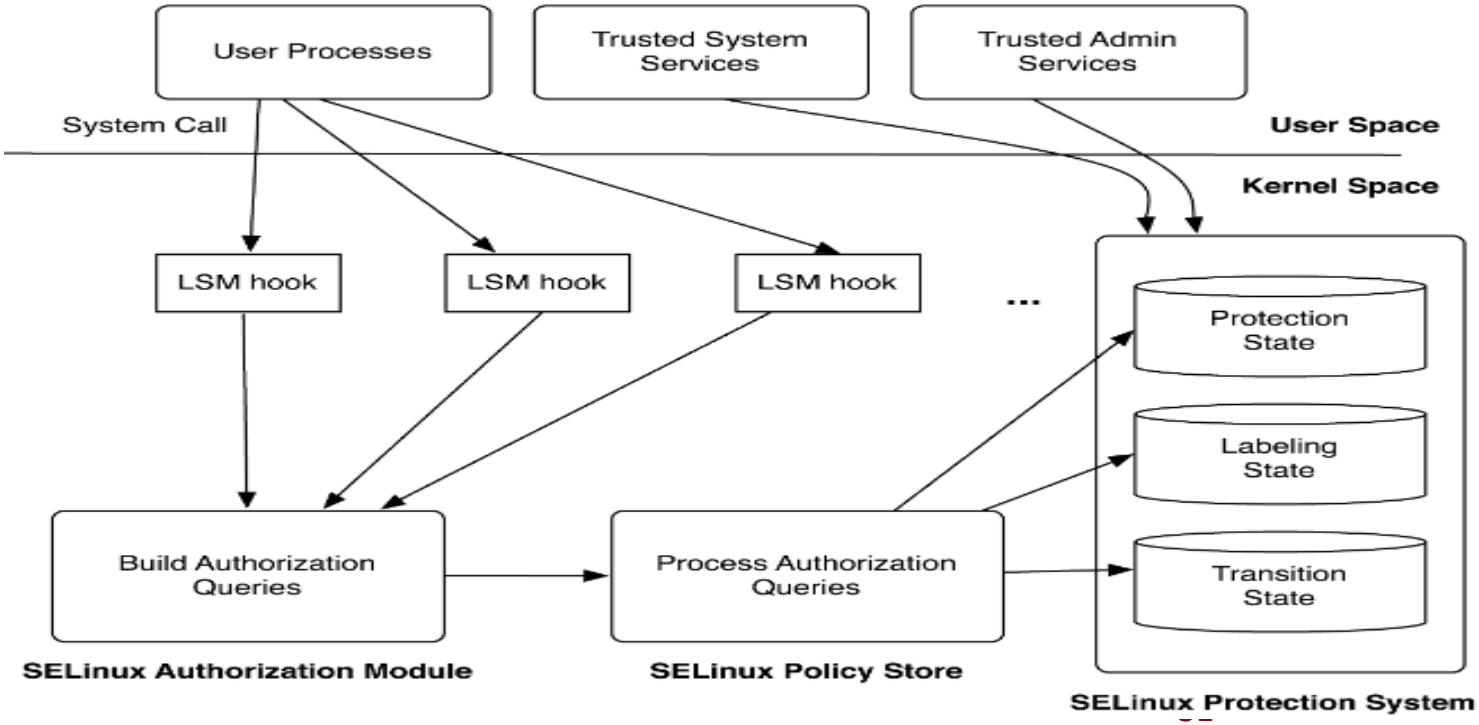
Reference Monitor

- An *authorization* system that determines whether a subject is allowed to perform an operation on an object
  - Takes as input a request
  - Returns a binary response indicating whether the request is authorized or not

SELinux as Reference Monitor

- SELinux Policies are comprised of 3 components:
  - Labeling State: defines security contexts for every file (object) and user (subject). Somewhat comparable to BLP clearances and classifications.
  - Protection State: defines the permitted <subject, object, operation> tuples.
  - Transition State: permits ways for subjects and objects to move between security contexts (move between higher to lower rings/gates)

Policy Enforcement Architecture



SELinux Security Evaluation

- Requirements
  - Mediation
  - Tamperproof
  - Verifiable

SELinux Security Evaluation

- **Mediation**

- How does the reference monitor interface ensure all security-sensitive operations are mediated without security problems (TOCTTOU)?
  - Reference monitor interface is designed to authorize access to the actual objects used by the kernel in security-sensitive operations to prevent vulnerabilities, such as TOCTTOU.

SELinux Security Evaluation

- **Mediation**

- How do we verify that the reference monitor interface provides complete mediation?
  - Source code analysis tools verify the mediation of security-sensitive kernel data structures in a consistent manner.
  - However, these tools are an approximation of the complete mediation requirements,and they are not applied on a regular basis.
  - No errors found in the RM interface since introduction in kernel v2.6

SELinux Security Evaluation

- **Tamperproof**

- How does the system protect the reference monitor from modification?
  - RM interface runs in the supervisor protection ring, as protected as the kernel.
  - Protection state configured to limit access to trusted processes (with trusted subject type labels).
  - But, does not provide input filtering at the level of Multics gates

SELinux Security Evaluation

- **Tamperproof**

- How does it protect the trusted computing base programs?
  - Evaluation of SELinux policy showed trusted processes which defined a tamper-protected, trusted computing base could be identified.
  - Requires these trusted processes must be implicitly trusted to protect themselves from low integrity inputs (cannot satisfy BIBA integrity protection)

SELinux Security Evaluation

- **Verifiability**

- What is basis for the correctness of the system's trusted computing base?
  - Verifying correctness a very complex task.
  - Large code base, written mostly in non type-safe languages, by variety of developers (verification cannot be complete in practice)

SELinux Security Evaluation

- **Verifiability**

- Does the protection system enforce the system's security goals?
  - SELinux defines a precise, mandatory specification of the allowed operations in the system (possible to build an information flow representation from the policies)
  - MLS policy ensures information flow secrecy satisfies the simple-security and *-security properties.
  - But, again based on implicit trust of subject types, enables system to be "securable"

References

- http://www.ibm.com/developerworks/library/l-secure-linux-ru/l-secure-linux-ru-pdf.pdf
- http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=5508513
- https://users.ece.cmu.edu/~jmmccune/papers/mccunej_shamon.pdf
- https://courses.engr.illinois.edu/cs423/sp2019/slides/33-reference_monitor.pdf
- http://www.cse.psu.edu/~trj1/cse544-s10/slides/cse544-lec14-selinux.pdf
- http://www.cse.psu.edu/~trj1/cse443-s12/docs/ch9.pdf
- https://ieeexplore.ieee.org/abstract/document/6060002/
- https://ieeexplore.ieee.org/abstract/document/4556572/

CHROME OPERATING SYSTEM – CASE STUDY

MALAVIKA PRABHAKAR

DSCI-523

NOV 20, 2020

TABLE OF CONTENTS

- What is Chrome OS?
- Security Requirements
- Threat Model
  - Mitigating Remote System Compromise
    - OS Hardening
    - Verified Boot
  - Mitigating Device Theft
- Layers of Chrome OS Security
  - Auto-Update
  - Security Boundaries in Chrome OS
  - vs Security Boundaries in Windows
- Trusted Platform Module (TPM)
  - Titan C. - The Nucleus of Trust
- Minimization (Chrome OS vs Other Operating Systems)
- Assurance Techniques Applied to Chrome OS
- How to Improve the Security of the System
- Consequences of Security Failure
- References

WHAT IS CHROME OS?

- Chrome OS is a Gentoo Linux–based operating system designed by Google.
- It is derived from the open-source project Chromium OS and uses the Google Chrome web browser as its principal user interface.
- It aims to provide a fast, simple, and more secure computing experience for people who spend most of their time on the web.
- Chrome OS is available primarily on Chromebooks.

SECURITY REQUIREMENTS

- The owner should be able to delegate login rights to users of their choice.
- The user can manage their risk with respect to data loss, even in the face of device loss or theft.
- A user's data can't be exposed due to the mistakes of other users on the system.
- The system provides a multi-tiered defense against malicious websites and other network-based attackers.
- Recovering from an attack that replaces or modifies system binaries should be as simple as rebooting.
- In the event of a security bug, once an update is pushed, the user can reboot and be safe.

THREAT MODEL

Remote System Compromise

- Attack vectors through which an adversary might try to compromise a Chrome OS device remotely include:
  - an exploit that gives them control of browser processes
  - an exploit in a plugin
  - tricking the user into giving a malicious web app unwarranted access
  - trying to subvert the auto-update process in order to get some malicious code

Device Theft

- The challenges involved here are:
  - Wanting to protect user data while also enabling users to opt-in to auto-login.
  - Wanting to protect user data while also allowing users to share the device.
  - Wanting to protect user credentials without giving up offline login, auto-login, and device sharing.
  - Wanting to provide disk encryption with only minimal impact on battery life and performance speed.
  - The attacker can remove the hard drive to circumvent OS-level protections.
  - The attacker can boot the device from a USB device.

MITIGATING REMOTE SYSTEM COMPROMISE

- **OS Hardening**
- This limits the attack surface, reduces the likelihood of successful attack, and reduces the usefulness of successful user-level exploits. Techniques include:
  - Process sandboxing
    - Mandatory access control implementation that limits resource, process, and kernel interactions
    - Chrooting and process name-spacing for reducing resource and cross-process attack surfaces
    - Media device interposition to reduce direct kernel interface access from Chrome browser and plugin processes
  - Kernel hardening
  - Additional file system restrictions
    - Read-only root partition
    - User home directories that can't have executable files

MITIGATING REMOTE SYSTEM COMPROMISE

- **Verified Boot**
- Verified boot provides a means of getting cryptographic assurances that the Linux kernel, non-volatile system memory, and the partition table are untampered with when the system starts up. The design is broken down into two stages:
  - **Firmware-based verification**
  - **Kernel-based verification**
- When combined, the two verification systems will perform as follows:
  - Detects changes at boot-time
    - Files, or read-write firmware, changed by an opportunistic attacker with a bootable USB drive will be caught on reboot.
    - Changes performed by a successful runtime attack will also be detected on next reboot.
  - Provides a secure recovery path so that new installs are safe from past attacks.

MITIGATING DEVICE THEFT

Data Protection:

Key requirements for protecting cached user data (at rest) are as follows:

- Each user has their own encrypted store.
- All user data stored by the operating system, browser, and any plugins are encrypted.
- Users cannot access each other's data on a

CAPTCHAs

- Google Accounts APIs respond with CAPTCHAs if the servers believe an attack is underway.
- If the user has correctly provided their credentials, they can be successfully logged in without needing to face CAPTCHAs.

Account Management

- **"The owner"**
  - Each device has one and only one owner.
  - *User preferences* (such as setting a time zone) are distinct from *system settings* (such as Wi-Fi).
- **Whitelisting**
  - The owner can whitelist other users, who can then log in.
- **Promiscuous mode**
  - The owner can opt into a mode where anyone with a Google account can log in.

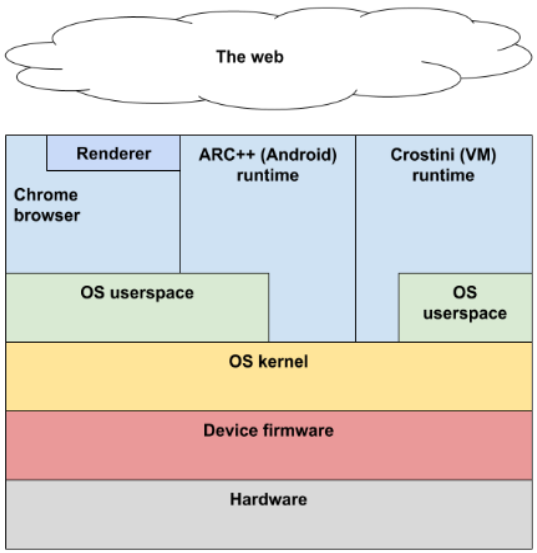
Single Sign-On

- Upon login, the system decrypts the user's profile and performs a request for their Google Accounts cookies in the background. As a result, their profile gets fresh cookies, and they are logged in to all their Google services.

e a completely which does not sync

- All system settings would be kept out of this session, including networking config.

LAYERS OF CHROME OS SECURITY



- This is a high-level view of the Chrome OS software stack.
- It depicts each of the layers of defenses Chrome OS deploys from the user's primary point of interaction (a browser accessing the web, an Android app, or a VM) to the device hardware.
- The principle of ***deploying defenses in depth*** is paramount here, as the layers of defense combine to provide a system more robust in its protection against threats.

LAYERS OF CHROME OS SECURITY

Security and the Web

- Accessing a web page via a browser amounts to executing untrusted code on a user's device.
- Solution: ***principle of least privilege***
  - This code is being executed by a system (the web browser) which can take many steps to limit the privileges that untrusted code may hold.

Security in the Chrome Browser

- The Chrome browser implements a *multi-process* architecture.
- Web pages are loaded and executing inside a heavily-restricted runtime environment using Linux-based ***sandboxing***. Web pages loaded on Chrome OS have no access to the device's filesystem, nor to user files.

Protecting User Data

- Chrome OS devices are intended to be both portable and safely shared, and so protection for user data stored on the local disk is required.
- This is accomplished via ***system-level encryption*** of all the user's data. This also includes the entire Chrome browser profile so there is no risk of browser data leaking outside of encrypted storage.
- In a nutshell, each user gets a unique "vault" directory and keyset that is created at their first login. The vault is used as the underlying encrypted storage for their data. The keyset is tied to their login credentials and is required to allow the system to both retrieve and store information in the vault. The vault is opened transparently to the user at login. On logout or reboot, the user's data is locked away again.
- Chrome OS devices use a Trusted Platform Module (TPM) chip to protect against brute-force attempts to recover a user's keyset (and therefore the data it protects), and against attempts to directly extract the keys from the hardware.

Android Applications and Chrome OS

- Many Chrome OS devices, including tablets, support running Android applications.
- Android applications are executed inside a Linux container which restricts interactions with the rest of the system and access to user data.
- Moreover, Android applications are confined to the currently logged-in user session, which means that Android applications cannot be used to access information belonging to other users, nor to attack other user accounts.
- For devices managed by an organization, the ability to run Android applications can

Linux Applications and Chrome OS

- Chrome OS devices with sufficient storage and computing power may support running user-installed Linux applications.
- Linux applications are executed in a container running inside a hardware-based virtual machine.
- This provides an effective security barrier between Linux applications and the Chrome OS system, without sharing the host Chrome OS kernel with the virtual machine guest.
- This functionality provides a Linux environment, in effect a Linux “sandbox”, that is well isolated from the rest of the system and should allow executing untrusted workloads without exposing the main Chrome OS environment to these workloads.

Hardware Root-of-Trust and Verified Boot

- Chrome OS enforces a *hardware* root-of-trust for the software running on the device, meaning that its integrity is ensured by Google.
- This assurance is tied to the hardware on the device and cannot be subverted by purely software means. This is done using the ***Verified boot***.
- An attacker needs to be physically present and have control of the device, and moreover needs to perform a hardware change to the device to be able to run untrusted code or

Root versus the OS Kernel

- Chrome OS enforces a clear separation between the *root* or supervisor user in the underlying Linux system, and the operating system *kernel* (*principle of least privilege*)
- On Chrome OS, kernel module loading is restricted to modules loaded from the root partition covered by Verified boot. This means that while the root user can modify the kernel at runtime by loading kernel modules, these modules are still trusted code. Module loading from outside the verified root partition is disallowed. Chrome OS also restricts kernel functionality exposed to processes running as root using seccomp.
- By enforcing the kernel/root barrier, we ensure that an attacker that attempts to compromise the kernel will need to exploit a kernel privilege escalation bug in addition to all the other bugs that they needed to exploit to obtain root code execution, making a kernel compromise less likely.

AUTO-UPDATE

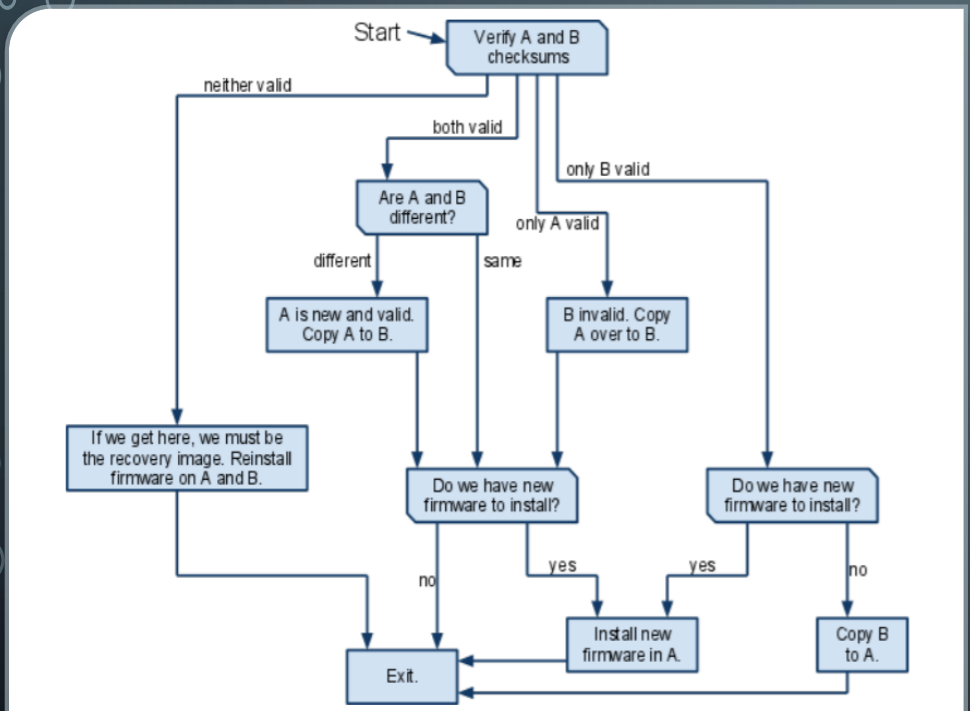


Figure 2. Firmware update checks firmware A and B. Firmware A is the primary firmware and will take new updates while B acts as a backup copy.

- Chrome OS can effectively push out security updates and minimize the window of vulnerability rather than waiting for user interaction.
- The firmware and software update mechanisms are very similar. Once updated, verified boot can detect whether the update was successful or not.

SECURITY BOUNDARIES IN CHROME OS

- Chrome OS implements the principle of *defense in depth*, where we deploy more than one layer of defense between untrusted content or code and sensitive data or device access. These layers define security boundaries.

Chrome renderer process to Chrome browser process:
this boundary is traversed only with Chrome IPC or Mojo. This is enforced by the Chrome sandbox.

Chrome browser process to system services:
this boundary is traversed with D-Bus or Mojo. This is enforced by UID separation.

ARC++ container to Chrome browser or Chrome OS system:
this boundary is traversed with Mojo IPC. Trust decisions should always be made outside of the container.

User-space processes to kernel: this boundary is traversed by system calls. Seccomp is used to secure this boundary.

Kernel to firmware:
it should not be possible for a kernel compromise to result in persistent, undetectable firmware manipulation. This is enforced by Verified boot.

VS SECURITY BOUNDARIES IN WINDOWS

Security Boundary	Security Goal
Network boundary	An unauthorized network endpoint cannot access or tamper with the code and data on a customer's device.
Kernel boundary	A non-administrative user mode process cannot access or tamper with kernel code and data. Administrator-to-kernel is not a security boundary.
Process boundary	An unauthorized user mode process cannot access or tamper with the code and data of another process.
AppContainer sandbox boundary	An AppContainer-based sandbox process cannot access or tamper with code and data outside of the sandbox based on the container capabilities
User boundary	A user cannot access or tamper with the code and data of another user without being authorized.
Session boundary	A user logon session cannot access or tamper with another user logon session without being authorized.
Web browser boundary	An unauthorized website cannot violate the same-origin policy, nor can it access or tamper with the native code and data of the Microsoft Edge web browser sandbox.
Virtual machine boundary	An unauthorized Hyper-V guest virtual machine cannot access or tamper with the code and data of another guest virtual machine; this includes Hyper-V Isolated Containers.
Virtual Secure Mode boundary	Data and code within a VSM trustlet or enclave cannot be accessed or tampered with by code executing outside of the VSM trustlet or enclave.

- Some boundaries are similar to those in Chrome OS including kernel boundary, AppContainer sandbox boundary, web browser boundary, etc.
- Boundaries unique to Chrome OS:
  - **Chrome renderer process to Chrome browser process**
  - **Kernel to firmware**
- Boundaries unique to Windows:
  - **User boundary**
  - **Session boundary**

TRUSTED PLATFORM MODULE

- **Trusted Platform Module** is an international standard for a secure crypto-processor, a dedicated microcontroller designed to secure hardware through integrated cryptographic keys.
- Chrome OS uses the TPM for these tasks:
  - Preventing software and firmware version rollback
  - Maintaining information to detect transitions between normal and developer modes
  - Protecting user data encryption keys
  - Protecting certain user RSA keys ('hardware-backed' certificates)
  - Providing tamper evidence for installation attributes
  - Protecting stateful partition encryption keys
  - Attesting TPM-protected keys
  - Attesting device mode
- The TPM is not directly available outside of Chrome OS for any purpose; that is, no remote computer has access to the TPM.

TITAN C. - THE NUCLEUS OF TRUST

- Titan C is the Google-designed security chip on Chromebooks. It defends from the core to keep devices secure, protect user identity, and ensure system integrity.

Continuous Security

- ***Designed by Google***
  - Google designs and monitors the manufacturing process to ensure quality.
- ***Updated by Google***
  - All firmware updates to these chips are pushed out by Google.
- ***Standard on Chromebooks***
  - Titan C chips are built into Chromebooks, are always on and require no configuration to enable.

System Integrity

- ***Protection from malicious tampering of OS & firmware***
  - Titan C assists with the Verified Boot process, which prevents malicious code from modifying Chrome OS.
- ***Protection from enterprise policy non-compliance***
  - Titan C helps ensure that many policies set with Chrome Enterprise, like the ability to prevent users from putting their device into developer mode, are enforced on managed Chromebooks.

User Protection

- ***Protection from login attempts on remote hardware***
  - Titan C guards access to user data encryption keys.
- ***Protection from brute force password attempts***
- ***Protection from Phishing Attacks***
  - Titan C enables authentication methods, like two-factor authentication with the power button, that would require a power button press in addition to the password to log into the device.

MINIMIZATION (CHROME OS VS OTHER OPERATING SYSTEMS)

Disadvantages	Advantages
Chrome OS is a light-weight operating system, meaning that there are less features offered compared to other operating systems.	Less features means that it has a smaller attack surface compared to other OS.
Since the hardware specs are limited, tasks like gaming and video editing are difficult.	Chromebooks are designed to minimize hardware processing and memory requirements by offloading many traditional functions to the cloud. They do not require bigger batteries or generate excessive heat, and they are free of the "bloatware" that traditional laptops have, enabling faster booting and overall performance.
Local disk capacity (storage) is limited.	
Lack of support from third-party applications.	Many Chrome OS devices support running Android applications, installable from the Google Play Store and covered by Google Play Protect. Some devices also support running user-installed Linux applications.
Lack of customization.	Making the device secure by default

ASSURANCE TECHNIQUES APPLIED TO CHROME OS

- Minimization
- System Integrity
- Trusted Platform Module
- Verified Boot
- Encryption
- Sandboxing
- Kernel Hardening
- Auto-Update
- Kernel Name Spacing
- Secure Computing Mode
- Kernel / Root Barrier
- UID Separation
- Defense in Depth
- Principle of Least Privilege

HOW TO IMPROVE THE SECURITY OF THE SYSTEM

- Boot More Often
  - By design, verified boot and auto-update only runs on boot. However, if the user does not reboot the system very often, then these features cannot be utilized.
  - An alternative would be for Chromebook to silently reboot whenever the user locks the screen or puts the machine to sleep for some amount of time (e.g. 5 minutes). In this situation, the user probably forgot to log out or is otherwise unlikely to return within the next 8-10 seconds, and a silent reboot would go unnoticed.

CONSEQUENCES OF SECURITY FAILURE

- Manipulation of data
- Loss of data
- Disclosure of personal information / breach of privacy
  - Could lead to identity theft, credit card fraud, coercion, etc.
- Denial of service
- Elevation of privilege resulting in future exploits

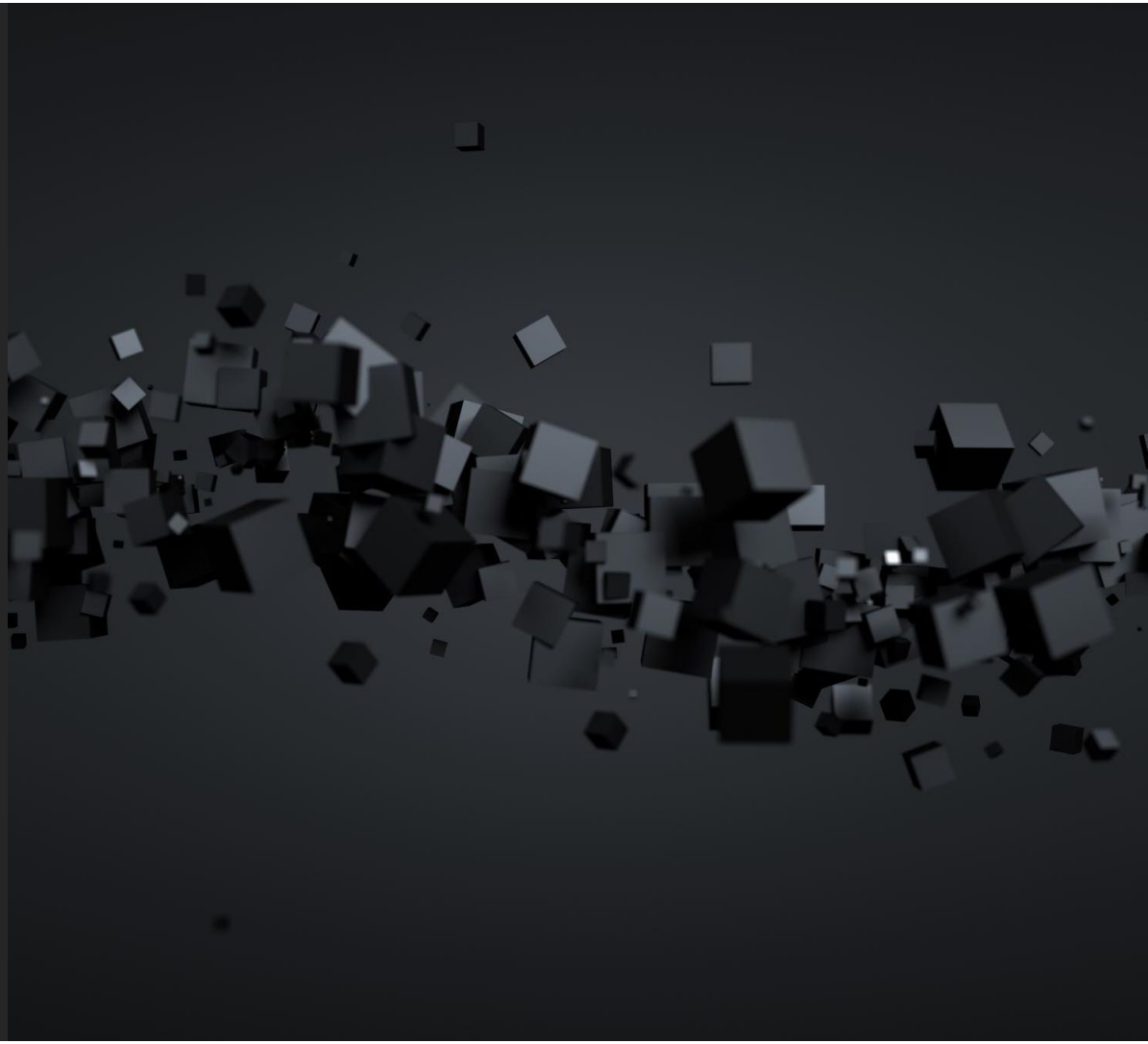
REFERENCES

- https://www.chromium.org/chromium-os/chromiumos-design-docs/security-overview
- https://chromium.googlesource.com/chromiumos/docs/+/master/security/chromeos_security_whitepaper.md
- http://www.chromium.org/developers/design-documents/tpm-usage
- https://www.google.com/chromebook/chrome-os/#secure
- https://chromeenterprise.google/os/security/
- https://storage.googleapis.com/chrome-enterprise/pdf/Titan_C_The_Nucleus_of_Trust_one_pager.pdf
- https://en.wikipedia.org/wiki/Trusted_Platform_Module
- http://dhanus.mit.edu/docs/ChromeOSSecurity.pdf
- https://chromium.googlesource.com/chromiumos/docs/+/refs/heads/factory-grunt-11164.B/security_severity_guidelines.md
- https://support.google.com/chromebook/answer/3438631?hl=en
- https://www.forbes.com/sites/kevinmurnane/2019/04/21/a-chromebooks-superb-security-is-another-good-reason-to-leave-windows-10s-update-failures-behind/#138fbd3a9a97

THANK YOU

The Assurance of Voting Systems

By Anthony Cassar

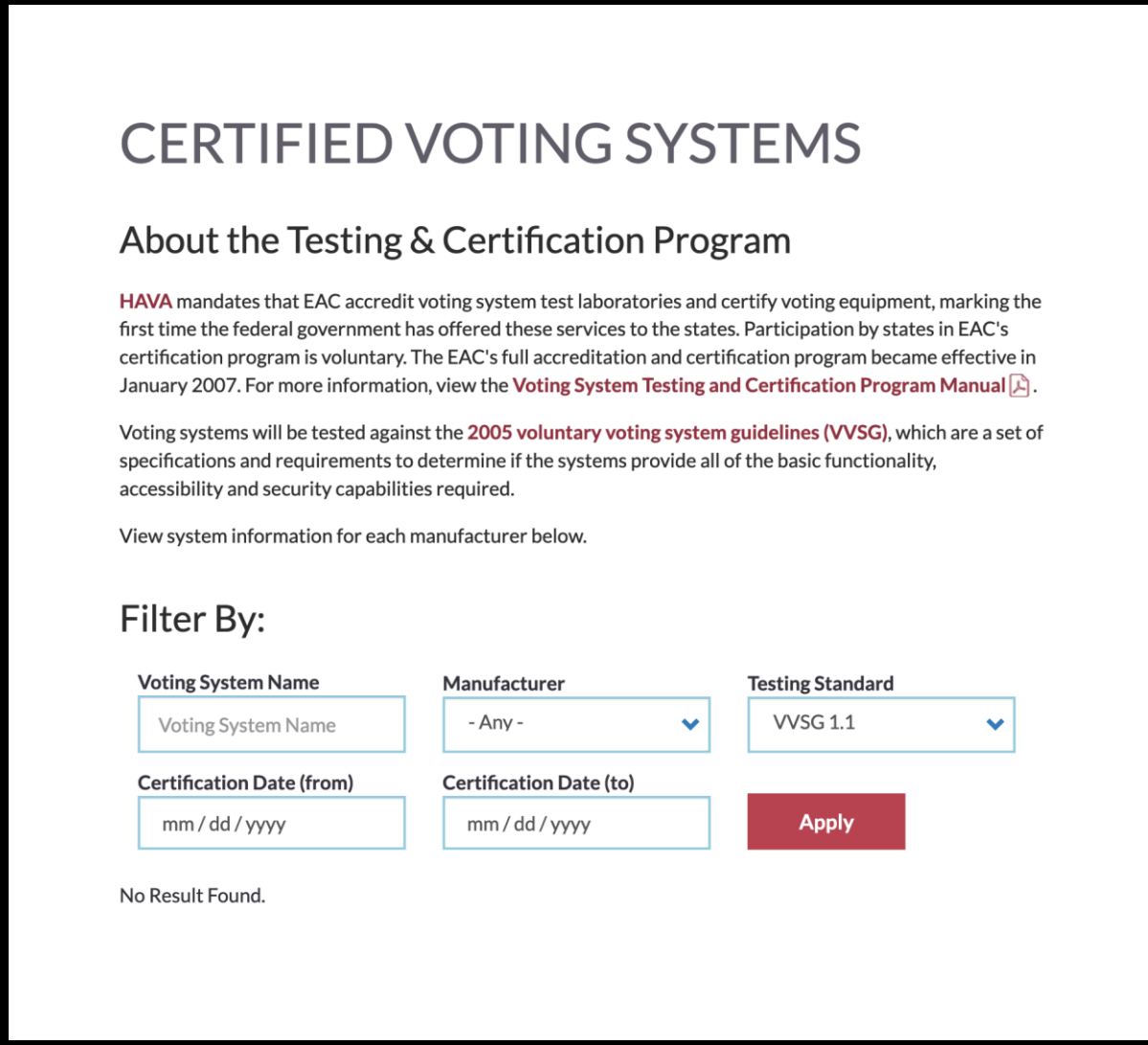


What is Assurance for Voting Machines?

- Assurance: Implementing security requirements that enforce mechanism which fulfil policy. Certifying the machines by testing the all hardware components , software, firmware within the machine.
- Case Study: Democracy Suite System(Dominion)

Trusted Build Concept

- The Certification Process,
- 2005 Requirements: In Use
- VVSG 1.1 2015 Requirements



CERTIFIED VOTING SYSTEMS

About the Testing & Certification Program

HAVA mandates that EAC accredit voting system test laboratories and certify voting equipment, marking the first time the federal government has offered these services to the states. Participation by states in EAC's certification program is voluntary. The EAC's full accreditation and certification program became effective in January 2007. For more information, view the Voting System Testing and Certification Program Manual.

Voting systems will be tested against the 2005 voluntary voting system guidelines (VVSG), which are a set of specifications and requirements to determine if the systems provide all of the basic functionality, accessibility and security capabilities required.

View system information for each manufacturer below.

Filter By:

Voting System Name: Voting System Name

Manufacturer: - Any -

Testing Standard: VVSG 1.1

Certification Date (from): mm / dd / yyyy

Certification Date (to): mm / dd / yyyy

Apply

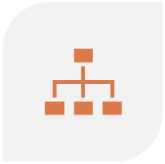
No Result Found.

Initial Issues with Assessment:

- Current Documentation: VVSG
- Open-Source?
- Testing
- (EAC) Elections Assistance Commission

The Flaws of the EAC?

SECURITY REQUIREMENTS ARE "SHALL OR SHALL NOT DO THIS"



ORGANIZATIONAL PROBLEMS:

POLICY>MECHANISM>ENFORCEMENT>TEST

PROOF OF TEST, ISSUE THAT AROSE, CHANGES MADE

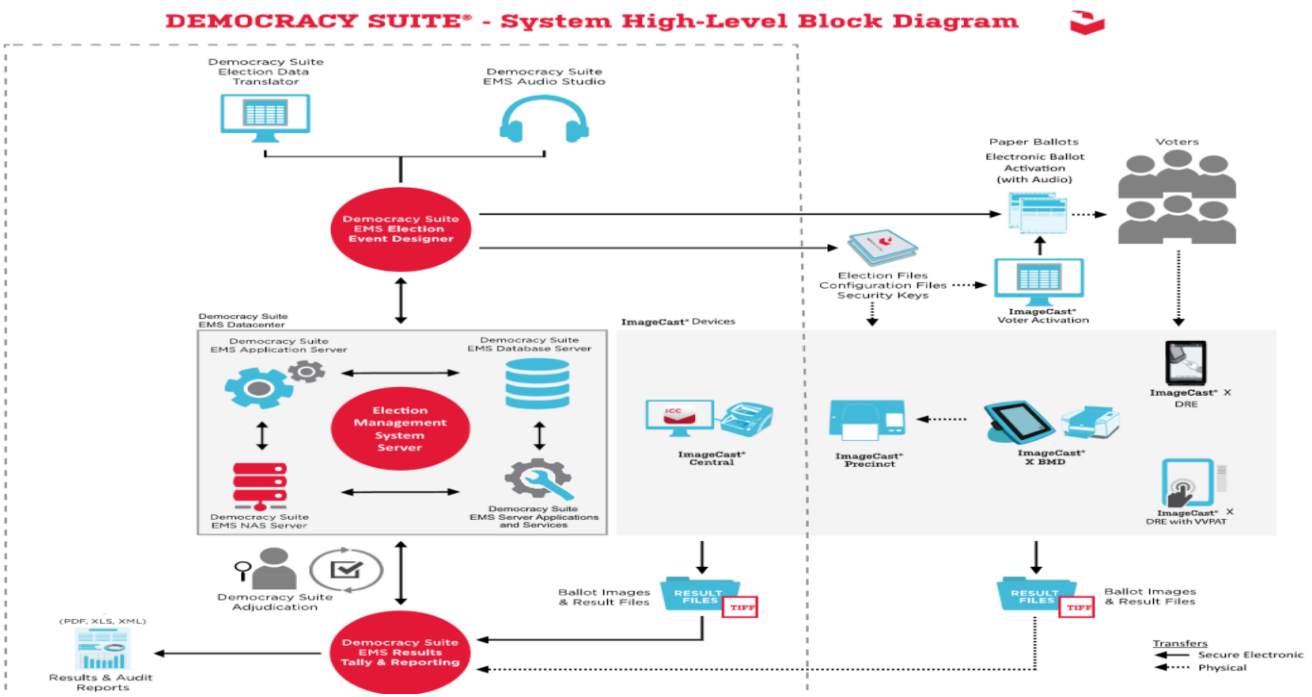
EXAMPLE: AUTHENTICATION

USE OF MAGNETIC CARD:

CAN IT BE COPIED EASILY?

WHERE IS IT MADE?

The Layout:


DEMOCRACY SUITE® - System High-Level Block Diagram
Democracy Suite Election Data Translator
Democracy Suite EMS Audio Studio
Democracy Suite EMS Election Event Designer
Paper Ballots
Electronic Ballot Activation (with Audio)
Voters
Election Files Configuration Files Security Keys
ImageCast® Voter Activation
Democracy Suite EMS Datacenter
Democracy Suite EMS Application Server
Democracy Suite EMS Database Server
Election Management System Server
Democracy Suite EMS NAS Server
Democracy Suite EMS Server Applications and Services
ImageCast® Devices
ImageCast® Central
ImageCast® Precinct
ImageCast® X BMD
ImageCast® X DRE
ImageCast® X DRE with VVPAT
Democracy Suite Adjudication
Ballot Images & Result Files
RESULT FILES
TIFF
Ballot Images & Result Files
(PDF, XLS, XML)
Results & Audit Reports
Democracy Suite EMS Results Tally & Reporting
Transfers
Secure Electronic
Physical

What is the Operating System?

- Device: ImageCast X
- Android OS version 8.1
- Isolating Root Process:
- Sandboxing

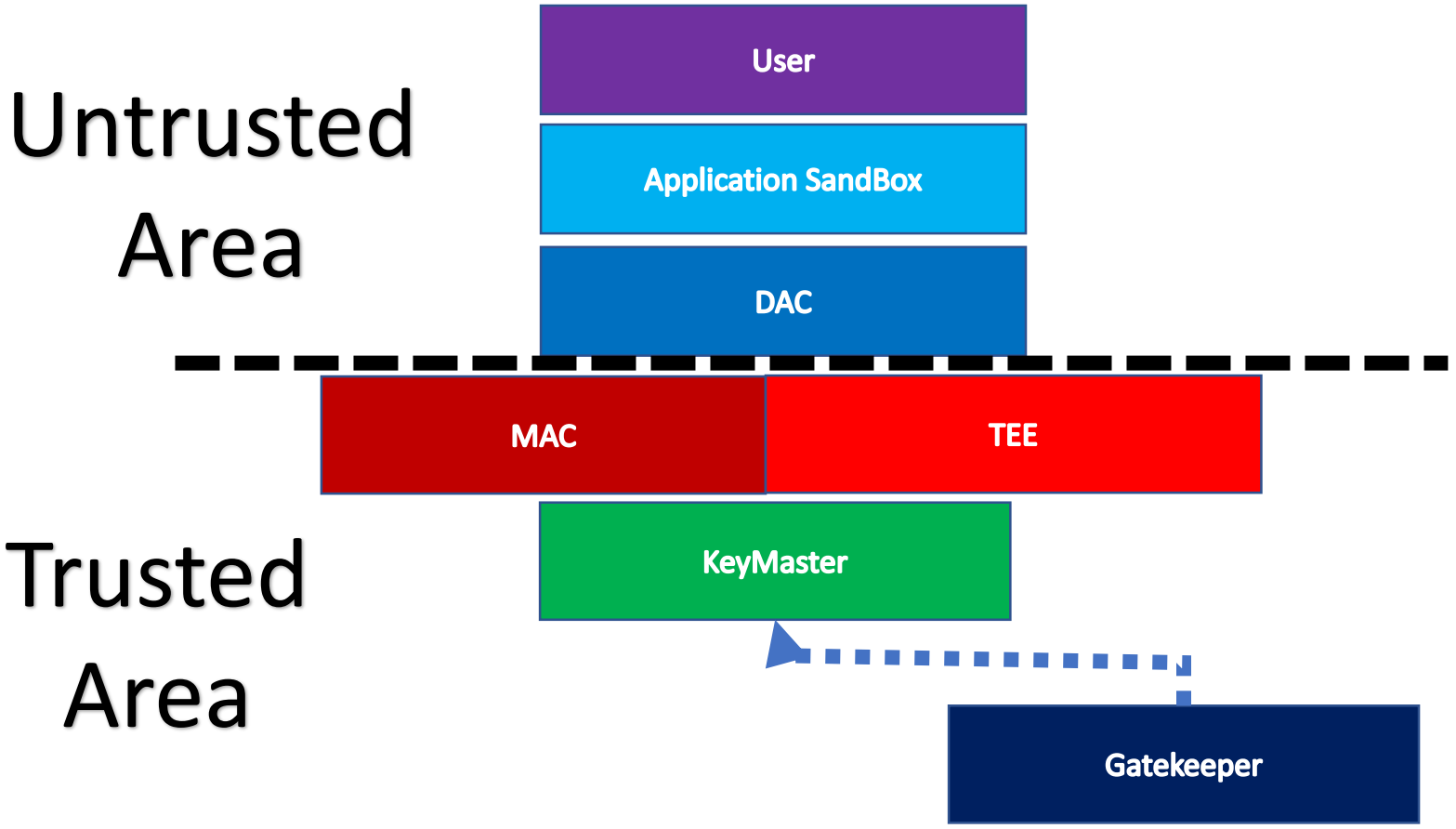
Existing Hardware:

Samsung Galaxy Tab Pro 12.2

HP LaserJet Pro m203dw

Smart Card Reader

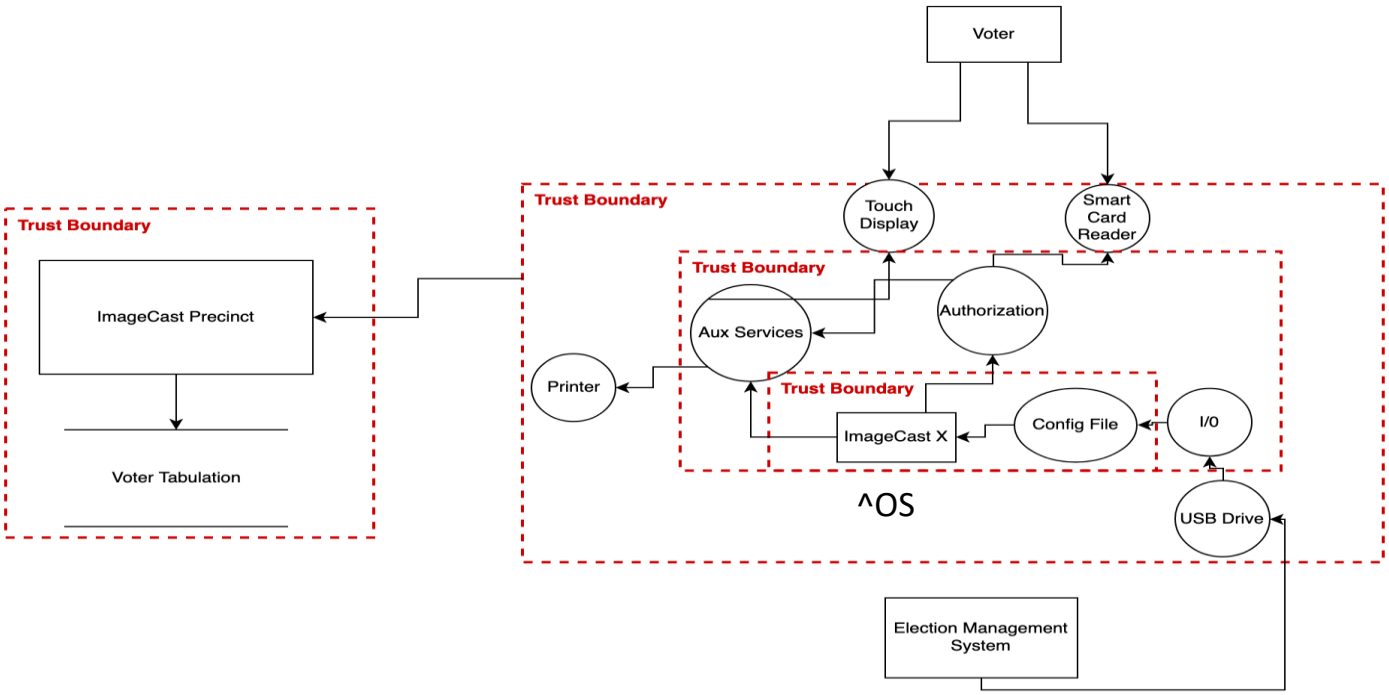
The Android OS 8.1 TCB:



Security Kernel Features:

- Separates User Processes
- Limits Resource Use
- *Enforcing* mode

Data Flow Diagram:



Software Can Effect the Security of the Device

- CVE-2020-0409
- Bug with File mapping:
- Fail explicitly on length overflow. Instead of aborting when FileMap::create detects an overflow, detect the overflow directly and fail the call. Bug: 156997193
- Bypass User Interaction Requirements

Strong Statements

- “Comprehensive, non-alterable audit logs”
- Fake NEWS!
- Nothing In Security Is 100 Percent Successful.
- Overwrite Data(File Mapping)

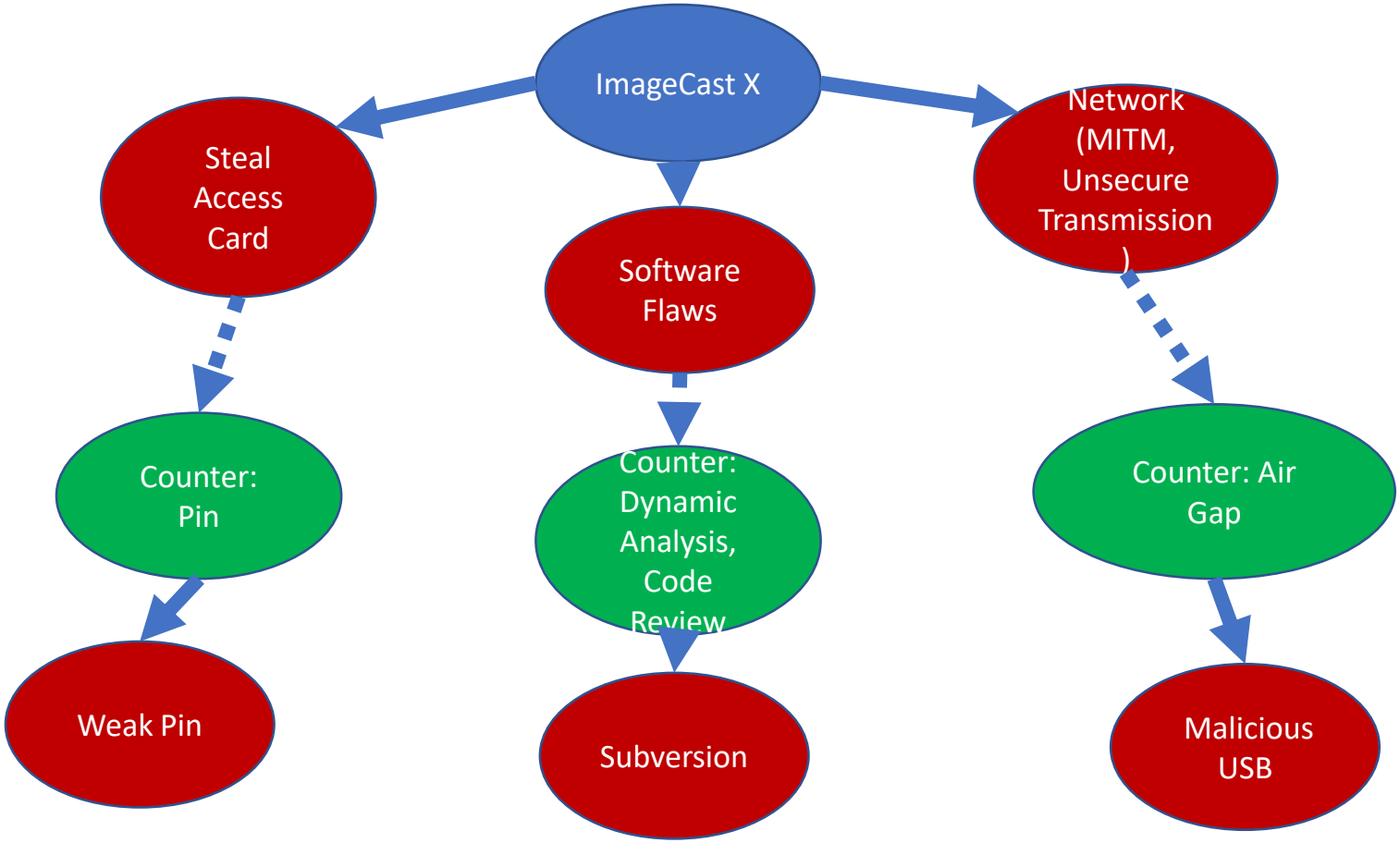
Election Management System (EMS) Front End and Back-End

- EMS Adjudication Service:Represents a server side application which provides ballot information such as contests, candidates and their coordinates from EMS to the Adjudication application.
- EMS Application Server:Represents a server side application responsible for executing long running processes, such as rendering ballots, generating audio files and election files, etc.
- EMS Database Server:Represents a server side RDBMS repository of the election project database which holds all the election project data, including pre-voting and post-voting data.
- EMS Data Center Manager:A server application that represents a system level configuration application used in EMS back-end data center configuration.
- EMS Election Device Manager:Application used for production and programming of election files, and other accompanying files, for ImageCast X terminals.
- EMS File System Service:A back-endapplication that acts as a stand-alone service that runs on client machines, enabling access to low level operating system API for partitioning CF cards, reading raw partition on ICP CF card, etc.
- EMS NAS Server:Represents a server side file repository of the election project file based artifacts, such as ballots, audio files, reports, log files, election files, etc.

Continued:

- EMS Adjudication:Represents the client component responsible for adjudication, including reporting and generation of adjudicated result files from ImageCast Central tabulators and adjudication of write-in selections from ImageCastPrecinct and ImageCast TR-01-01-DVS-2017-02.01Rev. B2Central tabulators. This client component is installed on both the server and the client machines.
- EMS Audio Studio:A client application that represents an end-user helper application used to record audio files for a given election project. As such, it is utilized during the pre-voting phase of the election cycle.
- EMS Election Data Translator:End-user application used to export election data from election project and import election data into election project.
- EMS Election EventDesigner:A client application that integrates election definition functionality together with ballot styling capabilities and represents a main pre-voting phase end-user application
- ImageCast Voter Activation:An application, installed on a workstationor laptop at the polling place, which allows the poll workers to program smart cards for voters. The smart cards are used to activate voting sessions on ImageCast X.
- EMS Results Tally and Reporting:A client application that integrates election results acquisition, validation, tabulation, reporting, and publishing capabilities and represents the main post-voting phase end-user application

Attack-Defense Tree



Password Policy

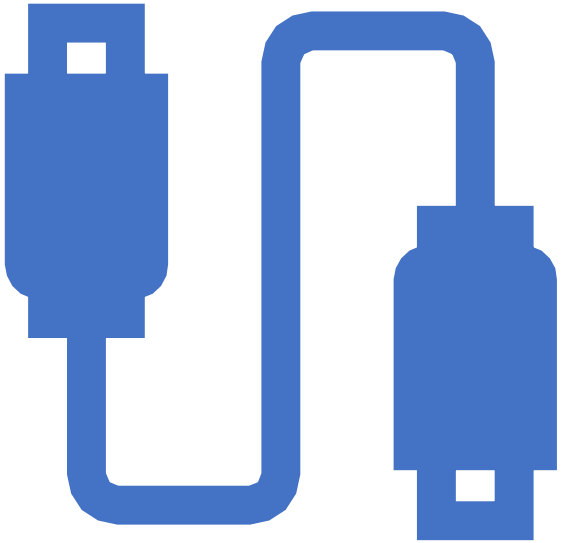
- BIOS Login
- OS Login
- Short Password, 8 Minimum Charecters, Some Complexity
- Some systems are required to have the same credentials?

The Systems and The Internet

- Transmission Data SSL or SFTP
- MITM Attacks
- Upgrade or Remove Internet Capability
- Airgap?

USB Management?

- Images begin with USB inserting to workstation
- Scanning with Windows Defender?
- On the cleaner station?



Integrity of the Votes

The Changing of Votes

Notification

Hashing

Source Code

- Unavailable Currently
- Unauthorized Modifications of OS
- Open-Source Vs Closed Source

Any Questions?

References:

- https://www.sos.state.co.us/pubs/elections/VotingSystems/DVS-DemocracySuite/documentation/2-02-DemocracySuiteSystemOverview.pdf
- https://www.sos.texas.gov/elections/forms/sysexam/jan2019-sneeringer.pdf
- https://www.sos.state.co.us/pubs/elections/VotingSystems/DVS-DemocracySuite511/documentation/UG-ICX-UserGuide-5-11-CO.pdf
- https://www.thecentersquare.com/national/officials-raised-concerns-for-years-about-security-of-u-s-voting-machines-software-systems/article_bec0fc86-2144-11eb-bc8c-bb85a60db758.html
- https://www.eac.gov/sites/default/files/voting_system/files/Dominion_Voting_Systems_D-Suite_5.5_Test_Report_Rev_B.pdf
- https://www.researchgate.net/publication/224204822_An_Android_Application_Sandbox_system_for_suspicious_software_detection
- https://arxiv.org/pdf/1904.05572.pdf
- https://source.android.com/security/
- https://votingsystems.cdn.sos.ca.gov/vendors/dominion/ds510-use-proc-jan.pdf
- https://votingsystems.cdn.sos.ca.gov/vendors/dominion/ds510-use-proc-jan.pdf
- https://sos.ga.gov/admin/files/Dominion%20RFI_No%20Redactions.pdf
- https://www.youtube.com/watch?v=v62C-riEZ0I

Internet of Vehicles: Trusted Communication

CHRIS SAMAYOA

NOVEMBER 20, 2020

Outline

- Mesh Networks
- Trusted Communication Concerns
- ECU Review
- TCB Considerations
- Method 1: TCM
- Method 2: Multi-Purpose ECUs (VMM)
- Method 3: KPS-Based Attestation
- Suggested Improvements
- STRIDE Threat Model
- Questions

Mesh Networks (Zhang, 2007)

Nodes are both terminals and routers

Trust needed for nodes to be able to distribute data

Uses for mesh networks with autonomous vehicles:

Bandwidth	Road Conditions	Obstacles	Weather
Vehicle Malfunction	Route Updates	Telemetry	Infrastructure Communication

- High bandwidth needed to share data from video, radar, Lidar between CAVs (Connected and Autonomous Vehicles)

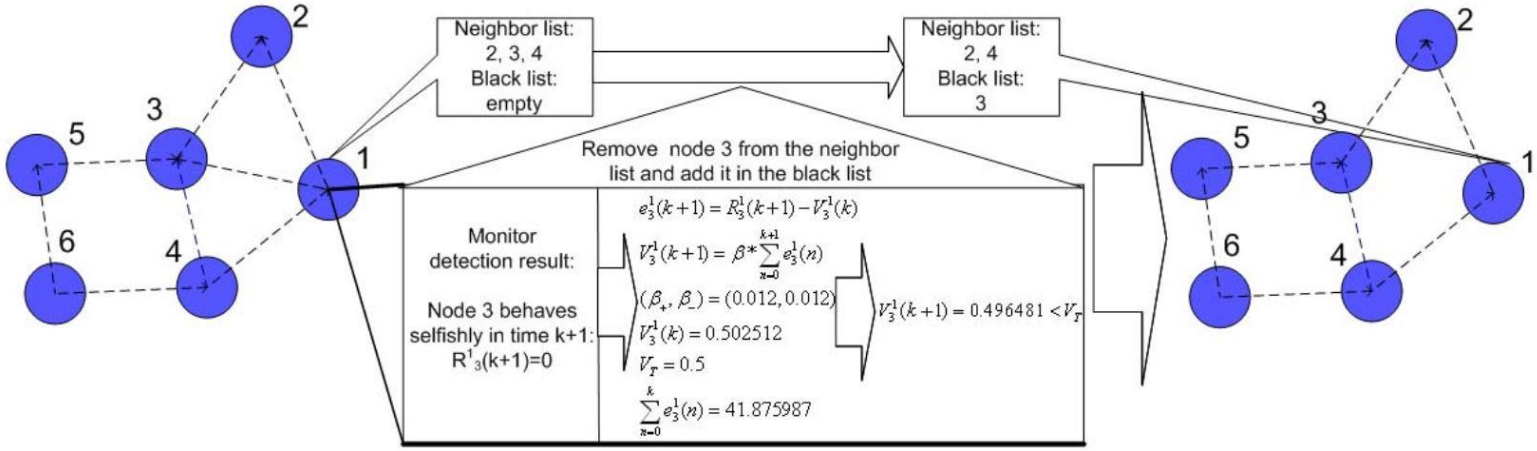
5.9 Ghz range reserved for CAVs using dedicated short-range communications (DSRC) for vehicles to communicate with other vehicles and infrastructure (Lindeman, 2017)

- Can also be used for auto safety; namely C-V2X (Cellular-Vehicle-to-Everything) technology

Mesh Networks (Zhang, 2007)

How to establish trust?

- Prevention based mechanism to avoid selfish behavior
  - ❖ Requires perfect mechanism in place
- Detection and reaction mechanism
  - ❖ Can use PID (proportional-integral-derivative) controller to automate process via punishment based reputation system



(Zhang, 2007)

Common Communication Concerns

Unauthenticated Communications

- Vehicle to Vehicle
- Vehicle to Infrastructure
- Intra-vehicle
- Cv2x (Cellular Vehicle-to-Everything) / v2x (Vehicle-to-Everything)
  - ❖ Other types of external communication carry similar concerns to mesh networks (UMTS, WiMAX, DSRC)

Compromised System Components

- ECUs (Electronic Control Unit)
- CAN (Controller Area Network)
- Internet Connected Infotainment System

Secure Software Distribution

Creating Trusted Relationships

Loss of Connectivity

Summary of ECU Topics (Oguma, 2008)

ECU Types

- Sensor ECUs: Collect information about the vehicle and its surroundings
- Management ECUs: Interpret data from sensor ECUs and instruct actuator ECUs to maintain vehicle systems
- Actuator ECUs: Control vehicle components based on instructions received
  - e.g. Adaptive Cruise Control

ECU Considerations

- 30-300 ECUs currently exist in modern vehicles
- Communicate via CAN
- Require low-latency communications to operate effectively
- Must be tamperproof to work within a trusted communication design

TCB Considerations

Design Factors

- Attestation (local versus remote)
- Minimization
- Layering

Real-World Issues

- Computational Speed
- Strength of Encryption Algorithms (longevity)
- Consumer Usability
- External Communication Issues
  - e.g. rural areas or Denial-of-Service attacks

Method 1: TCM (Wang, 2010)

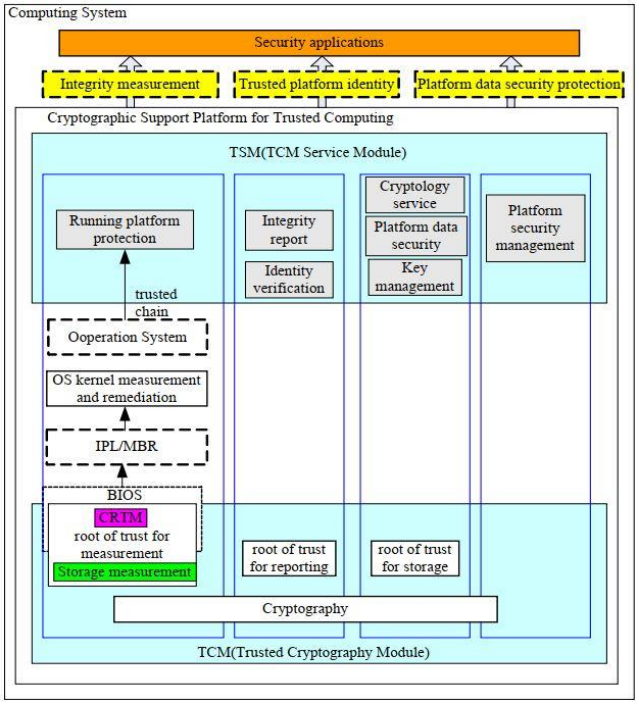
TCM (Trusted Cryptography Module)
- TPM issued by Chinese Government
- Functions as root of trust for measurement, reporting, and storage
- Provides attestation, memory protection, and outside authentication

TCB consists of Cryptographic Support Platform for Trusted Computing (CSPTC)
- TSM (TCM Service Module)
- TCM

Minimization: TSM used to hide resource management and TCM commands from untrusted applications
- Used as defined entry point into TCM

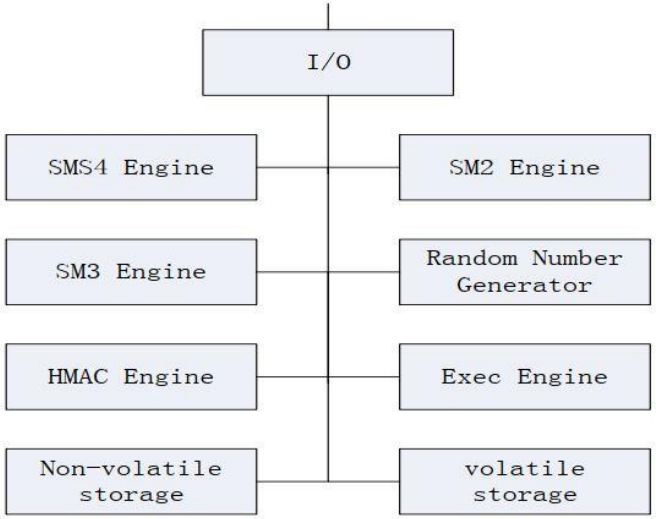
Allows for Secure Boot

Utilizes layered approach (next slide)

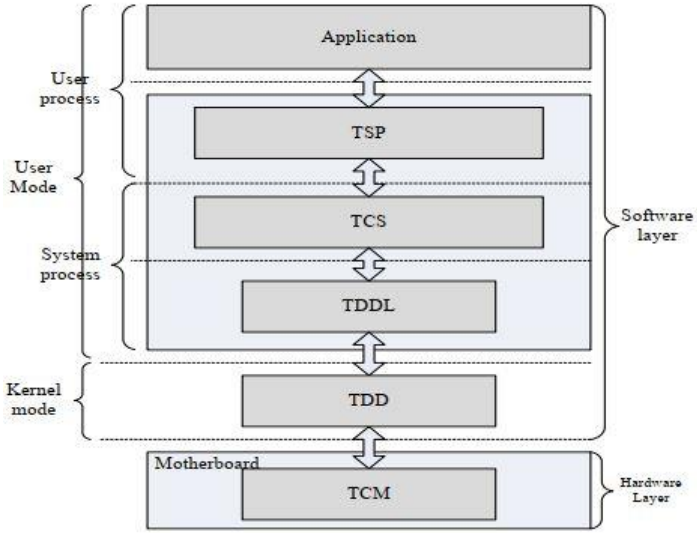


(Wang, 2010)

TCM and TSM Structures



TCM Structure (Wang, 2010)



TSM Structure (Wang, 2010)

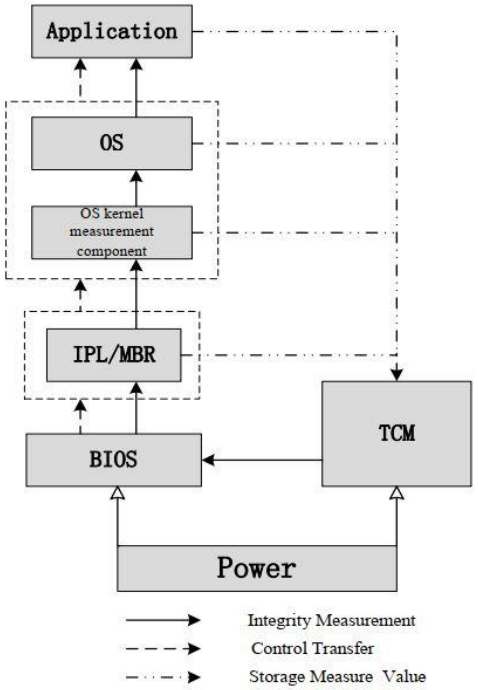
TCM Trust Chain

1) TCM Root of Trust used to measure integrity of BIOS
   - Value stored in PCR (Platform Configuration Register) within TCM and make determination

2) If BIOS passes -> It is allowed to run and BIOS measures integrity of IPL / MBR

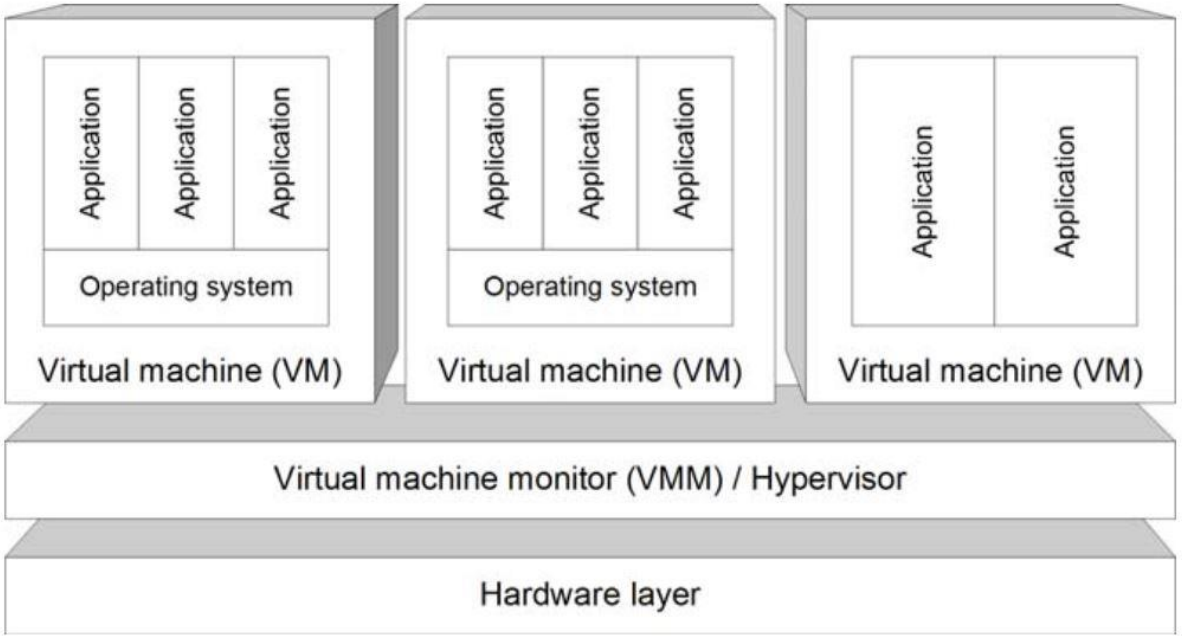
3) Process continues with each measurement being sent to the TCM and stored in PCR

This process allows for a Trust Chain to be built starting with the TCB



(Wang, 2010)

VMM Architecture Review



(Stumpf, 2009)

Method 2: Multi-Purpose ECUs (VMM)

(Stumpf, 2009)

Strategy developed by team that included several members of the BMW Group Research and Technology team

Based on trend towards centralization using multipurpose ECUs

- More economic
- Lighter weight
- Hardware efficiencies
- Centralized security measures

Autonomous and Safety Functions require v2x communication of ECU data

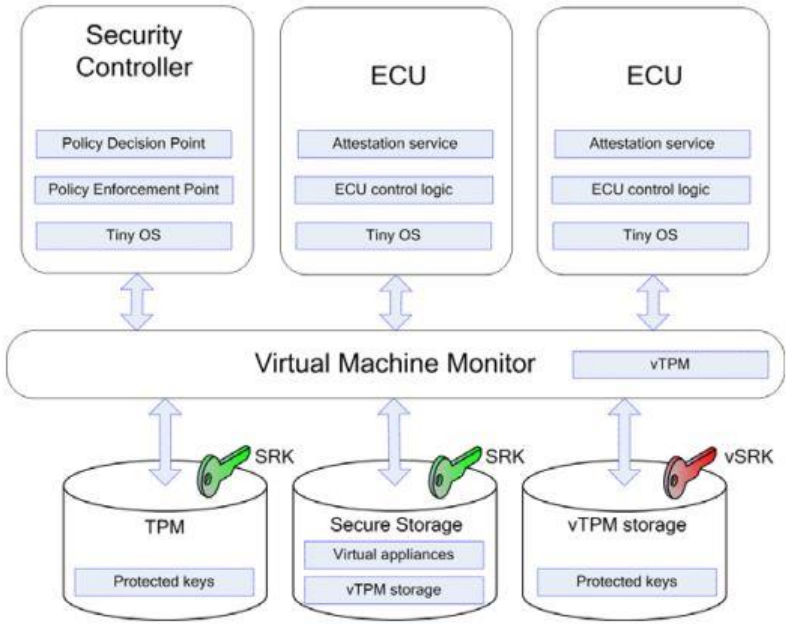
- Major goal of safety features are to reduce fatal traffic accidents

Approach allows for relatively easy migration of existing ECU architectures

- Requires minimal security service implementation

Facilitates internal and v2x communication

- Evaluates integrity and trustworthiness of senders



(Stumpf, 2009)

VMM Application Domains

Each ECU represents a separate domain

- These separate domains need the ability to communicate
- Different levels of trust
- Ability to communicate determined by policy-based MAC controls
  - ❖ Implemented using TCB (VMM + TPM + Security Controller)
  - ❖ Potentially too much code for minimization practice

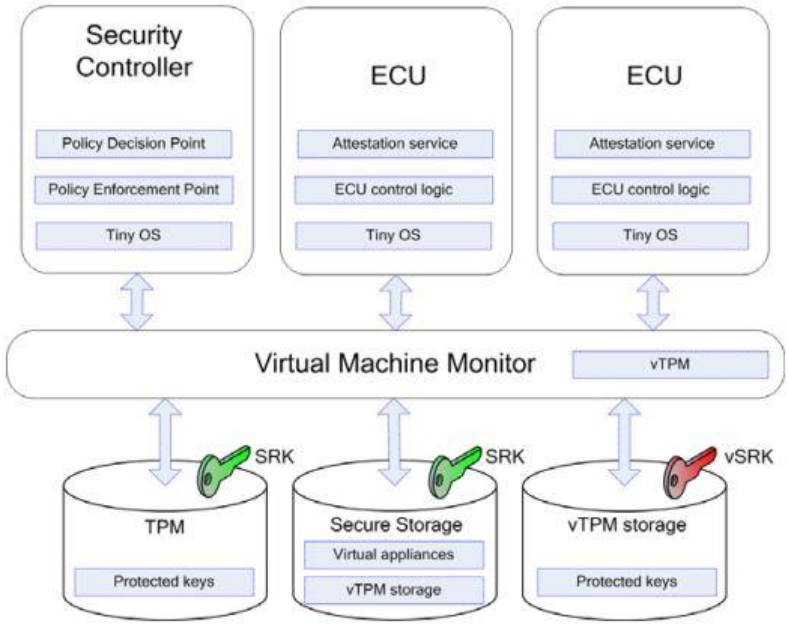
Security controller acts as Reference Monitor

- Controls all information flow between ECUs

TPM performs all security-critical operations

- Protects against application-level subversion
- Secret keys never leave TPM

Supports Secure Boot



(Stumpf, 2009)

VMM Application Domains

vTPM: Instance initialized for each ECU VM

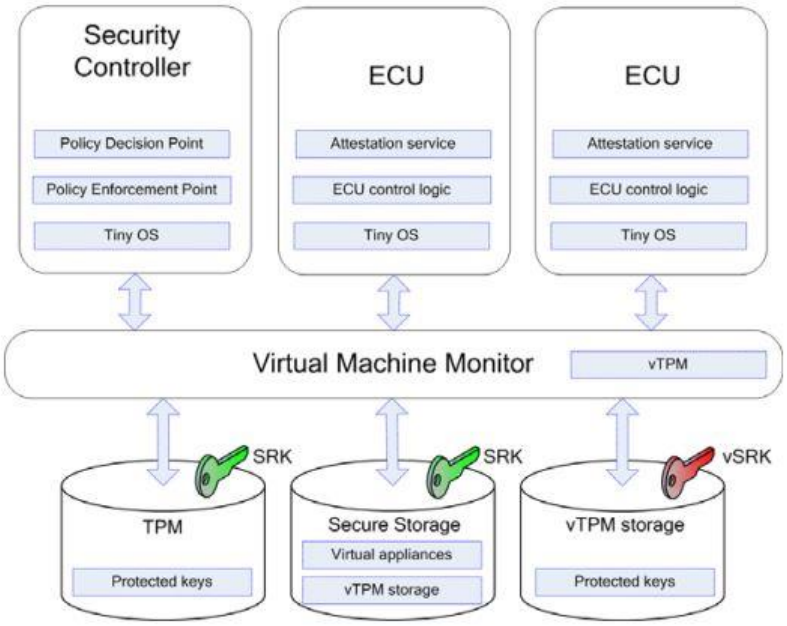
- Allows for each VM to attest to secure state to external actor without having to provide information directly from TPM

Uses of minimization:

- Security Controller code is kept small for verification purposes
- Each VM consists of minimal OS and single application

External attestation also relies on trusted third party (Privacy Certification Authority) used to guarantee Attestation Identity Key's (AIKs) created by the TPMs endorsement key (EK) for each VM

- Creates anonymity for specific vehicle
- What happens if connectivity is unavailable for remote entity?



(Stumpf, 2009)

Method 3: KPS-based attestation (Oguma, 2008)

Background: Seeks to solve issues regarding time constraints created by attestation for autonomous vehicles

- Includes both inter-vehicle and v2x communication
- Both forms of communication rely on ECUs to be attested in order for trust to be established
- Focuses on idea that ECUs must communicate quickly in order to function as intended and claims that traditional PKI/RSA based TPM approach is too slow for practical use in autonomous vehicles – needs to happen in less than 1ms

Requirements / Assumptions:

- Unauthorized modifications are not allowed by ECUs
- All communication between ECUs must be authenticated (trust) and encrypted (integrity)
- Authentication and encryption checks should function even if an ECU has been replaced or software has received an authorized update
- Cryptography method used should be expected to last 15-20 years

Proposed technique is to provide remote attestation using TPM as root

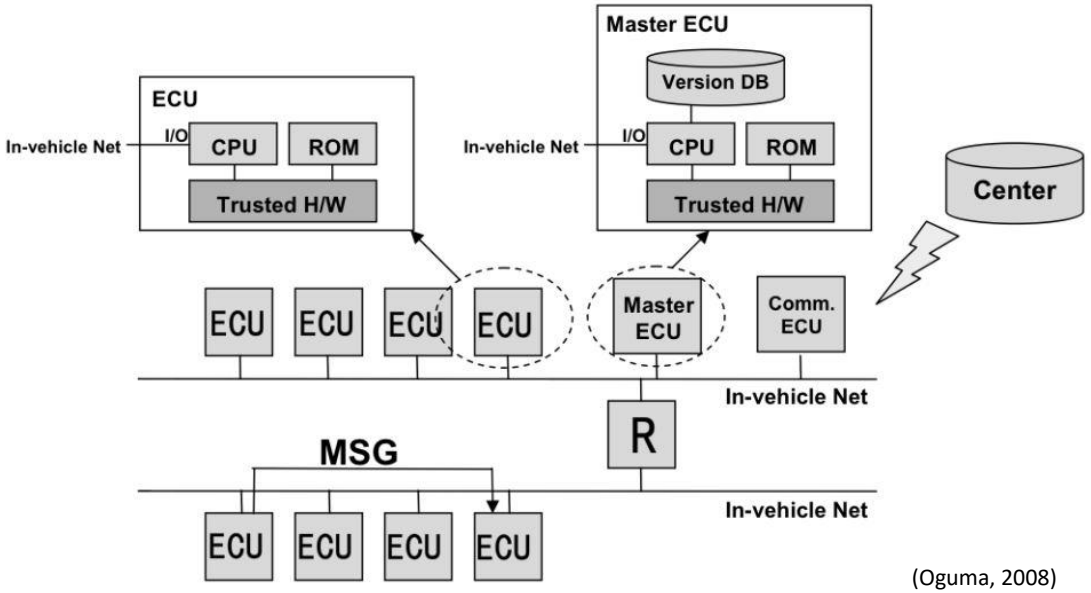
- Does not rely on external third party

Method 3: KPS-based attestation

Center
- List of all valid software versions and their ROM hash values
- Shared symmetric key
- KPS key generation matrix randomly generated for specific vehicle

Master ECUs
- Unique identifier for Master ECU
- Shared key with Center
- List of acceptable hash values
- KPS key generation algorithms for ECUs

ECUs
- Unique identifier
- Shared key with Center
- KPS key generation algorithm



(Oguma, 2008)

KPS-based Attestation Summary

Attestation performed locally within the vehicle

- Secret keys for each Master ECU and normal ECU based on KPS (Key Predistribution System)

Keys predistributed because number of ECUs are static

- Removes need for TPM (Center) to perform every attestation on the system; workload can be distributed among several Master ECUs
- Removes most computationally expensive processes from TPM (signing and decrypting)

Researchers claim this model to be 42 times faster than an RSA-based model

- Dependent on length of RSA key (assumed at RSA1280)
- KPS-based Calculations performed with the assumption of 300 ECUs

PERFORMANCE COMPARISON

(A) KPS Performance "decryption" (Intel Core2Duo 1.06GHz)

T=30	T=100	T=300
$87\mu s$	$222\mu s$	$358\mu s$

(B) RSA Performance "decryption" (Intel Core2Duo 1.06GHz, OpenSSL)

RSA512	RSA1024	RSA1280	RSA2048
$1,247\mu s$	$5,811\mu s$	$(15,000\mu s)^*$	$32,265\mu s$

*estimated

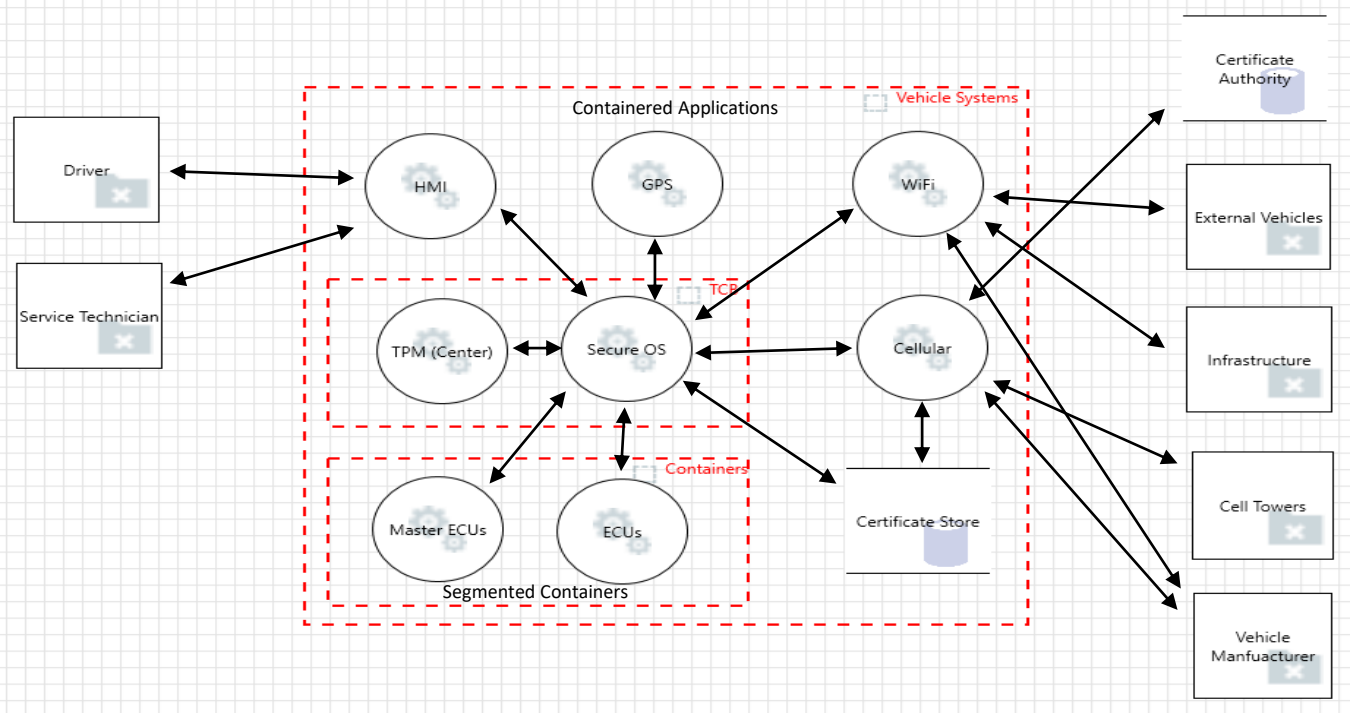
(Oguma, 2008)

Suggested Improvements

Based on the research of the previous three methods:

- Utilize containers instead of a VMM architecture to create multi-purpose ECUs
- Custom build Secure OS for containers to run on
  - Need to minimize lines of code in order to allow for verification
  - Viable solution due to simplicity of ECUs compared to typical modern day applications running on Windows/Linux
  - Could replace CAN for autonomous vehicles
- Integrate KPS-based attestation for internal vehicle communications
- Also utilize TPM-based attestation for v2x communication when possible
  - Third party certificate authorities (CA) could be used consistently or pseudo-randomly
    - KPS-based attestation could function as a backup when internet is not available if external CAs are being used
  - PKI infrastructure should still be used for secure software distribution

STRIDE Data-Flow-Diagram (DFD)



References

[1] Contreras J., Zeadally S., Guerrero-Ibanez J.A. Internet of vehicles: Architecture, protocols, and security. IEEE Internet Things J., vol 5, no 5., 3701-3709, 2017.

[2] F. Stumpf, C. Meves, B. Weyl, and M. Wolf. A security architecture for multipurpose ECUs in vehicles. In 25th Joint VDI/VW Automotive Security Conference, Ingolstadt, Germany, 2009.

[3] H. Oguma, A. Yoshioka, M. Nishikawa, R. Shigetomi, A. Otsuka and H. Imai, "New Attestation-Based Security Architecture for In-Vehicle Communication", Proc. of IEEE Global Telecommunications Conference (GLOBECOM), pp. 1-6, 2008.

[4] Lindeman, T. (2017, December 13). The 5G Network Probably Won't Be Good Enough for Self-Driving Cars. Retrieved November 9, 2020, from https://www.vice.com/en/article/qvzd3v/5g-network-wont-be-good-enough-for-autonomous-cars.

[5] Wang, J., & Gao, W. (2010). Securing Internet of Vehicles Using TCM. International Journal of Digital Content Technology and Its Applications, vol 4, issue 7., 226-233.

[6] Zhang Z. M., Jiang W. R., Xue Y. B. A Trust Model Based Cooperation Enforcement Mechanism in Mesh Networks. In: Proceedings of IEEE ICN, pp. 28-33, 2007.

Questions?

Autonomous Vehicles

Assurance by Design and Architecture

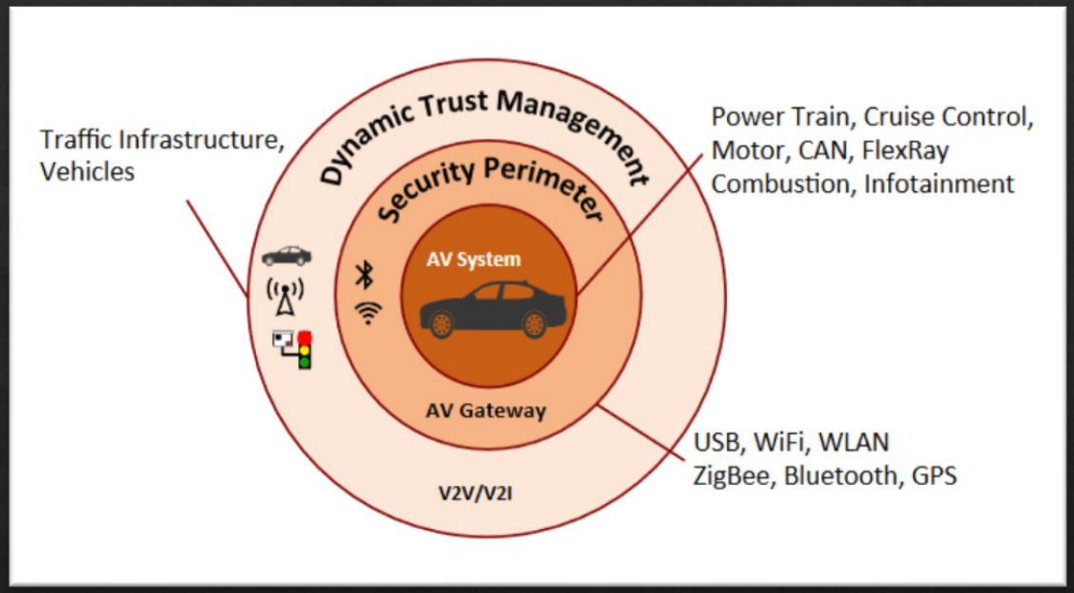
Amarbir Singh

Outline

- What needs to be high assurance?
- Identify the trusted computing base/bases
- Security by design, design for assurance
- Information hiding, minimization, modularization and layering
- Trusted platform module, root of trust
- Architectural assurance of the AV
- Infrastructure assurance, different infrastructures
- Evidence for assurance
- Trusted distribution, maintenance and disposal assurance
- Security assurance framework and safety standards

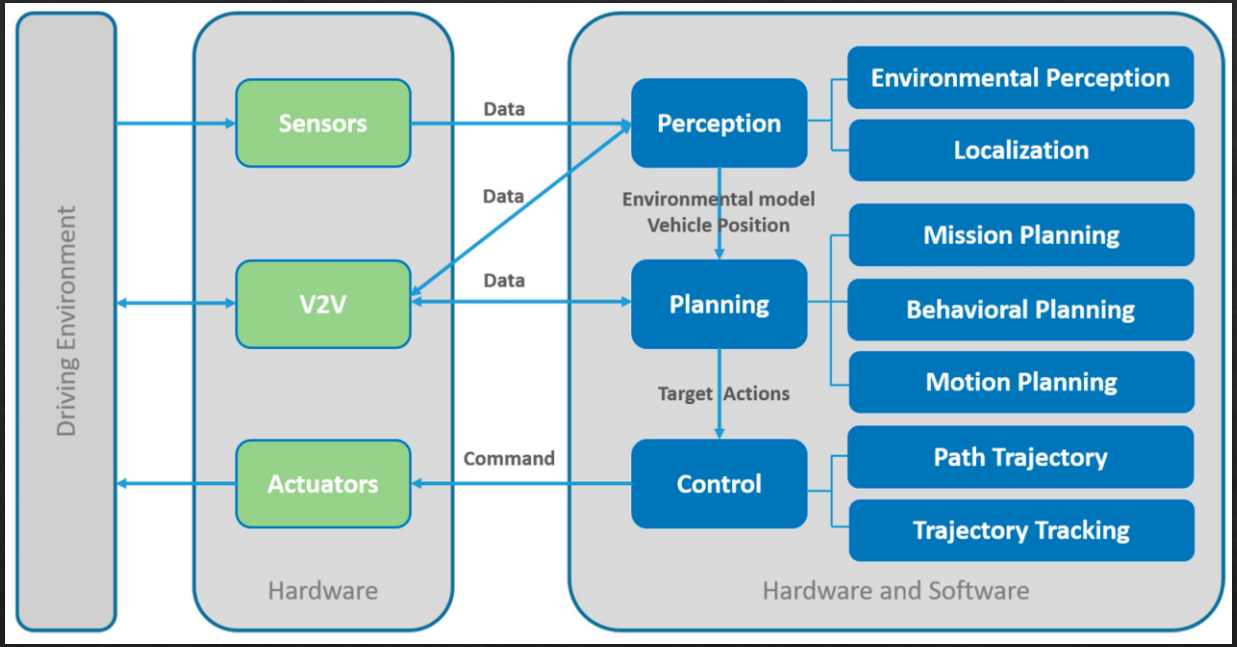
Trusted Computing Base

- TCB- The totality of protection mechanisms within a system, including hardware, firmware, and software which is responsible for enforcing a security policy
- Spread over multiple devices controlled by different parties
- Security perimeter

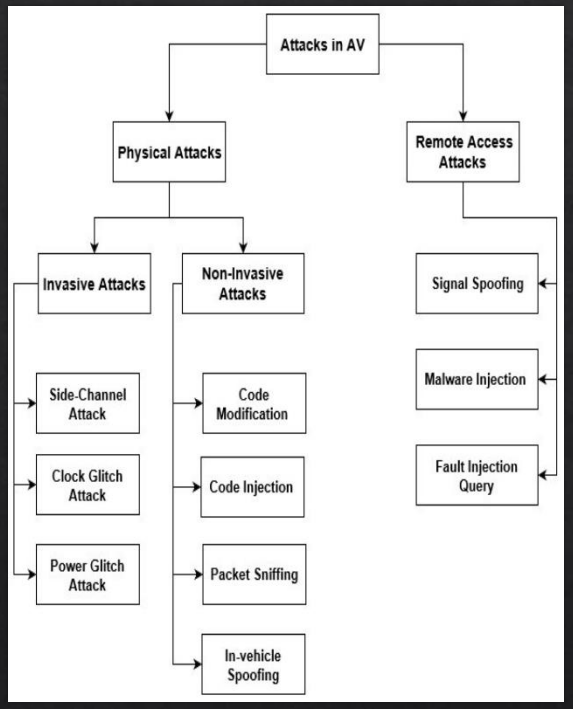


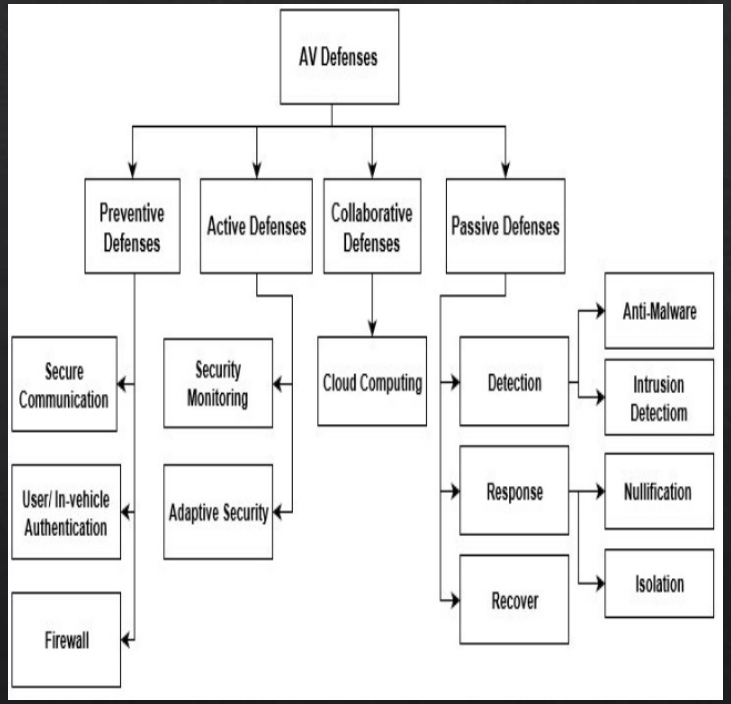
Autonomous Vehicle: Security by Design

AV Architecture



Threat Modeling





Threat Modeling

- Bayesian network for CAV cyber-risk classification
- Incorporates known software vulnerabilities
- US National Vulnerability Database (NIST)
- Derived from Common Vulnerability Scoring Scheme (CVSS)

- https://www.sciencedirect.com/science/article/pii/S096585641830555X

Input Data												
Case	Attack	Mitigation Technique	Version	Access Vector	Access Cmplx.	Authentication	Conf. Impact	Integrity Impact	Avail. Impact	Base Score	Base Severity	Exploitability
1	GPS Spoofing	None	2	NETWORK	L	N	N	COMPLETE	COMPLETE	9.4	H	PROOF OF CONCEPT
2	GPS Spoofing	Military-grade Cryptography	2	NETWORK	H	N	N	COMPLETE	COMPLETE	6.6	M	UNPROVEN
3	GPS Jamming	None	2	NETWORK	L	N	N	PARTIAL	COMPLETE	8.5	H	PROOF OF CONCEPT
4	GPS Jamming	Military-grade Cryptography	2	NETWORK	H	N	N	PARTIAL	COMPLETE	5.6	M	UNPROVEN

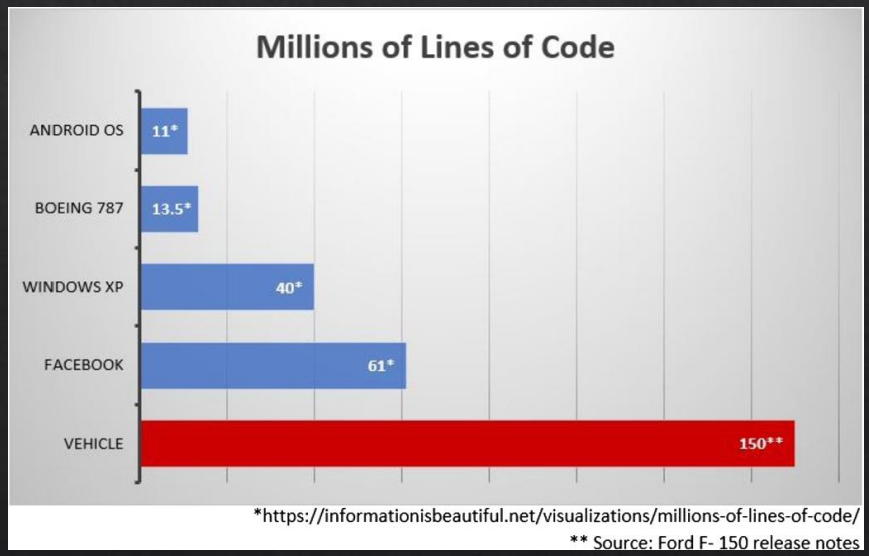
Security Objectives and Requirements

- Integrity of remote-control functions of the AV
- Integrity of the sensor systems
- Integrity of the safety-related control operations braking and speed control
- Integrity of the navigation-related control operation
- Confidentiality of communications between AV and traffic management system
- Confidentiality/integrity of communications between AV and its manufacturer
- Integrity and authenticity of communications between AV and maintenance workshop
- Confidentiality of cryptographic keying materials
- Derived from safety objectives

Minimization

- Minimize the trusted components of the system
- Smaller attack surface
- Easier to formally verify and prove secure
- Easier to understand, test and debug
- Leverage root of trust, trusted third party

- Many using exploitable Linux kernel, processors
- TCB spread over multiple devices/domains
- Constantly changing with OTA updates



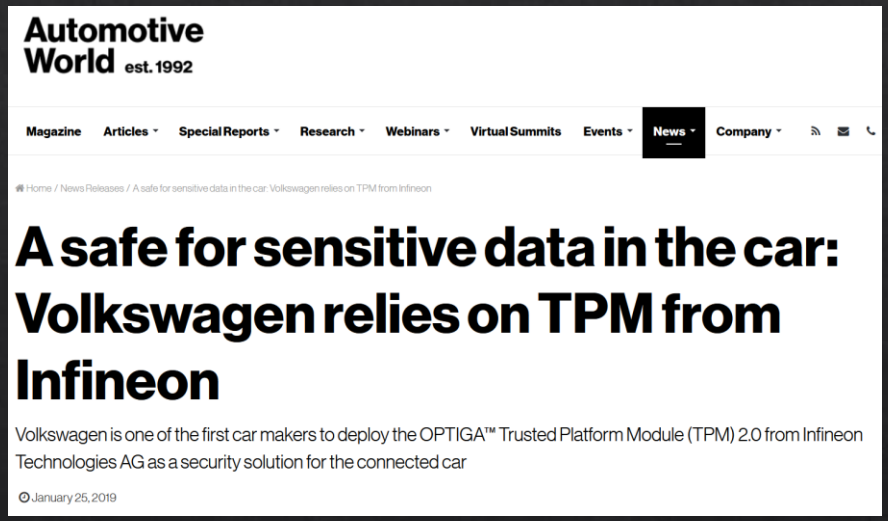
Minimization

- Treat only the most critical components as trusted:
- Collision Avoidance, Navigation, Remote Control
- A minimized component which does not need to be changed i.e. TPM in Security Kernel
- With embedded hard-wired keys and manufacturer certificates, read-only
- Use it to verify the system's integrity on start up
- Use it to verify updates are from legitimate manufacturer

- Treat all other components- Infotainment, climate control etc. as untrusted and mediated by security kernel
- Self driving system trusted or untrusted? Needs to be changed often.

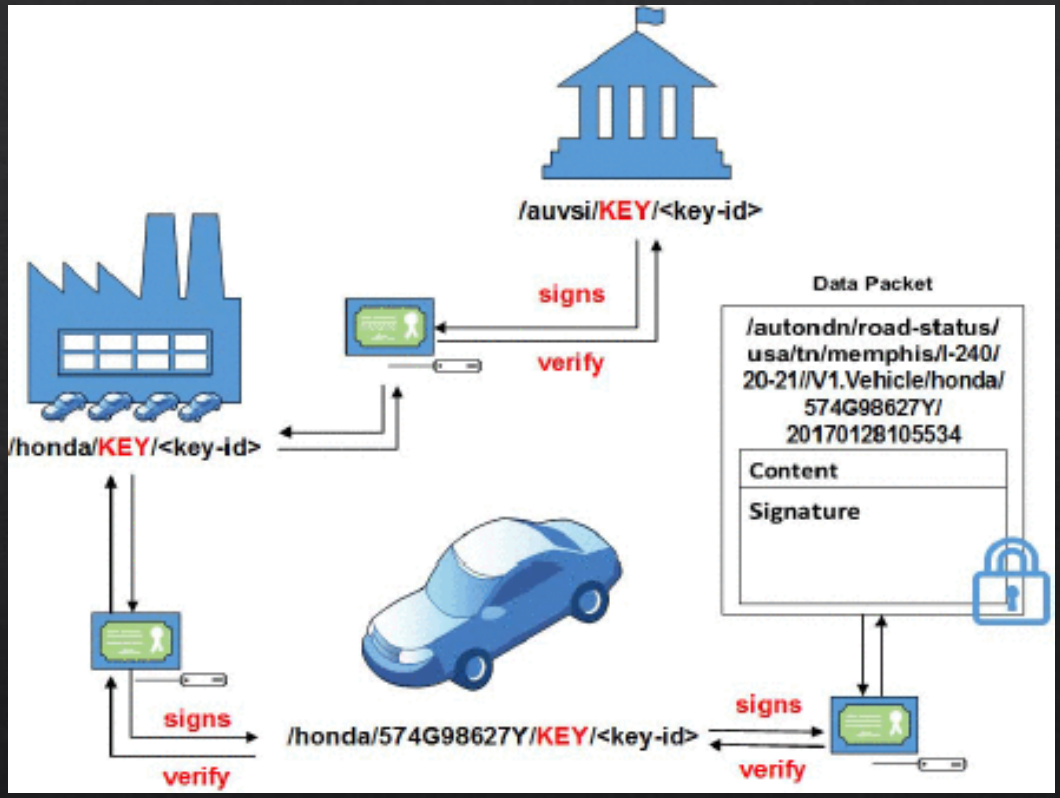
Trust Management

- Trusted Third Party- key exchange and entity authentication
- Trusted Platform Module- secure environment for storing the secret cryptographic key, authentication/attestation
- Infineon Technologies OPTIGA TPM 2.0
- "*Its firmware, including cryptographic mechanisms ("crypto-agility"), can be updated remotely making sure that its security technology is always state-of-the-art"*
- Physically Unclonable Function (PUF)
- Trusted V2V communication
- Trusted Infrastructure
- PKI-based trusted infrastructure in smart phones



Root of Trust

- Assuring the association of a cryptographic key with the truthful identity of the key owner
- In order to reduce the likelihood of accepting false information, a vehicle should authenticate all received data or only safety critical data
- AV is also assumed to have a secure environment for storing the secret cryptographic key



Modularization and Layering

- Separate TCBs on separate subsystems-
- Vehicle itself, connected phone, traffic infrastructure etc.
- Use TCB subsets approach, TNI to reason about assurance of overall system

- Trusted execution, trusted boot
- BIOS/firmware -> bootloader-> kernel-> software image

- Well defined interfaces promoting information hiding
- Low coupling and high cohesion

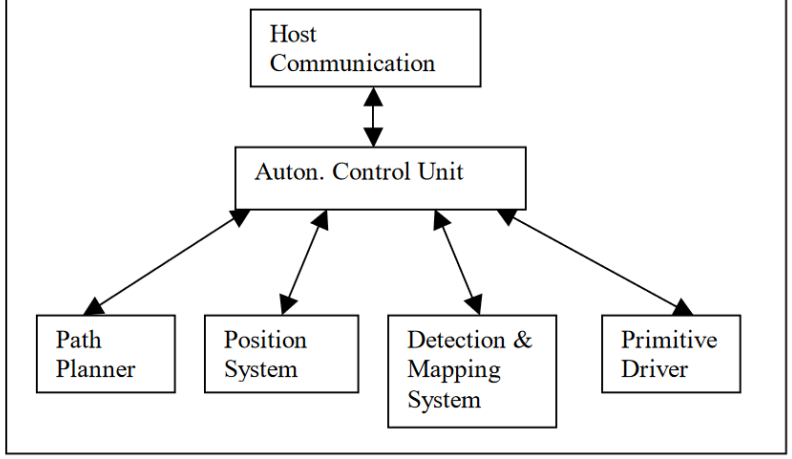
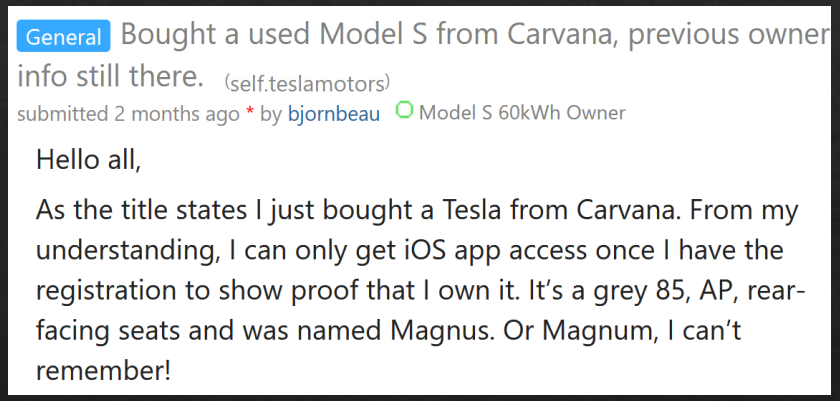


Figure 2: Modular and Scalable Architecture

Assurance after Manufacturing

- Use trusted distribution techniques- checksums and seals, when consumer uses AV for the first time
- Store a snapshot of the "secure" state of the system
- Use of VPNs for OTA updates
- Frequent checks against the secure state
- Verify updates before applying using embedded keys in the security kernel
- Appropriate steps after repairs and change of ownership
- Secure Disposal


General Bought a used Model S from Carvana, previous owner info still there. (self.teslamotors)
submitted 2 months ago * by bjornbeau Model S 60kWh Owner

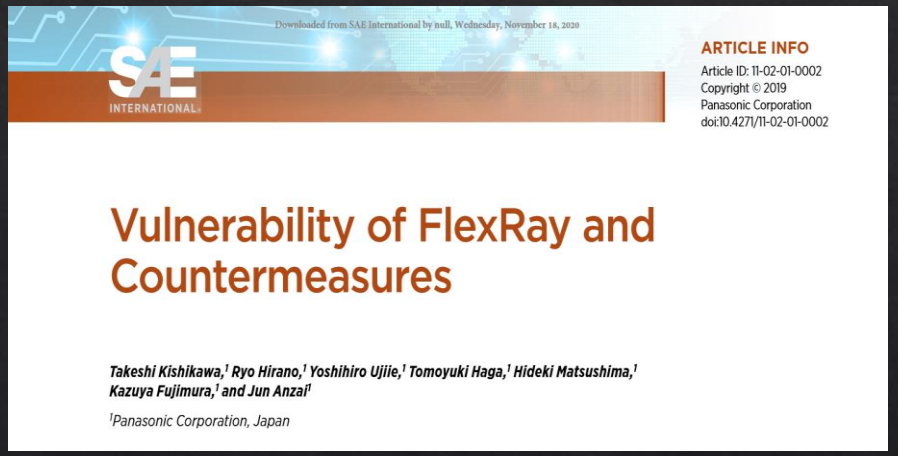
Hello all,

As the title states I just bought a Tesla from Carvana. From my understanding, I can only get iOS app access once I have the registration to show proof that I own it. It's a grey 85, AP, rear-facing seats and was named Magnus. Or Magnum, I can't remember!

Intra-vehicle Communication

- Multiple sensors and subsystem must work together
- Control Area Network (CAN), FlexRay, Local Interconnect Network (LIN), Media Oriented System Transport (MOST), and ethernet
- Inherits vulnerabilities of all these protocols

- Encrypted communication between different links
- Use guards to stop information flow counter to the policy
- Guards must be high assurance
- Interface specifications



SAE INTERNATIONAL

ARTICLE INFO
Article ID: 11-02-01-0002

doi:10.4271/11-02-01-0002

Vulnerability of FlexRay and Countermeasures

Takeshi Kishikawa,[1] Ryo Hirano,[1] Yoshihiro Ujiie,[1] Tomoyuki Haga,[1] Hideki Matsushima,[1] Kazuya Fujimura,[1] and Jun Anzai[1]

[1]Panasonic Corporation, Japan

Inter-vehicle Communication

- WIFI, Bluetooth, Zigbee, Cellular communication, 5G
- Disseminate False Messages, disturb system
- Vehicular Botnets
- Bogus Electronic Control Unit (ECU), Bogus infrastructure

- Strongest form of protection profiles
- Use of Virtual Private Networks(VPNs) for updates
- Input validation, output sanitization, whitelisting
- Interface specification
- Always fail safe

Manufacturer Capability

- Previously mentioned Jeep recall
- No/vague regulations for AV manufacturers
- Time to market, competition

- Capability Maturity Models
- Secure Software Development Lifecycle
- Open-Source Software, bug bounties
- Government oversight, 3rd party evaluations
- Universal standards and practices

Infrastructure Assurance

- Command and control locations (Manufacturers)
- Existing traffic infrastructure (Government)
- Telecommunication infrastructure (3rd parties)
- Combination of different infrastructures (3rd parties)
- Heterogeneous network integration
- Resolving conflicts between different sources of information

- Hardware subversions, same supplier for parts
- Government mandated backdoors

Testing

- Testing is still necessary
- Amount of miles driven by AVs
- Penetration testing and Fuzzing
- Outside testers
- Network testing

- Testing is never complete

These Chinese hackers tricked Tesla's Autopilot into suddenly switching lanes

Published Wed, Apr 3 2019•11:17 AM EDT • Updated Wed, Apr 3 2019•12:22 PM EDT

Tom Huddleston Jr.

Share

Tesla Model S P85D dual electric motor sedan. Myung J. Chun | Los Angeles Times | Getty Images

A group of Chinese hackers published a report showing how they tricked Tesla's Autopilot self-driving software into swerving into an oncoming traffic lane.

The group of cybersecurity researchers from Keen Security Labs in China placed bright-colored stickers on the road to create a "fake lane" that tricked the self-driving software of a Tesla Model S into veering from the appropriate driving lane into the opposing lane on a test course, where oncoming traffic would be driving in a real-life scenario.

Trending Now

1. Chuck Schumer says Biden could forgive $50,000 in student debt with executive order
2. Dr. Fauci says masks, social distancing will still be needed after a Covid-19 vaccine—here's why
3. Bill Gates on his WFH schedule during the pandemic, including what he likes about it
4. Bill Gates on the difference between Elon Musk and Steve Jobs
5. Shark Tank: This single mom built a million-dollar online business—her story made Kevin O'Leary cry

Security Assurance Framework

- EU-funded H2020 SAFERtec project
- *"assurance framework to assess the level of confidence that the involved security-, privacy- and safety- needs of the connected vehicle system are satisfied"*
- Framework relies on Common Criteria (CC)
- Experimental evaluation over a reference implementation

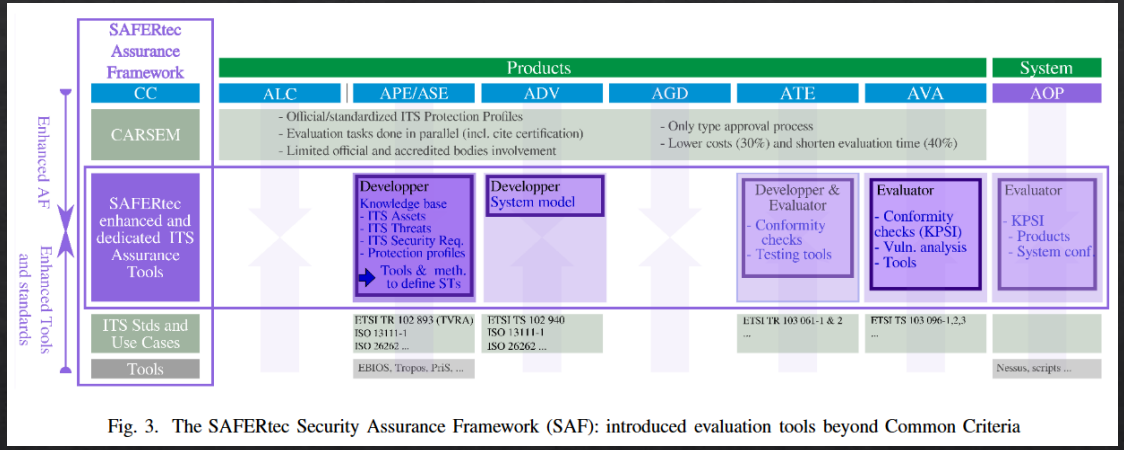


Fig. 3. The SAFERtec Security Assurance Framework (SAF): introduced evaluation tools beyond Common Criteria

Current safety standards

- ISO 26262- Safety lifecycle throughout the phases like management, development, production, operation, service, and decommissioning.
- Automotive Safety Integrity Level (ASIL)
- ASIL = Severity×Exposure×Controllability

- Society of Automotive Engineers (SAE) J3061- Establish the awareness and a common terminology across the AV supply chain.
- Cybersecurity-critical system are not necessarily safety-critical, however, the reverse is true

- European Union Agency for Network and Information Security (ENIS)- develop best practices in keeping smart cars safe from cyber threats

Conclusion

- Safety objectives require high assurance of security objectives
- Large attack surface over multiple domains and subsystems
- Need for minimized TCB, security kernel, MAC rather than RBAC
- On the right track by using TPM and secure communication protocols
- Require a lot more government oversight, 3rd party evaluations
- Need for security/safety standards and assurance frameworks
- Universal protocols, combining different infrastructures
- Develop, use and leverage high assurance products from ground up
- NHTSA developing a formally verified and mathematically proven message parser for V2V communication interfaces
- Much bigger problem as the technology progresses

References

- https://arxiv.org/pdf/1810.00545.pdf
- https://www.mdpi.com/1424-8220/19/3/648/htm
- VEHICULAR 2019 : The Eighth International Conference on Advances in Vehicular Systems, Technologies and Applications
- https://www.sciencedirect.com/science/article/pii/S096585641830555X
- https://www.sciencedirect.com/science/article/pii/S221420961930261X
- https://news.ycombinator.com/item?id=21934290
- https://www.automotiveworld.com/news-releases/a-safe-for-sensitive-data-in-the-car-volkswagen-relies-on-tpm-from-infineon/
- https://ieeexplore-ieee-org.libproxy1.usc.edu/document/7946859
- https://cimar.mae.ufl.edu/cimar/pages/pubs/auvsi_2000.pdf
- http://illmatics.com/securing_self_driving_cars.pdf
- file:///C:/Users/amarb/AppData/Local/Temp/11-02-01-0002.pdf
- https://www.pentestpartners.com/security-blog/reverse-engineering-the-tesla-firmware-update-process/
- https://www.cnbc.com/2019/04/03/chinese-hackers-tricked-teslas-autopilot-into-switching-lanes.html
- https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/
- http://cgi.di.uoa.gr/~ppantaz/publications/2018SmartVeh.SAFERtec.pdf
- https://www.nhtsa.gov/technology-innovation/vehicle-cybersecurity
- https://www.hitachi-systems-security.com/blog/smart-car-security-threats-is-the-connected-car-a-good-idea/

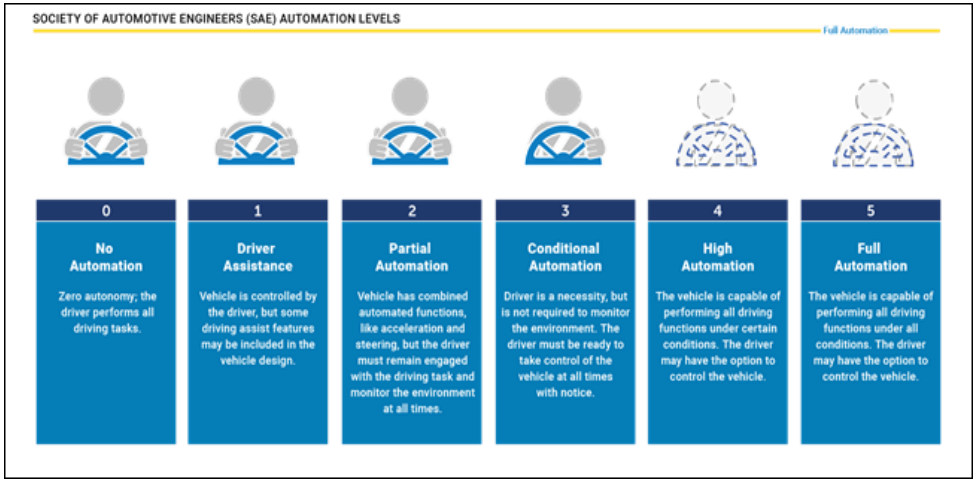
Security Assurance in Connected & Automated Vehicles

Abhishek Tatti
DSCI -523

Outline:

- Introduction and why assurance is difficult ?
- Threat Modelling
- Identifying Trust Boundaries
- Architecture - Secure Cyber Physical Systems
- Issues that can help build assurance arguments
- CAV Hacks
- Questions and Feedback
- References

Overview



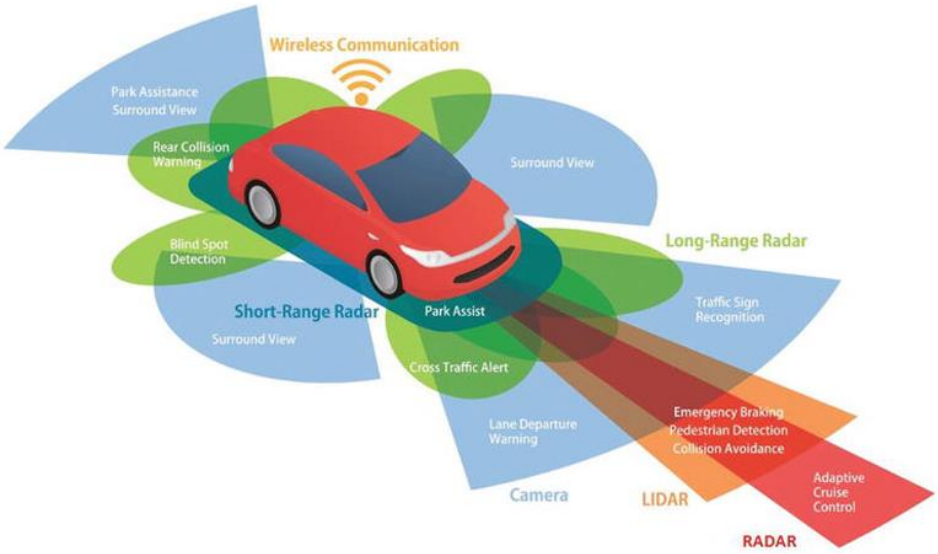
Autonomous vehicles are a reality - Waymo has racked up more than 10 million miles of running autonomous cars on public roads.

Laws and regulations to support CAVs - Arizona, California.

Key Players - Waymo, GM Cruise, AI Agro, Tesla, Uber and many more.

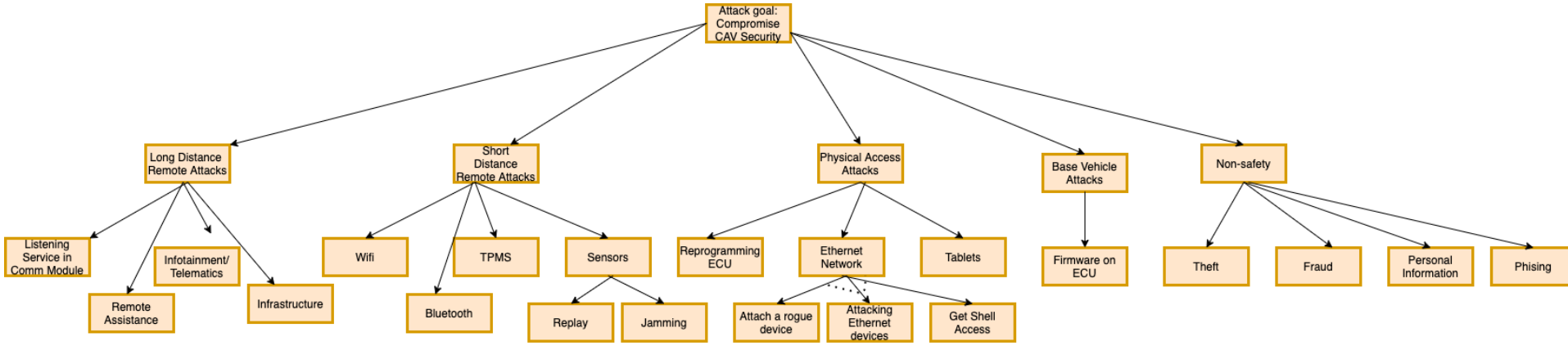
Challenges with CAV assurance:

- **Life of Vehicle is more** - More chance for hacker to find vulnerability
- **Assurance solutions are time consuming** - Encryption, authentication
- **ECU** (Memory & Computations constraints)
- **Privacy** - Location, Driving History
- **Cyber Physical** - Sensors, actuators - LIDAR, cameras, radar, light matrices, devices for sensing angular momentum of the wheels, & automated brake and steering control.
- **Computation and communication**- under hard real-time requirements

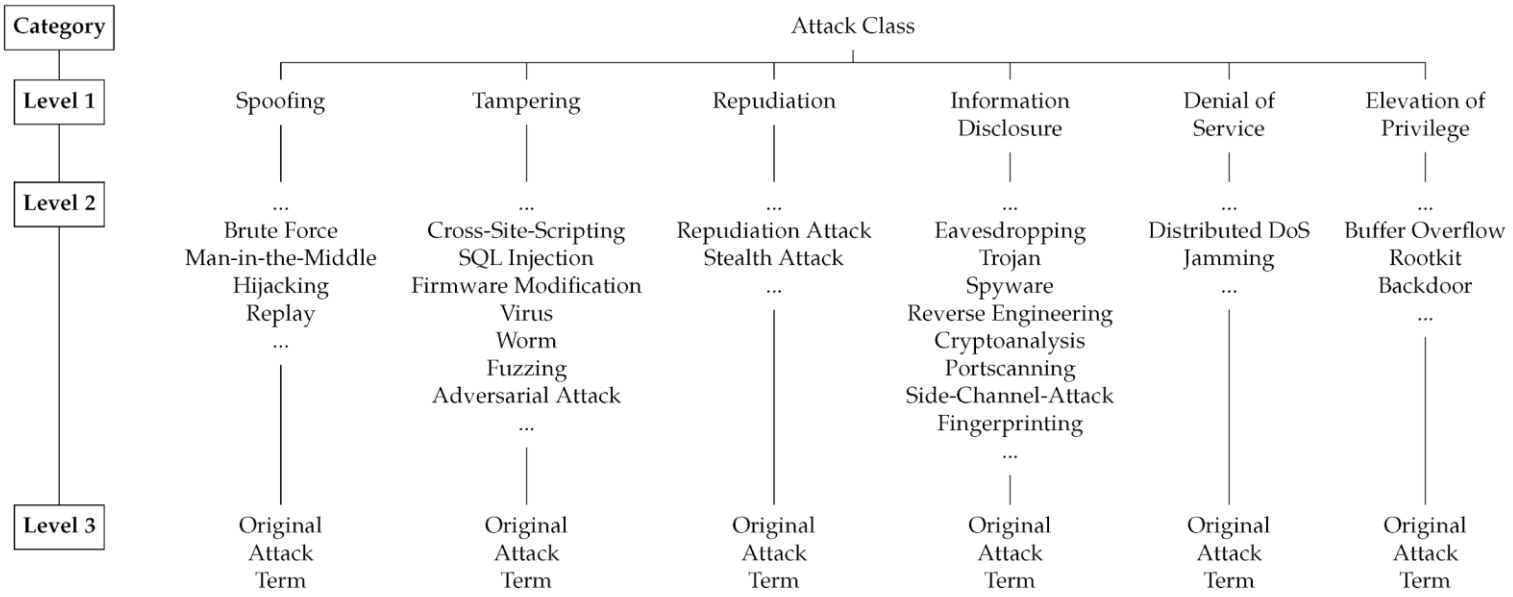


"The top attack vectors as of 2018, according to an Upstream Security report,were remote server attacks (21%) and keyless entry attacks (19%)"

Threat Modelling - Attack Tree



Threat Modelling - STRIDE Model For CAV



Identifying Trust Boundaries - (1)

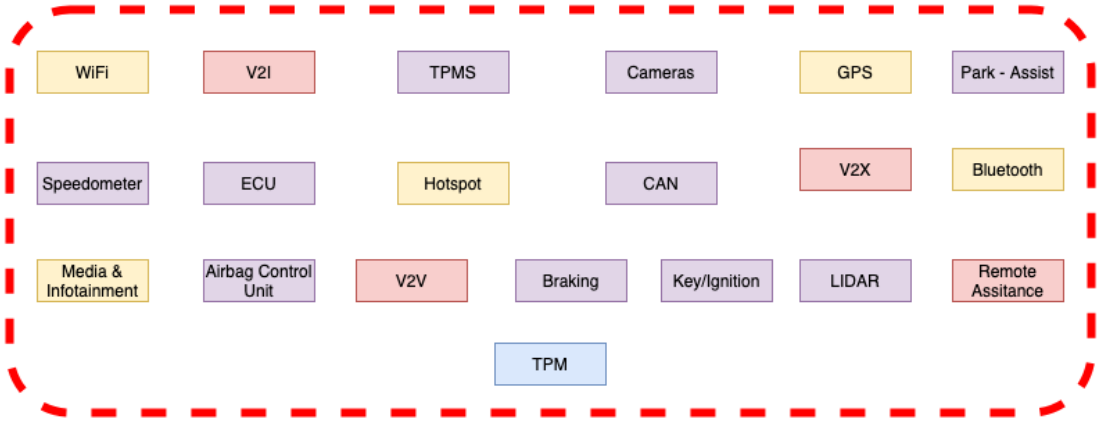
SAFETY CRITICAL ASSURANCE ARGUMENTS:

1. Ability to start & stop in any situation
2. Ability to change direction in any situation – Wheels/Steering
3. Airbags to be released in specific conditions
4. Ability to control the Speed, for example the speed should not increase beyond a certain limit in any situation.
5. Ability to safely park vehicle when needed
6. Ability to get in and out of the vehicle at will
7. Ability to Boot in Valid State

ATTACK TYPES

1. Denial of Service
2. Network Protocol attacks
3. Rogue Updates
4. Malware
5. Crypto attacks
6. Phishing

Identifying Trust Boundaries - (2)



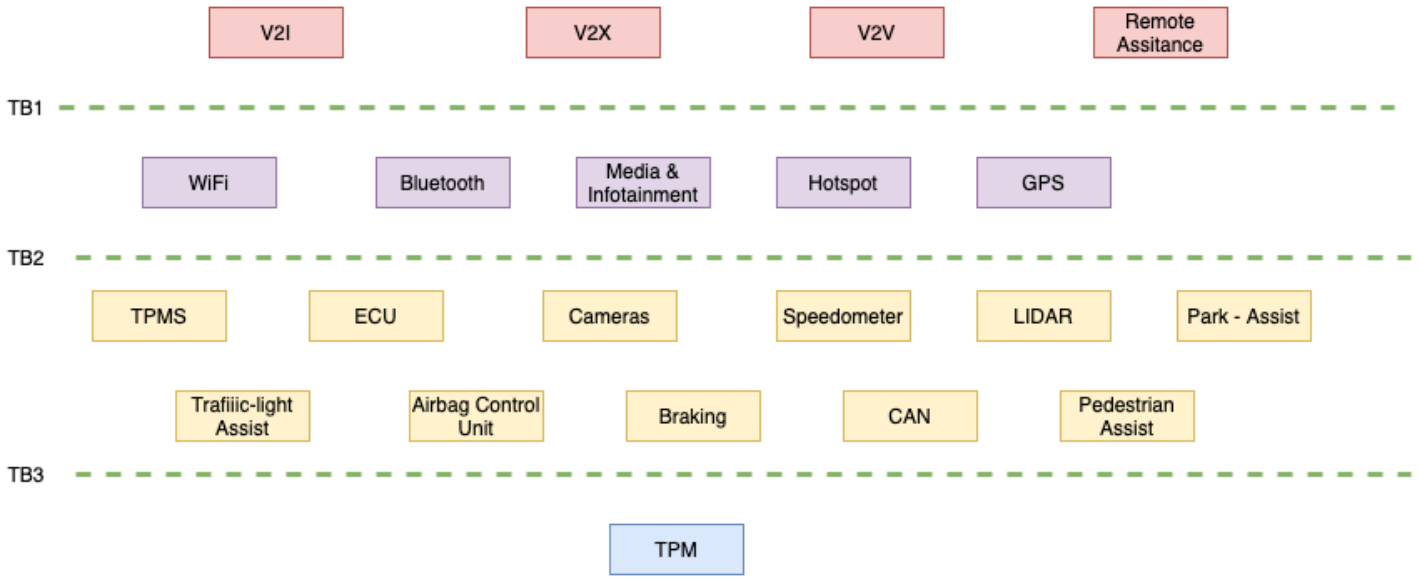
Components that affect the Safety Critical properties:

Components susceptible to malware/subversion due to the running software/rogue updates/phishing

Components susceptible to DOS/N-P attacks/ Crypto-attacks

Components susceptible to access control/authentication/digital signature attacks

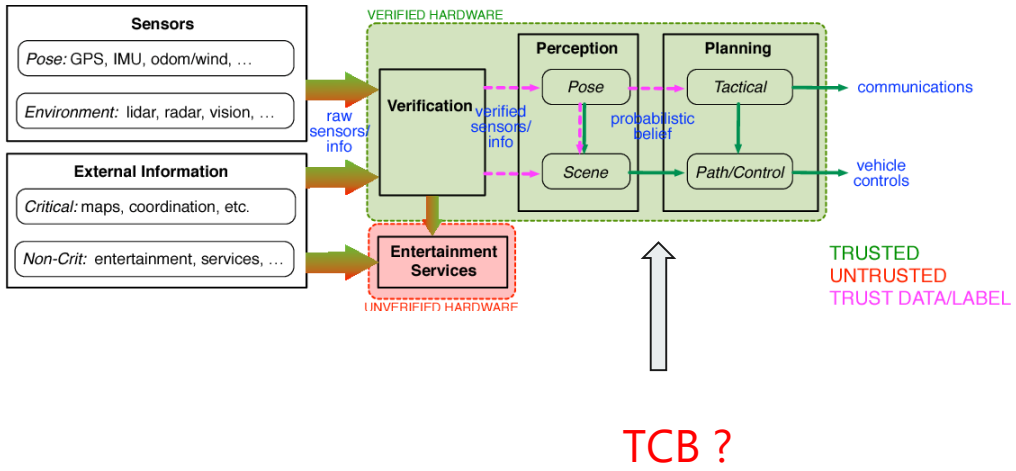
Identifying Trust Boundaries – (3)



Architecture - Cyber Physical Autonomous Assurance

Verifiable Information Flow Control:

- Input verification, collision-avoidance software
- Software-level information flow control and verification.
- Verified hardware platform for information flow control.



Source: https://dl.acm.org/doi/10.1145/3264888.3264889

Arguments helpful to build for assurance:

- **Trusted Computing – Secure Boot & Update**
- **Secure Key Storage**
- **Attack Surface Reduction**
- **Encryption**
- **Segregation, Segmentation & Isolation**
- **Secure Communication**
- **Message Signing**
- **Remote Assistance**
- **Threat Detection**
- **Sensors , Ethernet Switch**
- **Tablets**
- **Other**

Trusted Computing - Secure Boot & Updates

- **Trusted Execution** i.e. ensuring **Code verifiability** on system boot up
- Secure Boot method to verify the code cryptographically using a **trusted chain anchored by a ke**y
- Ensure this key stored in **tamper-resistant and write-protected** part of the system
- The code may still get tampered at run-time
- Code and Data Signing i.e. Not only ensuring signed code is running at boot but also for any **updates**

Secure Key Storage

- **TPM Hardware Chip**
  - Tamper-resistant hardware device like a smart-card , securely bound to the computing platform
  - Root of Trust for integrity measurement and reporting
  - Ensures that malicious software cannot compromise any cryptographic secrets
  - Security-critical operations such as key generations and decryption operations are done “on-chip”, so that secret keys do not have to leave the chip
- **HSM (Hardware Security Module):** HSM is a removable or external device that can generate, store, and manage RSA keys used in asymmetric encryption
- **ARM Trustzone:** allows running a secure operating system (OS) and a normal OS simultaneously from a single core
- **Secure Enclave:** hardware-based key manager, which is isolated from the main processor to provide an extra layer of **security**

OPTIGA™ TPM 2.0 - Infineon Technologies

- First TPM for CAV
- Deployed by Volkswagen
- Encryption, decryption, signing and verification
- Supports advanced encryption algorithms RSA-2048, ECC-256 and SHA-256
- Extended temperature range from -40°C to 105°C
- Firmware, including cryptographic mechanisms, can be revised remotely ensuring that its security technology is always updated.
- To be used by many others

Encryption

- **Full encryption when the CAV is powered off -** Protect intellectual property
- Have the base operating system start up, authenticate to a service to get an encryption key and then use it to decrypt the proprietary software
- If known a vehicle or component **is missing**, we can **deny** its attempt to get the decryption key for the software
- Encryption is a useful tool in preventing the **unauthorized recovery** and **analysis of firmware**
- Firmware binary images may also be obtained from a firmware updating process hence organizations should reduce any opportunities for a third party to obtain unencrypted firmware during software updates

Segregation, Segmentation & Isolation

- **Privilege separation** with boundary controls
- Network segregation **to isolate connected** components from the components that **control** the automobile.
- Secure separate gateway module between **Ethernet** and **CAN networks**.
- Allow traffic from given ports / IP addresses to specific ports / IP addresses.
- Strong boundary controls, such as strict **white list-based filtering** of message flow between different segments, should be used to secure interfaces

Secure Communication

- **Restrict outbound communications** only to the fleet management and key servers
- Control Communication to Back-End Servers using **Certificate Validati**on & Outbound connections should authenticate the endpoint, either via TLS/SSL or some application-level protocol
- Avoid sending safety signals as messages on common data buses
- Employ a **message authentication** scheme to limit message spoofing
- **Message Signing**
  - Is it possible to sign all messages sent across the ethernet network considering the performance constraints in CAVs ?
  - **Prioritize messages** and classify them as critical
  - Ensure that every message that could directly affect **steering, braking, or acceleration** is signed
- All the communication to take place over a secure encrypted channel.

Attack Surface Reduction

- All unnecessary components should be removed.
- **For example:**
  - Reduce the inbound connections over the internet and allow outbound connections only
  - Radio/head unit is not needed due to the tablets being employed in the back seat, remove it

Threat Detection

- Analyze the ethernet network and the CAN network traffic in real time to identify **anomalies**
- If any anomaly is detected, it should report to the fleet management server.
- For more severe or more likely attack situations, the vehicle may bring itself out of service, safely pull over, or power down.
- Maintain **logs** of all security critical operations

Sensors

- Use multiple sensors and multiple sources of input.
- This way, there multiple sources of data from which to draw conclusions in case one of them is under attack.
- In such scenarios build a mechanism which decides which sensor is to be believed and acted upon.

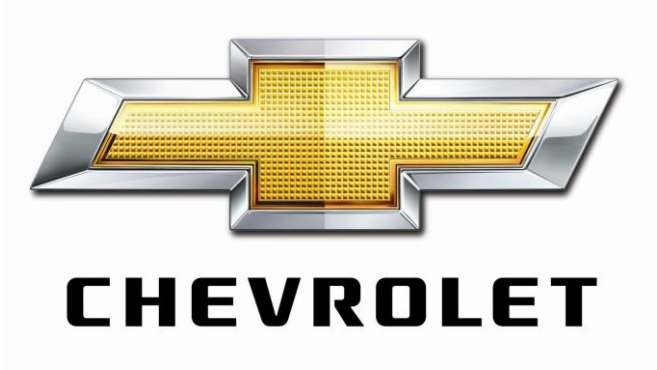
Ethernet Switches

- Ethernet switch or switches control the flow of the Ethernet traffic.
- Use 802.1x when possible as it provides **device authentication** and thus prevent unauthorized devices being plugged into the network.
- Ideal but is it feasible?

Miscellaneous

- Individual TCBs for each subsystem. I.e. multiple, distributed TCBs
- Developers should employ **good security coding practices, thorough testing** and use tools that support security outcomes in their development processes
- Tablets or other Media - consider them as low assurance components and block communication with critical components
- **Build fail safe mechanisms**: if a vehicle has failed to contact the fleet management server for a specified time, say 5 minutes, it should assume there is a problem and remove itself from service and return to the garage for inspection. If this fails, it should pull over and power off.
- Computers capable of performing **remote assistance** should be locked down, boot from read-only media, and not have unnecessary Internet access.
- Third party risk assessment, testing and accreditation of CAVs
- Establishing standards and governance authority that measures the assurance score of a CAV and only then grants permission to function

CAV attacks



2011 Chevy Malibu

- Researchers compromised the vehicle's radio using a vulnerability in the Bluetooth stack
- After the radio was compromised, the attacker were isolated from the high-speed CAN network by a gateway device.
- However, they could reprogram this gateway device from the low-speed CAN network, with which they had access. After this, they were able to send messages which could make the brakes lock up.
- In another attack, they could remotely compromise the vehicle's telematics unit using its cellular connection. They could do this by dialing the unit's phone number and communicating with its modem

2015 Jeep Cherokee

- Compromise the head unit of the Jeep due to a vulnerability that was accessible through the Internet
- After getting code running on the head unit, they were able to reprogram the firmware of another processor on the head unit that had CAN network access. After this, they could send CAN messages which controlled the steering, brakes, and acceleration of the vehicle

Jeep

CAV attacks

2016 Tesla Model S

- The CID (i.e. information display) ran an old version of a web browser. If they could trick a user into visiting a malicious website with the browser, or if the car had previously joined a well-known Wi-Fi network (like the Tesla Guest Wi-Fi network at a dealership), they could exploit this vulnerability if they were in physical proximity.
- After exploiting a vulnerability in the CID, they could reprogram the vehicle's gateway device which was connected to the CID over ethernet. This granted them the ability to send CAN messages which they used to engage the brakes of the vehicle.
- Later Tesla added code signing to the gateway device to make reprogramming of it harder. These researchers later demonstrated a way to bypass this security mechanism and still reprogram it with unsigned code

References:

- **Connected and autonomous vehicles - A cyber-risk classification framework**
  - Barry Sheehan , Finbarr Murphy, Martin Mullins, Cian Ryan
- **Cyber Threats Facing Autonomous and Connected Vehicles - Future Challenges**
  - Simon Parkinson , Paul Ward , Kyle Wilson , Jonathan Miller
- **The Security of Autonomous Driving Threats, Defences, and Future Directions**
  - Kui Ren, Qian Wang, Cong Wang, Zhan Qin, and Xiaodong Lin
- **Certified Control: A New Safety Architecture for Autonomous Vehicles**
  - Jeff Chow, Valerie Richmond et al
- **Survey and Classification of Automotive Security Attacks**

Jürgen Dürrwang and Reiner Kriesten

- **Secure Autonomous Cyber-Physical Systems Through Verifiable Information Flow Control**
  - Liu, Jed and Corbett-Davies, Joe and Ferraiuolo, Andrew and Ivanov, Alexander and Luo, Mulong and Suh, G. Edward and Myers, Andrew C. and Campbell, Mark
- **Securing Self-Driving Cars -** Charlie Miller & Chris Valasek
- **Waymo Safety Report - 2018**
- **National Highway Traffic Safety Administration. (2016, October). Cybersecurity best practices for modern vehicles. (Report No. DOT HS 812 333). Washington, DC: Author.**

Tesla Motors and their Technology Assurance Problem

Dwayne Robinson
DSCI 523
University of Southern California
November 22,2020

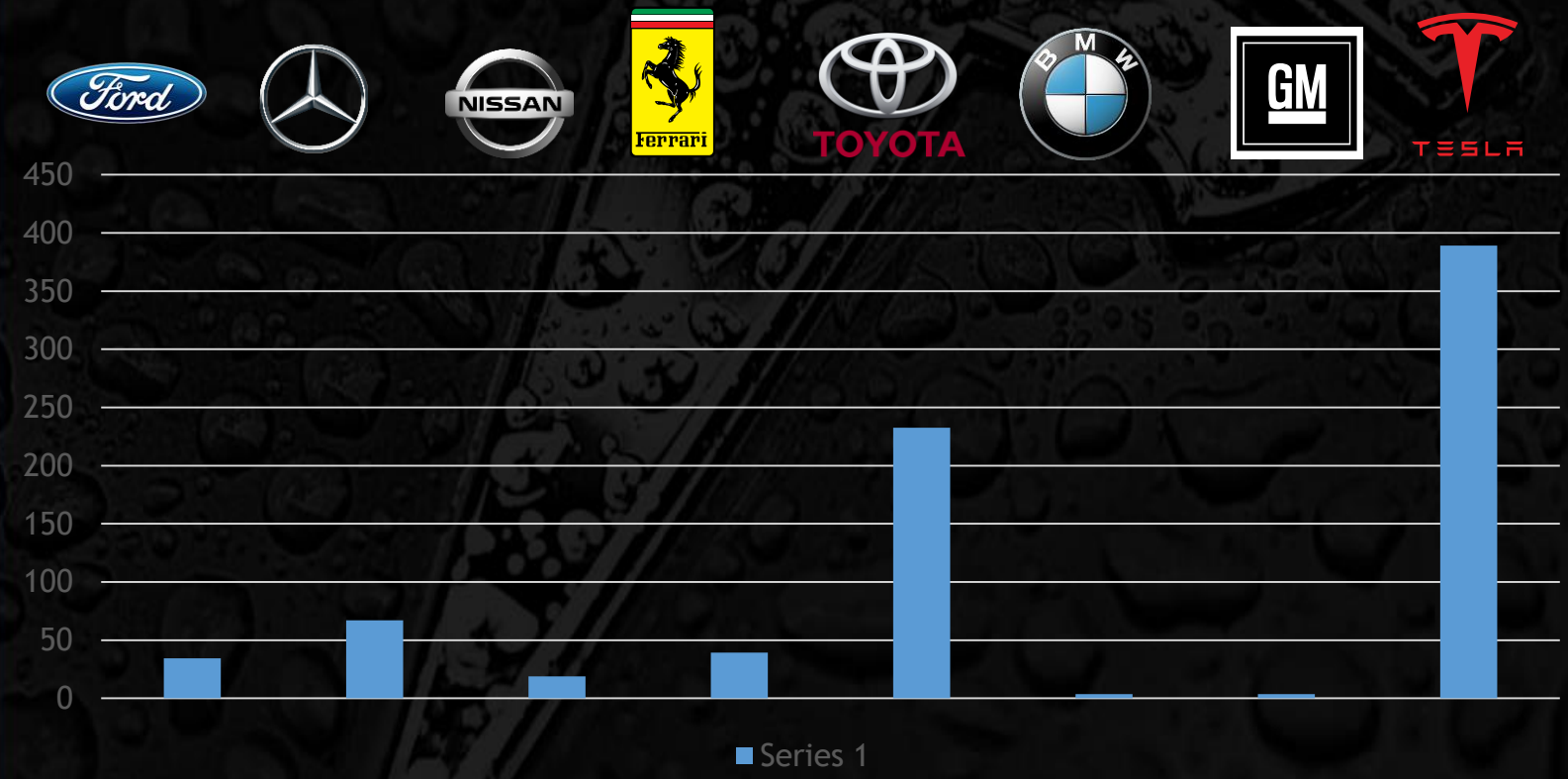
Table of Contents

- Tesla Motors – The Company
- Technology of Tesla vehicles
- Data Collection
- Usage of the Data Stored
- AutoPilot – Autonomous Vehicles
  - Levels
- Intrusion Opportunities
- Insure High Availability
- Threat Modeling Hacks against the fleet
- Assurance issues for the company
  - Assurance to the consumer for the data that is stored
  - Assurance issues for the vehicle components
  - Assurance issues for ’AutoPilot’
- Tesla Motors “Assurance” Program
- Conclusion

Tesla Motors The Company

- The company was founded in 2003
- Their 1st vehicle, the Roadster cost $109,000 and one of the original batch is for sale now for $1.5mm
- The Roadster was the first in a model of cards
- They Sold 2450 Roadsters until it was discontinued
- Today they have sold
  - 2012 Model S - Debuted
  - 2015 Model X – Debuted
  - 2017 Model 3 - Debuted
  - 2019 Model Y – Debuted
  - 500,000+ Number of Reservations for the Cyber Truck – Estimated Release date 2022

Tesla Motors The Company Market Capitalization in Billions US$



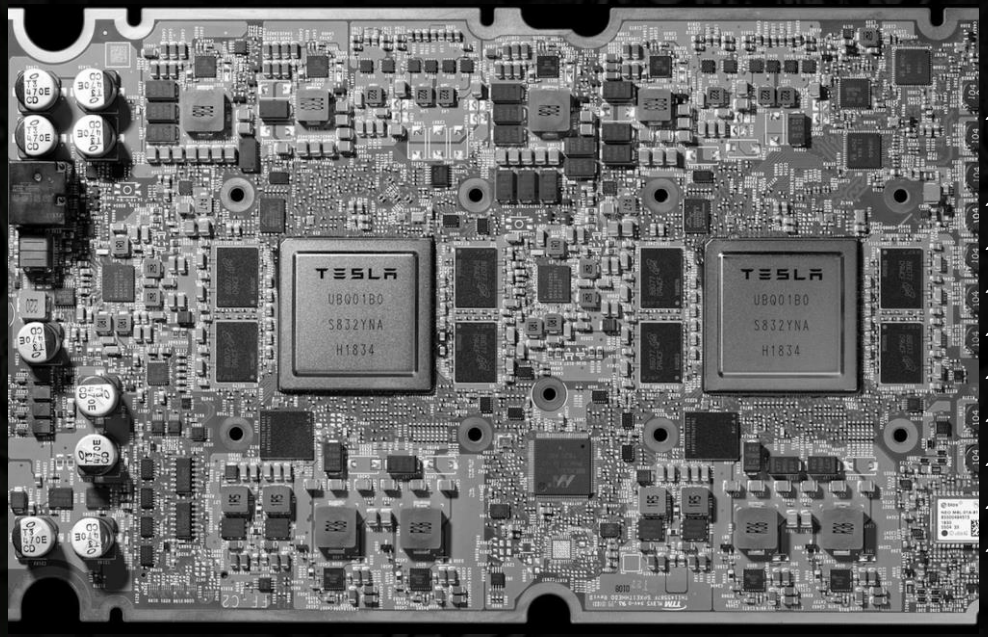
Automotive Companies and Autonomous Driving Initiative

- Baidu
- Tesla
- Argo AI
- GM
- Waymo
- Apple
- Daimler
- BMW
- Audi
- Volkswagen
- Lyft
- Uber

Tesla Motors The ‘TECH’ Company

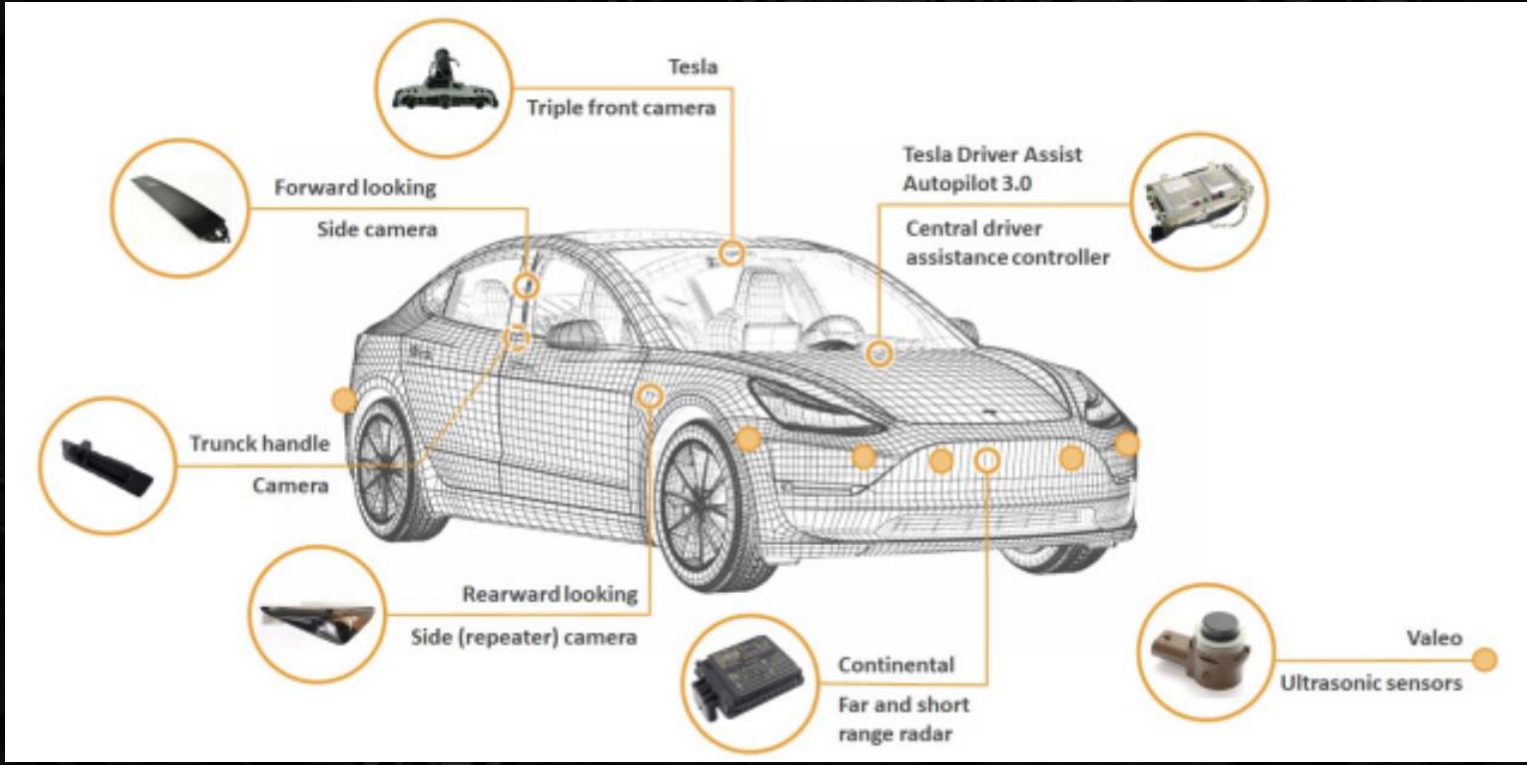
Model S (all test vehicles) not a car but a ‘sophisticated computer on wheels’
~Elon Musk La Times March 19, 2015

Image below is of Telsa self designed FSD / AutoPilot Chip / Processor



~ AI Chip Ruse 2Ghzm 36trillion operations per second
~ 2 – AI Brains for redundancy (Safety)
~ Neural Network Processor
~ Main System Processor
~ GPU
~ Image Signal Processor
~ LPDDR RAM Interface on Chip
~ Safety System
~ Video Encoder
~ Compute Encoder / Redundancy

Technology on the Exterior of The Vehicles



Technology Inside The Vehicles

What is in the Data Collection

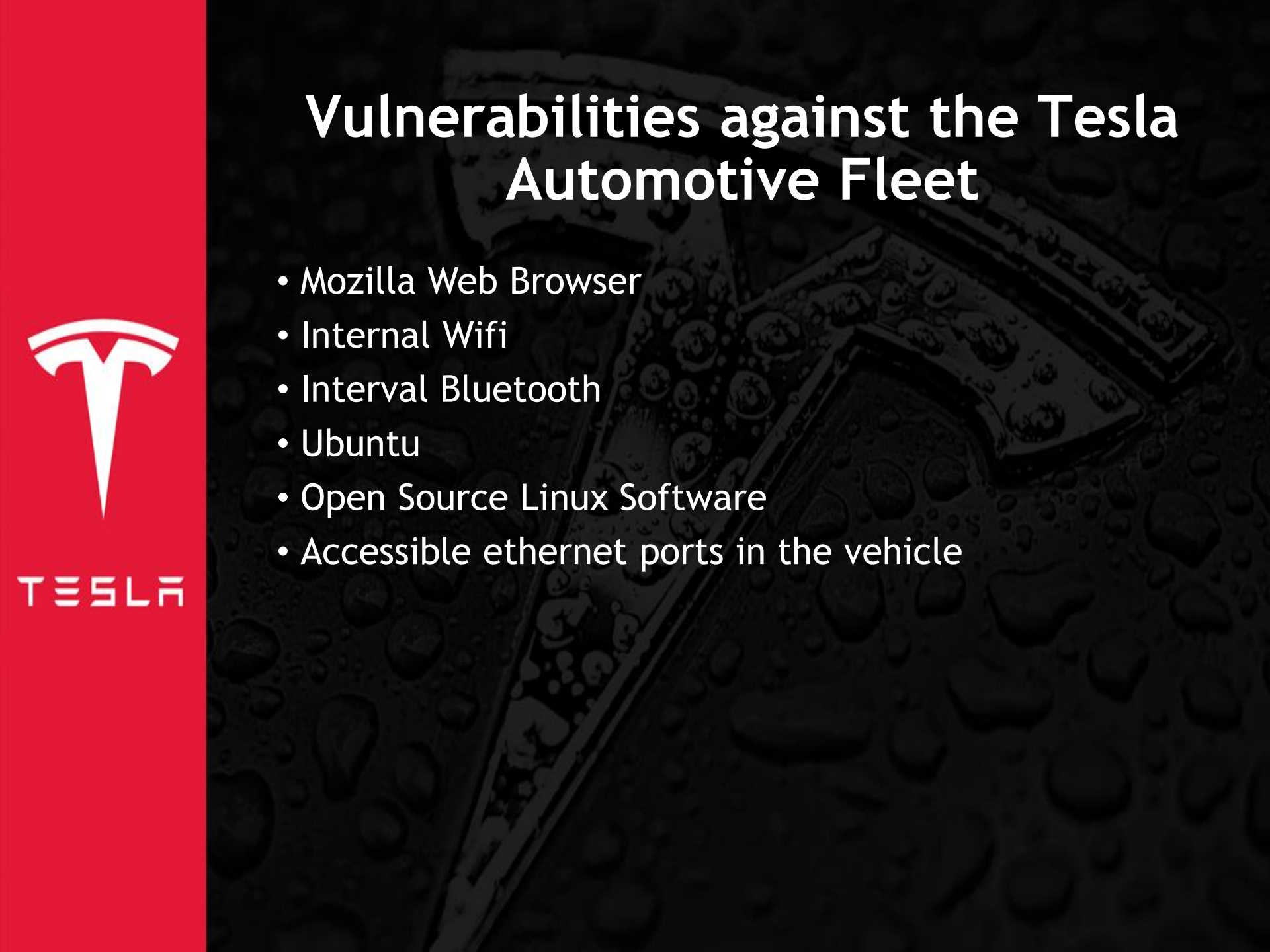
- Vehicle's location / GPS Coordinates
- Car's personal settings
- Contacts you've synced from your phone
- Addresses you've plugged into the navigation system
- Favorite radio stations.
- Speed and Driving habits
- Your mileage
- Where and when you charge the battery
- Airbag deployments
- Braking and acceleration details
- Autopilot, Tesla's assisted-driving feature, is engaged or disengaged
- Whether you have your hands on the wheel as you should
- Cameras and other sensors to log every detail about what they encounter while driving
- Short video clips from the car’s external cameras to learn how to recognize lane lines, street signs and traffic light positions
- Crash Data
- Fleet learning capability

Tesla AutoPilot

- Tesla is arguably the leader in the autonomous vehicle segment.
- Their autonomous vehicle feature in a market beta is consider Level 2 (1-5)
- As of April 2020, Tesla has reported more than 3billion miles have been logged under their autonomous driving feature ‘AutoPilot’
- All Tesla Vehicles produced after October 16m 2016 have the hardware capability for autonomous Level 5 once approved

With all these potential vehicles Tesla has a Assurance Dilemma

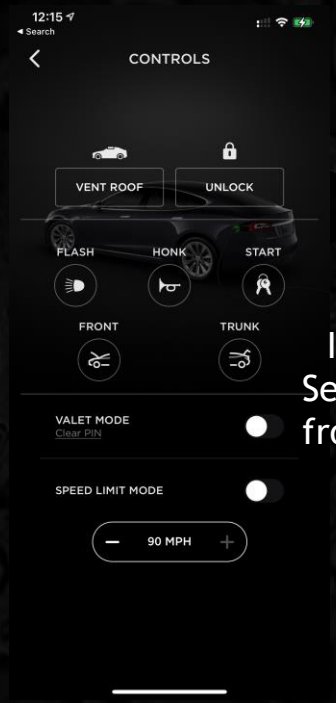
Intrusion Opportunities



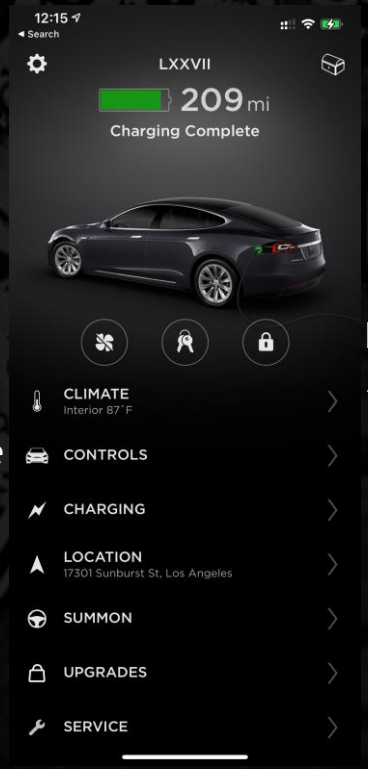
Vulnerabilities against the Tesla Automotive Fleet

- Mozilla Web Browser
- Internal Wifi
- Interval Bluetooth
- Ubuntu
- Open Source Linux Software
- Accessible ethernet ports in the vehicle

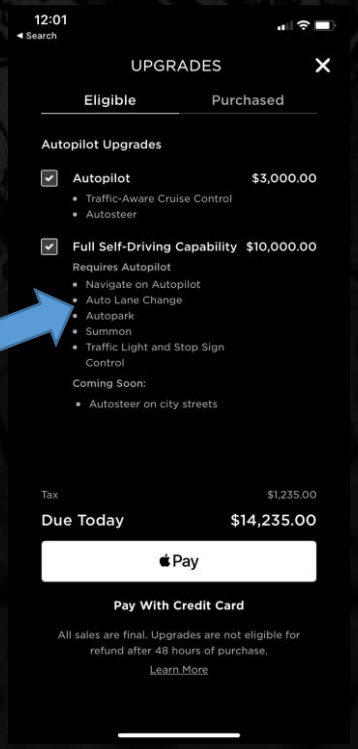
Intrusion Opportunities (Mobile)



Initiate Various Security & actions from Mobile phone



Purchase ADD-ONs



How can Tesla Improve?
Major issues acknowledgement.

One tidbit all Tesla Vehicles can be stated via a mobile phone.
Elon Musk Admits that Tesla
is late on security and vows to implement 2fA

High Availably Requirements (automotive industry)

- Reliable network-based structures, requiring redundant, real-time architecture design
- LIDAR / Radar Uninterrupted availability
- Timely detection and rapid response to potential vehicle cybersecurity incidents
- Architectures, methods, and measures that design cyber resiliency
- Facilitate rapid recovery from incidents when they occur
- Methods for effective intelligence and information sharing across the industry to facilitate quick adoption of industry-wide lessons learned.

High Availably Requirements (technical industry)

- Security against intrusions
- Roadmap to protect against current and future intrusions against vehicle infrastructure (patching)
  - Wifi
  - Bluetooth
  - Mag Keycard Hack
  - Mobile Phone Hack
- Privacy and Security against data stored in corporate data centers
- Protection from internal bad actors (Employees & Vendors)

Hacks against the Tesla Automotive Fleet

- Tricked Tesla's Autopilot self-driving software into swerving into an oncoming traffic lane.
- WiFi 'Automotive Breaking' Attack
- Wifi Vehicle Hack
- Published Acceleration Hacks
- Infotainment System
- Internal Web Browser

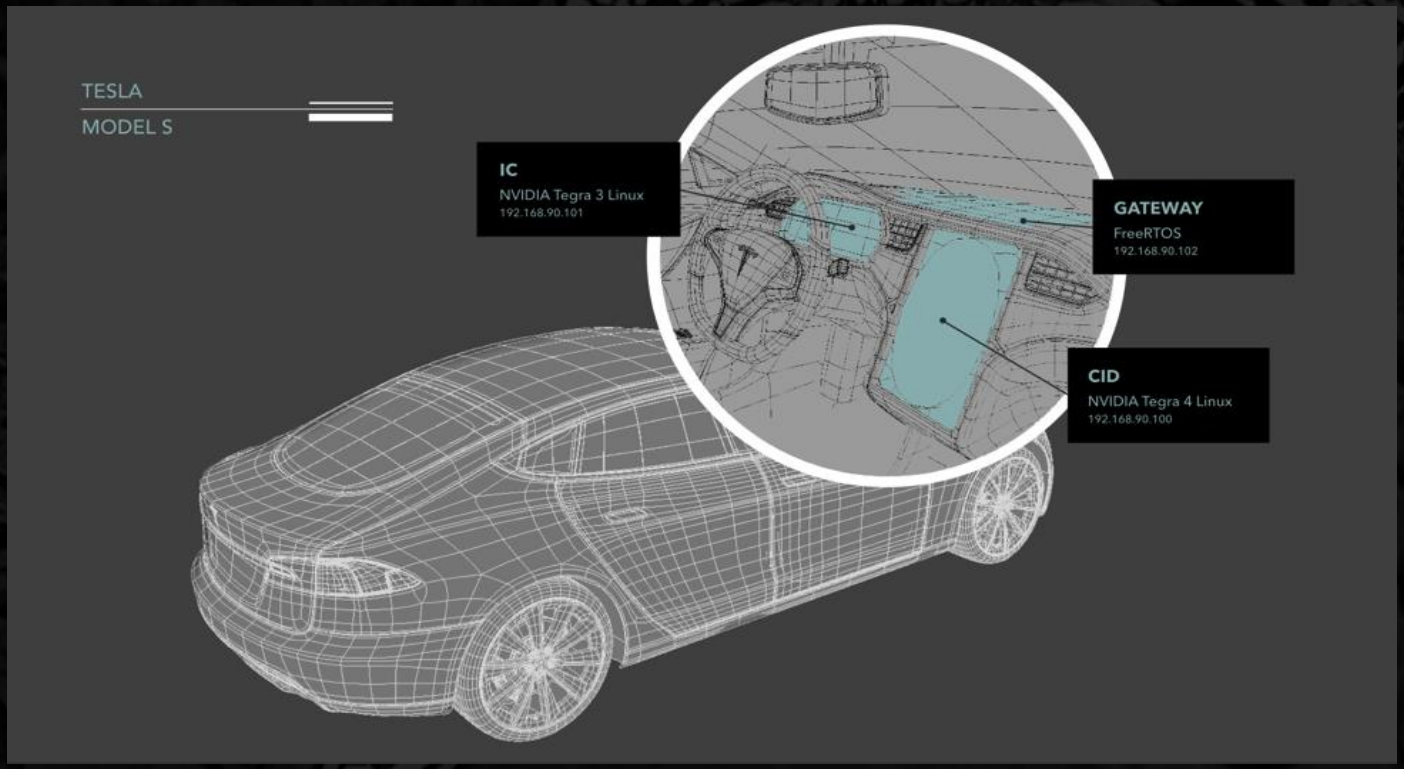
Assurance Issues for Tesla Motors

- Software upgrades highlight a broader concern with AVs: system security. Vehicles that are connected to each other, to infrastructure, or to the Internet are increasingly open to cyberattack
- GitHub
- Overzealous Owners
- People against Telsa and Elon Musk

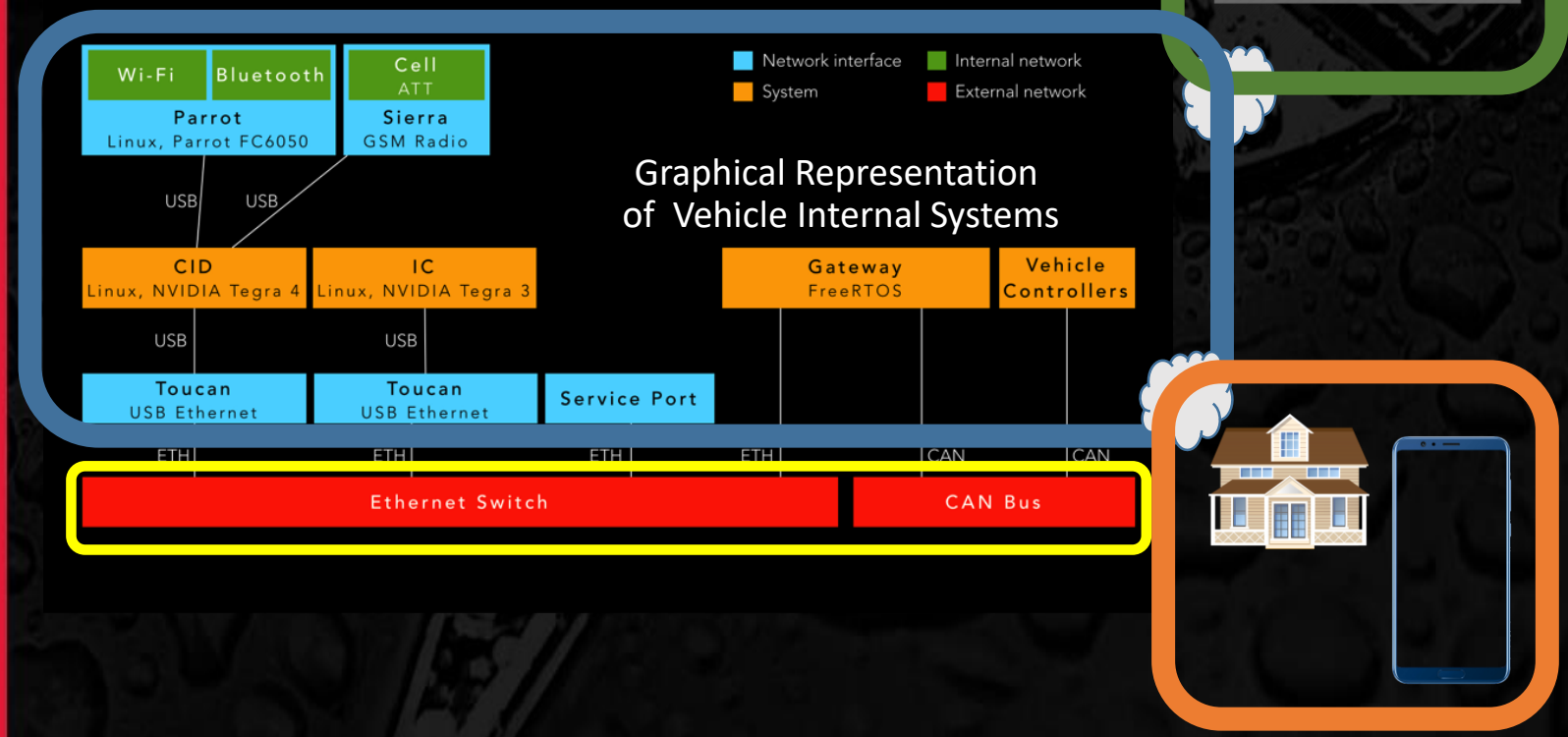
Assurance Issues for Tesla Motors Continued

- Vehicle configurations
  - Many of the vehicles are ‘preloaded’ with advanced features and when a user pays for the feature the ‘unlock’ is downloaded to the vehicle
    - As an example the vehicle sold as a 75kHw battery actually contains a 100kHw batter and users can pay for the 85kHw or 100kHw battery
    - All cars produces after 10/2016 have the capability for their AutoPilot offering and advanced AutoPilot when purchased.

TCB in the system
System Overview
Networking Scheme



Where should the TCB be in the Vehicle system



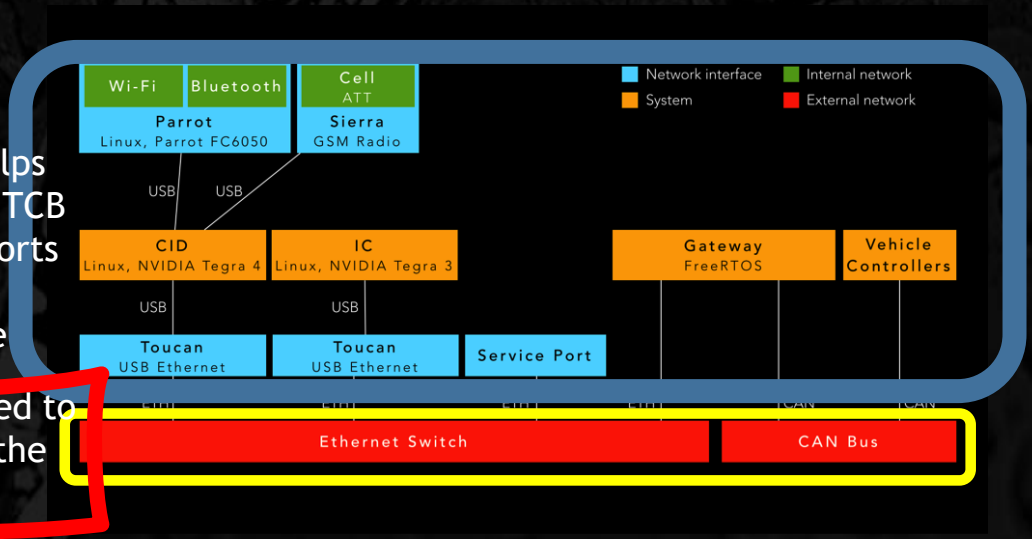
How does the system utilize minimization

Tesla vehicle configuration tries its best to implement minimization within the Trusted Computing Base.

The Blue Ring Highlighted below, contains the known components within the vehicle

The yellow Ring is another Trusted component that helps Insure minimization within TCB as though research these ports are isolated and must be configured specially for use

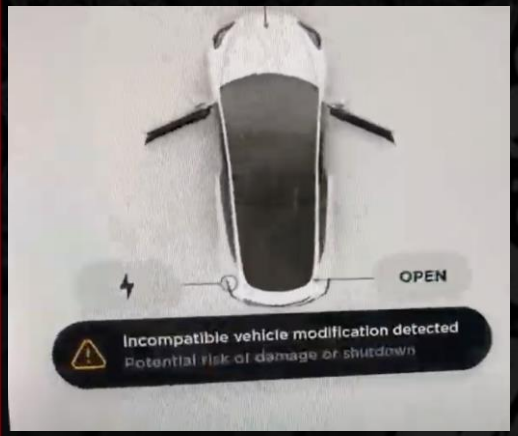
Cryptographic Keys required to Access the above BUS via the Ethernet switches.



Tesla Motors Biggest Vulnerability (in my option)

EASY ACCESS TO MCU (Media Control Unit)

People have designed 'Mods' to unlock performance features set buy the vehicle. This is a deterrent to assurance and safety and as of this the writing of this project the vulnerability still exists



Tesla Motors Answer to the Assurance Problem & Security Problem

What assurance techniques are applied

- Hack Reward of $1mm plus a free car is their vehicle is compromised
- Tesla has a bug reporting system in place to reward people who find and report vulnerabilities.
- Internal Penetration Testin
- Tesla does a good job of protecting their security. With the amount of vehicles on the road, they have had very little exposure to their fleet, but they have had exposure.
- Telsa holds hack events that provide up to $1mm and as most recently a Tesla Model 3 if someone could penetrate parts of the Tesla infrastructure.
- The internal infrastructure needs to be addressed

How can Tesla Improve the security of their system

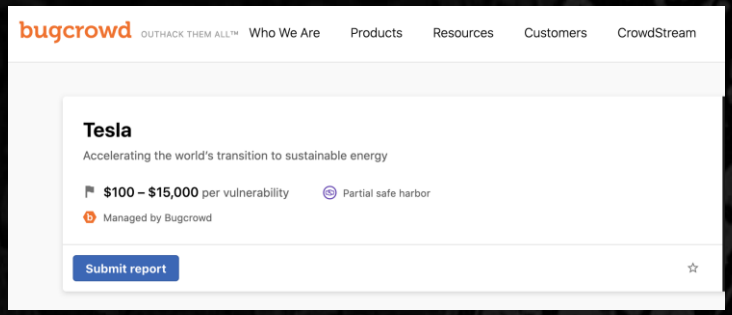
Target		Prize Amount	Master of Pwn Points	Additional Prize Options
Initial Vector	*Final Stage*			
Tuner, Wi-Fi, Bluetooth, or Modem	Infotainment	$250,000	25	Infotainment Root Persistence Add-on CAN Bus Add-on
Infotainment	VCSEC, Gateway, or Autopilot	$300,000	30	Infotainment Root Persistence Add-on Autopilot Root Persistence Add-on
Tuner, Wi-Fi, Bluetooth, or Modem	VCSEC, Gateway, or Autopilot	$400,000	40	Infotainment Root Persistence Add-on Autopilot Root Persistence Add-on

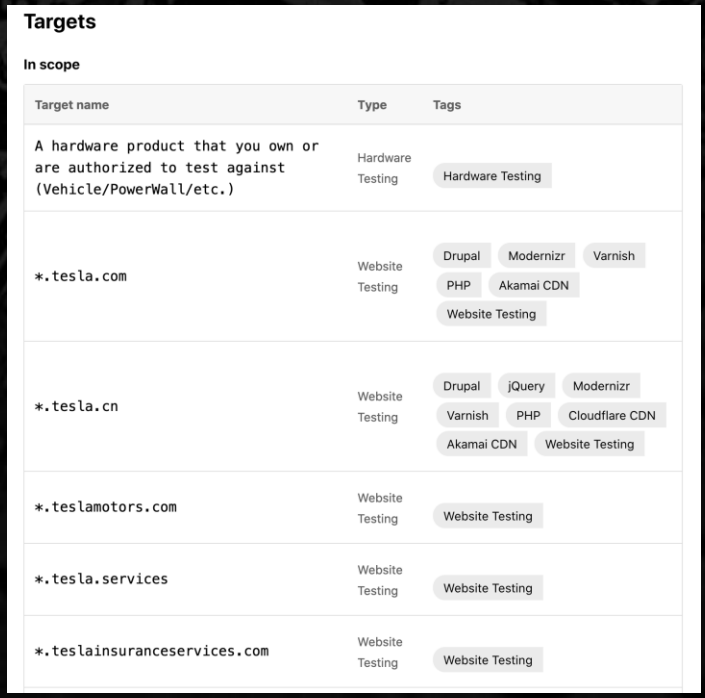
How can Tesla Improve the security of their system

Target		Prize Amount	Master of Pwn Points	Additional Prize Options
Initial Vector	*Option*			
Modem or Tuner	N/A	$60,000 USD	6	CAN Bus Add-on
Wi-Fi or Bluetooth	N/A	$60,000 USD	6	CAN Bus Add-on
Infotainment	N/A	$35,000 USD	3.5	N/A
	USB-based Attack	$35,000 USD	3.5	Infotainment Root Persistence Add-on CAN Bus Add-on
	Sandbox Escape	$100,000 USD	10	Infotainment Root Persistence Add-on CAN Bus Add-on
	Root/Kernel Escalation of Privilege	$100,000 USD	10	Infotainment Root Persistence Add-on CAN Bus Add-on
VCSEC, Gateway, or Autopilot	N/A	$200,000 USD	20	Vehicle Included Autopilot Root Persistence Add-on
Autopilot and Gateway *(Ethernet Attack Surface only)*	N/A	$100,000 USD	10	Vehicle Included Autopilot Root Persistence Add-on
Autopilot Denial-of-Service	N/A	$50,000 USD	5	N/A
Key Fobs or Phone-As-Key	N/A	$100,000 USD	10	Vehicle Included

Methods of attacks and how can Tesla Improve the security of their system

Example of Tesla's sub page of BugCrowd





Targets

In scope

Target name	Type	Tags
A hardware product that you own or are authorized to test against (Vehicle/PowerWall/etc.)	Hardware Testing	Hardware Testing
*.tesla.com	Website Testing	Drupal, Modernizr, Varnish, PHP, Akamai CDN, Website Testing
*.tesla.cn	Website Testing	Drupal, jQuery, Modernizr, Varnish, PHP, Cloudflare CDN, Akamai CDN, Website Testing
*.teslamotors.com	Website Testing	Website Testing
*.tesla.services	Website Testing	Website Testing
*.teslainsuranceservices.com	Website Testing	Website Testing

Methods of attacks and how can Tesla Improve the security of their system

PROVEN METHOD
Infotainment System
Manufacturer Mobile App
Manufacturer Website to Access Vehicle
Manufacturer Website to Inject APIs
Vehicle Wi-Fi
Vehicle Bluetooth
Onboard Diagnostic System (Ethernet / USB)
Owner Ego
MCU / Internal Computer Board access
KeyFob

Tesla is heavily invested in protecting their fleet and actively uses BugCrowd and they award for each vulnerability found and have routinely fixed the bugs quickly.

Conclusion

- With the limited information about Tesla combining information is tricky.
- The advanced features of the vehicle lead to a great number of possible methods to thwart the vehicle
- Thinking outside of the box, scouring Teslas current and prior job postings they have invested heavy in cryptography, technical architecture, pen testing, and advanced programmers.
- The company has a great financial investment to ensure assurance to their fleet, if the vehicles can we crippled via a attack, Teslas whole model will crumble.
- Their current design of the determined TCB and minimization assumptions proves Tesla is segmenting access vital areas of the network within the vehicle and overall infrastructure to access the vehicle which aims to prevent unauthorized access.

References

- https://spectrum.ieee.org/transportation/advanced-cars/6-key-connectivity-requirements-of-autonomous-driving
- https://www.nhtsa.gov/technology-innovation/automated-vehicles
- https://www.cio.com/article/3433931/tesla-the-data-company.html
- https://www.wired.com/2015/08/researchers-hacked-model-s-teslas-already/
- https://electrek.co/2020/06/10/tesla-hacker-unlocks-performance-upgrade-acceleration-boost/
- https://www.cnbc.com/2019/03/29/tesla-model-3-keeps-data-like-crash-videos-location-phone-contacts.html
- https://www.zdnet.com/article/tesla-starts-to-release-its-cars-open-source-linux-software-code/
- http://kernsec.org/files/lss2014/safford_tcb_integrity.pdf
- https://www.autoblog.com/2014/04/12/tesla-model-s-owners-hack-their-cars-find-ubuntu/?
- https://urgentcomm.com/2020/04/26/with-connected-cars-zero-trust-is-the-best-security-advice/
- https://www.tesmanian.com/blogs/tesmanian-blog/tesla-autopilot-os-vw-daimler-bmw
- https://www.bbc.com/news/technology-37426442
- https://hbr.org/2020/02/how-tesla-sets-itself-apart
- https://securityledger.com/2019/04/hackers-remotely-steer-tesla-model-s-using-autopilot-system/
- https://blog.lookout.com/hacking-a-tesla
- https://www.tesla.com/support/infotainment
- https://electrek.co/2020/08/27/tesla-hack-control-over-entire-fleet/
- https://www.carmagazine.co.uk/car-news/tech/tesla-fights-back-in-car-hacking-war/
- https://lexfridman.com/tesla-autopilot-miles-and-vehicles/
- https://electrek.co/2020/04/22/tesla-autopilot-data-3-billion-miles/#:~:text=Out%20of%20the%203%20billion,changes%2C%20according%20to%20Karpathy's%20presentation.
- https://electrek.co/2020/04/22/tesla-autopilot-data-3-billion-miles/#:~:text=Out%20of%20the%203%20billion,changes%2C%20according%20to%20Karpathy's%20presentation

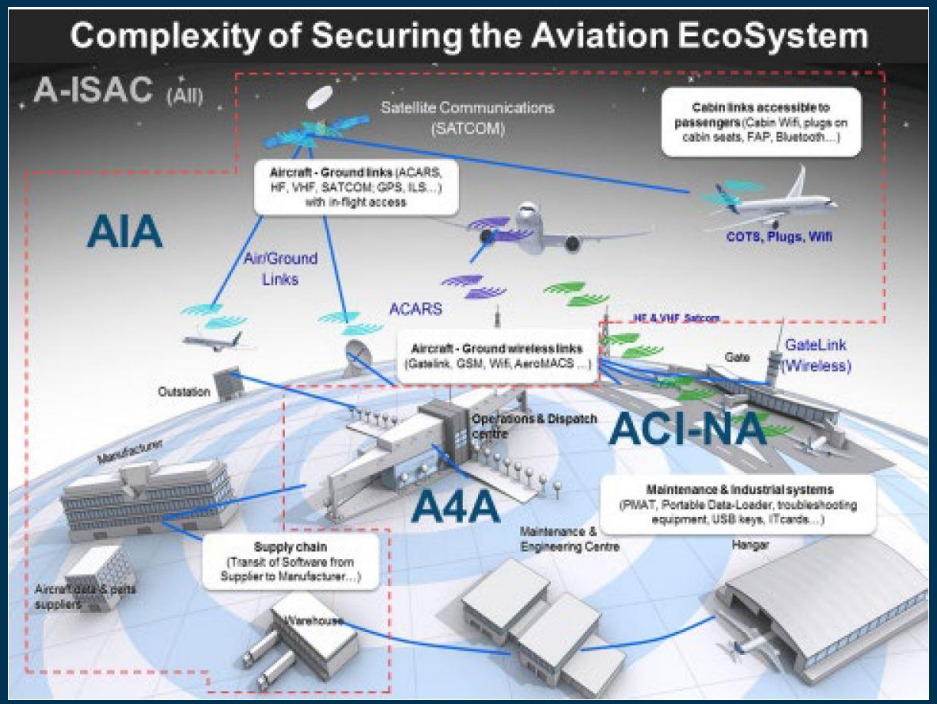
Avionics Ecosystem

-A Cybersecurity Perspective

Pratyush Prakhar
Dsci523
2468183206

What is avionics ?

Avionic Systems



- Avionic systems in the simplest of words are a class of electronic systems that are specifically designed and utilized in aviation.
- An amalgam of aviation hardware and management software that is responsible for the minutest of tasks in such a complex flight network.
- The ecosystem is just not constrained to the components on an airplane but also the ground. One can say it has developed now into an interconnected autonomous network of IoT devices.
- Commercial airliners, helicopters, military fighter jets, unmanned aerial vehicles (UAV), business jets, and spacecraft all use avionics in a different capacity. They range from engine controls to flight control systems to navigation, communications, and even performance monitors.
- The push towards drones and unmanned avionics push the cybersecurity dynamics towards an unchartered territory.

Why avionics?

Life Factor

An average of 4.7 billion people scheduled flights in 2019 which puts an average of 6 flights per traveller.

Commercial Factor

- Industry amounts to 5% of total US's GDP
- Global revenue for commercial airlines alone amounted to 838 billion dollars in 2019. The growth is exponential.

Travel

Still the fastest mode of transport till Hyperloop comes in !!

Global Impact

The very nature of operation in cross country domains makes cyber attacks a global issue.

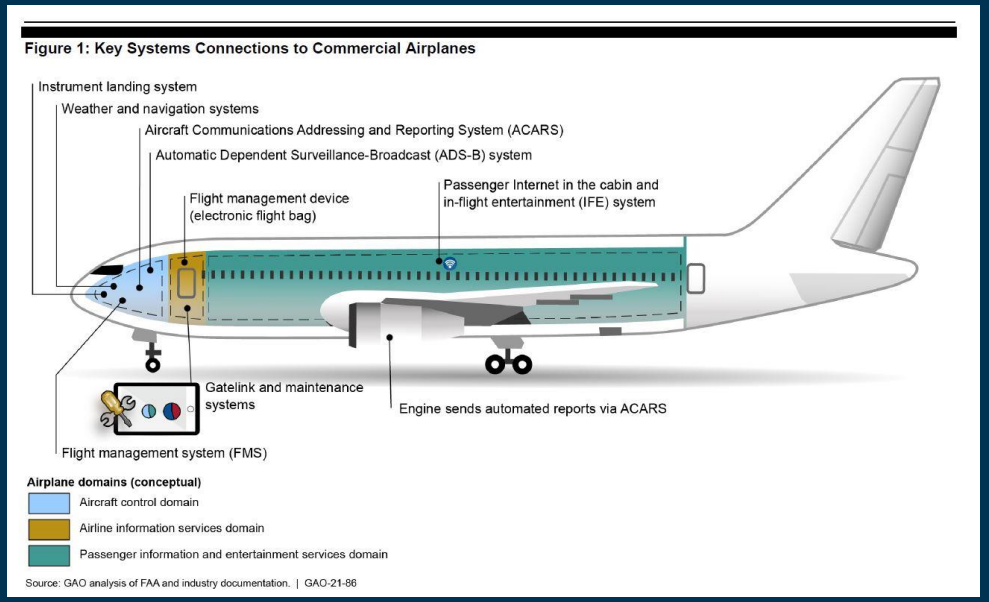
Military Aspect

- Recent technological development in weaponry employing Drones and UAVs make them prone to hostile takeovers.
- The cost of single units go as high as $ 123 million.

What comprises the Avionics Ecosystem?

Before Vs Now

- Earlier the ecosystem comprised of ATCs on the ground and Aircraft Control Domain consisting of SBAS, GBAS, ADS-B, and CPDLC on a plane. As every critical functionality was on the plane, the system was air-gapped and thus only required physical access controls.
- But due to the development in technology towards real-time functionality and for easiness of passengers, additional domains were added on.
  - Aircraft Control - Most critical domain responsible for safe operations of aircraft. Consists of critical avionics such as air traffic control, navigation controls, and plane controls. Separated from all other domains.
  - Airline Information Services – Provides services and connectivity between the 2 domains. Handled solely by plane's crew and consist of passenger and entertainment information, maintenance, etc. For secure operations, only allowed **to read data and report.** Can't issue any commands.
  - In-flight Entertainment – Responsible for all in-flight passenger services and management. Responsible for Wi-Fi connectivity, Entertainment systems, USB, etc. **Most prone to cyber-attacks.**



Connectivity

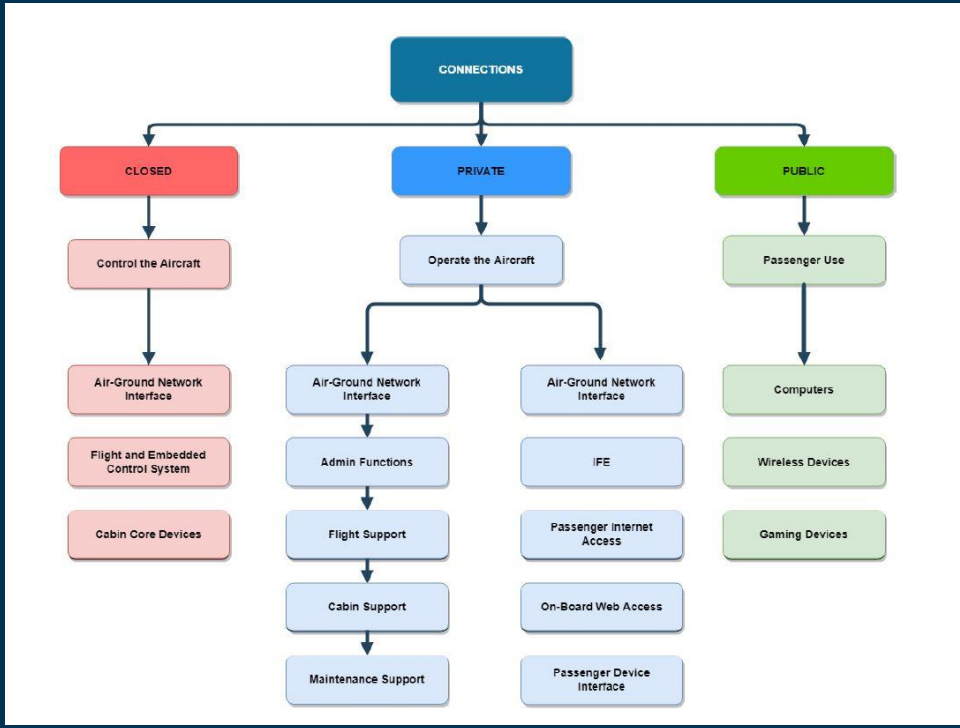
Several connections are provided by different internet service providers:

- **Air-Ground Datalink Service (ACARS)**: uses Air Trac Service providers and Airline Approved third party providers.
- **Airport Network (GateLink):** uses the Airline-Approved third-party providers.
- **Air-Ground Broadband Network (INMARSAT)**: uses Airline-Approved third-party providers

Several others allow for aircraft information interconnection:

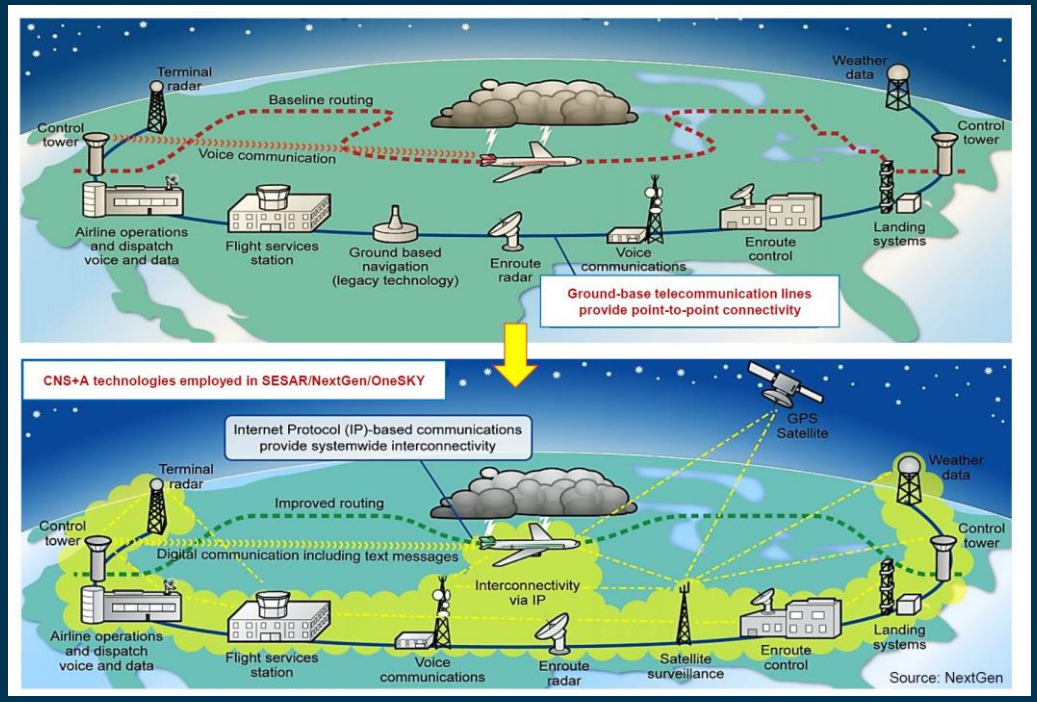
- **Control the Aircraft:** it is in the Aircraft Control Domain. It uses VHF/H-F/SatCom.
- **Operate the Aircraft**: it contains the Airline Info Services and the Passengers Information and Entertainment Services Domain. It uses Wireless LAN and a Broadband/Cellular network.
- **Passenger Use**: it is in the Passenger-Owned Devices Domain.

Every connection can be turned into a potential threat vector.



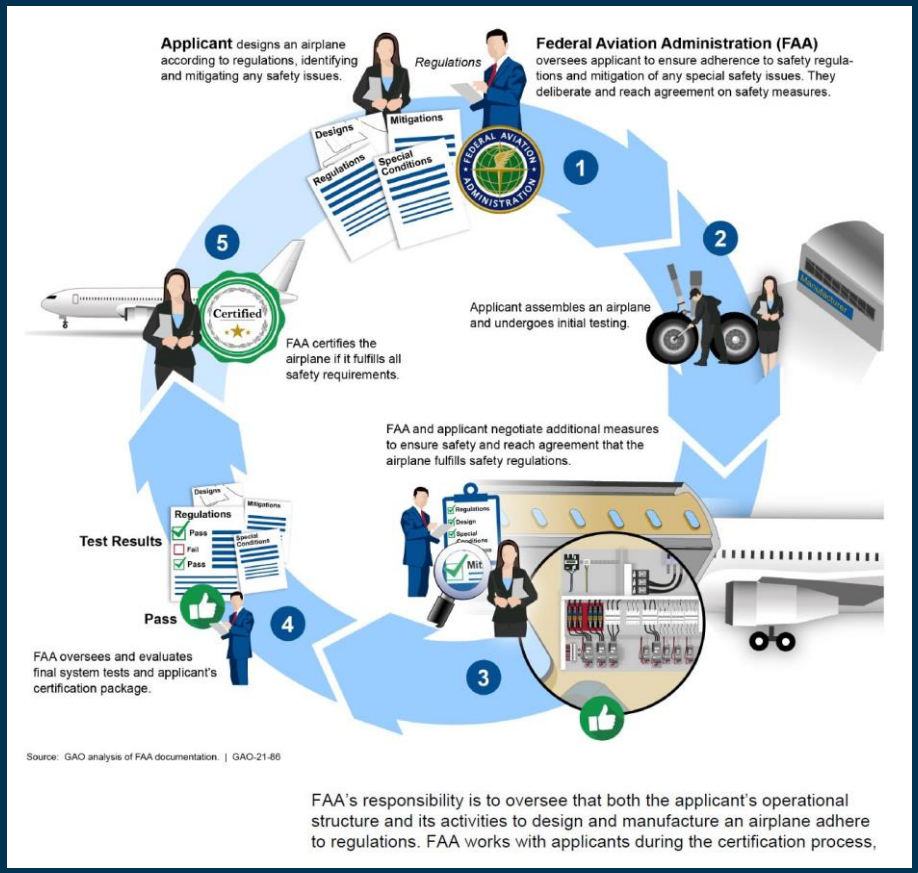
The Future

- The avionics industry has gone through a rigorous evolution cycle in the last 50 years or so. They transitioned from individual task controlling systems to an integrated Intelligent Transport System (ITS). The projects such as NextGen ATS and Single European Sky ATM Research (SESAR) are some initiatives in play.
- To accommodate the growing air travel, the avionic system becomes more sophisticated by the day. They are now an intricate system of navigation, communication, and surveillance systems/sensors for airplane's automated applications and multi-platform networking. The integration with legacy federated technologies gives rise to insecure paths.
- Also, more and more aviation systems are based on commercial off-the-shelf products and solutions owing to the commercial airplane boom. The increased reliance on vulnerable operating systems such as Linux, Windows, and wireless protocols.
- The result of moving forward is moving backward, as the widespread adoption of these technologies has unleashed increased vulnerabilities and greater risk to the avionics systems. There is also the issue of interoperability due to the nature of policies in various nations.



Life Cycle of Commercial Airlines

- The figure shows the certification of airworthiness of a commercial flight by the Federal Aviation Administration (FAA).
- FAA is responsible for overseeing both operational structure and manufacture of the plane is according to regulations and policy in place.
- For cybersecurity concerns, certain **'Special Conditions'** are included as they are not part of the above process. This comprises of a risk-based evaluation and management procedure for the airplanes utilizing e-services.
- After complying with all the regulations only, the applicant can create the prototype of the airplane and goes through this evaluation cycle to get certified for production.
- We will see in the next section that these regulations are highly detailed given the critical nature of the product.



Current Regulatory Standards

International Standards and Guidelines

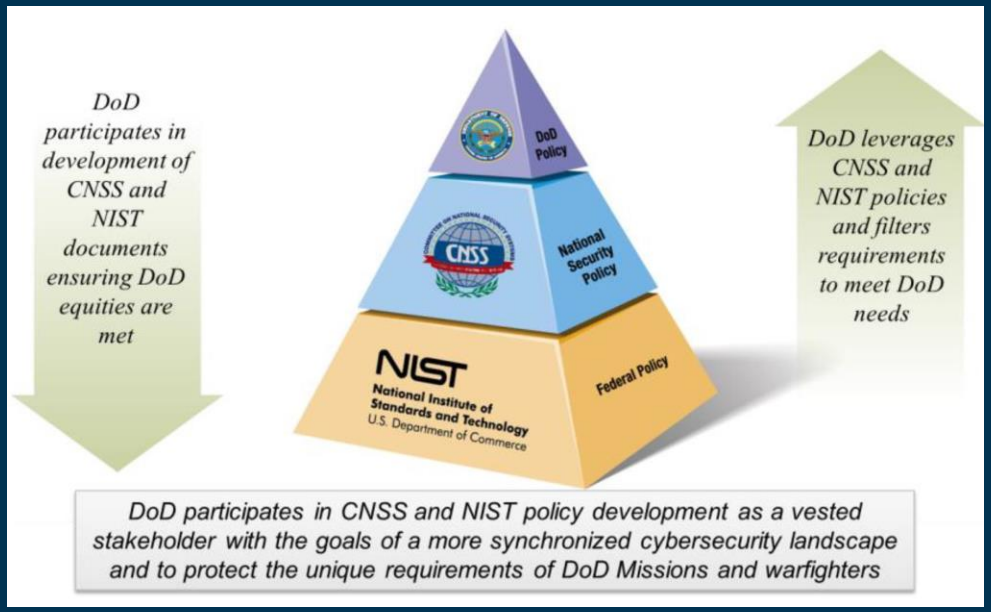
- International organizations such as the International Civil Aviation Organization (ICAO) and Radio Technical Commission for Aeronautics (RTCA) are responsible for setting certain airworthiness and security standards which are mandatory to be followed.

- The SC216 Committee on Security under ICAO defines standards for the security airworthiness. Three of them stand out the most and are considered parallel. They must be included in various stages of an airplane's manufacturing process. They are

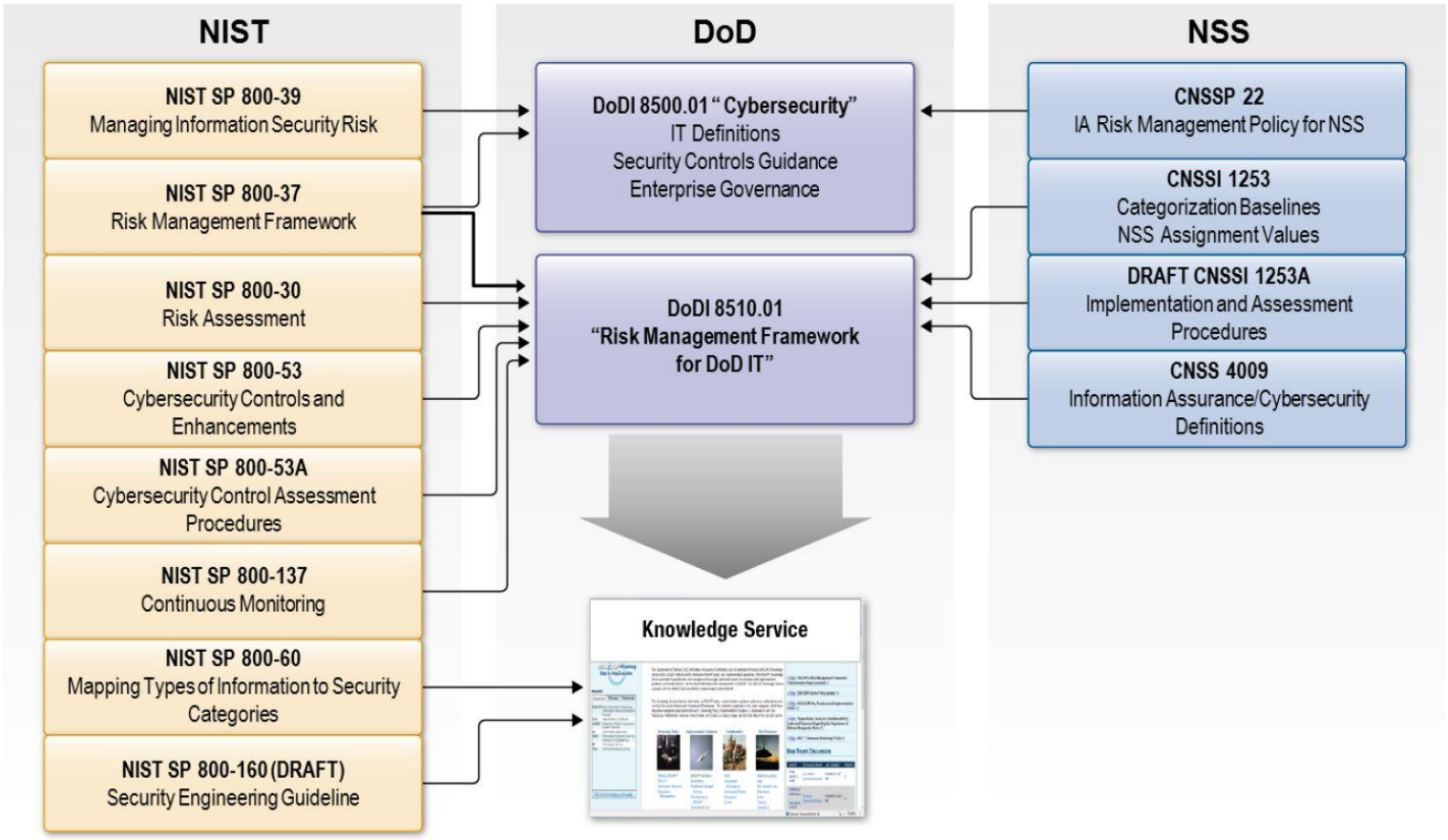
    - DO-326A: (RTCA DO-326A, 2014) - guidelines for aircraft certification: how to handle the threat of intentional unauthorized electronic interaction to aircraft safety, including handling malware, distorted data, or penetration of aircraft systems. It deals with activities that should be conducted in the operation and maintenance of the aircraft about information security threats.
    - DO-356A: (RTCA DO-356A, 2017) – Connected to DO-326A and provides guidelines of security mechanisms from project start to certification.
    - DO-355: (RTCA DO-355, 2014) – Companion document to DO-326A referring to situations where aircraft safety might be affected by the operation and maintenance of aircraft security threats.

- These three security standards are supported by other '**safety standards**' defined in **DO-178C, DO-330, DO-333, and DO-254.**

National Regulations



- Every country has its players who decide the cybersecurity standards for the aviation industry. In the US we can see them as FAA, DOD, and DHS.
- If we consider the US, the focus on cybersecurity in the aviation industry came under President Obama in 2013. The initial investment was around $50 million. Several Acts and guidelines have popped up since then.•
  - Government Accountability Office (GAO) Report – States that the FAA is responsible for NAS and looming threats to air transport. Recommendations for reducing the risk are published every year.
  - Cyber AIR Act – Established in 2017, it states full disclosure of air controls and management by manufacturers to FAA, so appropriate regulation changes can take place regarding cybersecurity.
  - The Federal Information Security Modernization Act (FISMA) – Protection against cyber threats. Applies to DOT and FAA in the aviation sector.
  - The 'Special Condition' clause is also published by the FAA for cybersecurity standards.
- The various regulations leverage the NIST SP-800 guidelines combined with CNSS policies. DOD combines them into a cybersecurity framework as shown in the next slide.


NIST
NIST SP 800-39
Managing Information Security Risk
NIST SP 800-37
Risk Management Framework
NIST SP 800-30
Risk Assessment
NIST SP 800-53
Cybersecurity Controls and Enhancements
NIST SP 800-53A
Cybersecurity Control Assessment Procedures
NIST SP 800-137
Continuous Monitoring
NIST SP 800-60
Mapping Types of Information to Security Categories
NIST SP 800-160 (DRAFT)
Security Engineering Guideline
DoD
DoDI 8500.01 " Cybersecurity"
IT Definitions
Security Controls Guidance
Enterprise Governance
DoDI 8510.01
"Risk Management Framework for DoD IT"
Knowledge Service
NSS
CNSSP 22
IA Risk Management Policy for NSS
CNSSI 1253
Categorization Baselines
NSS Assignment Values
DRAFT CNSSI 1253A
Implementation and Assessment Procedures
CNSS 4009
Information Assurance/Cybersecurity Definitions

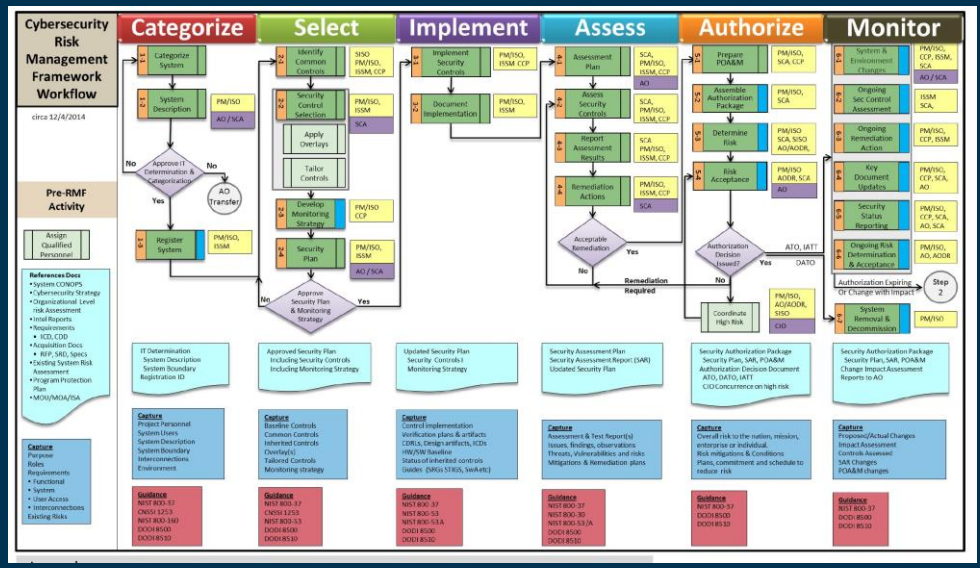
Cyber Security Incidents

- Although a sophisticated framework of evaluation is in place but still, there have been multiple incidents of cyber-attack in the aviation environment. The very complex nature is the cause. The principle of modularization and interoperability fails in such a framework giving adversary multiple attack paths.
- For instance, in 2016, the Director of the European Aviation Safety Agency (EASA) revealed that aviation systems were subject to an average of 1,000 attacks each month. Let's look at some prominent ones.
  - In 2006 an internet attack forced the U.S. FAA to shut down several of its air traffic control systems in Alaska.
  - In 2009 a report published in the U.S. by the Inspector General of DOT revealed that increasing use of web applications linked to FAA systems exposed the ATS to potential cyber-attacks through access control vulnerabilities.
  - In 2013 an attack at Istanbul airport caused severe problems in the passport control system, which resulted in delays of many flights. That year 75 airports in the US were targeted, possibly by malicious hacking and phishing.
  - In June 2017, Boryspil International Airport, Ukraine's largest terminal, came under cyber attack as part of a comprehensive cyber action against government infrastructures. The attack disabled the airport's computers and departure boards.
  - The disappearance of Malaysian Airlines Flight MH370 remains the biggest mystery of all. The most probable theory suggests a cyber attack on the plane's flight controls and turning off the transponder.

RMF & Cyber Security Assessment

Risk Management Framework (RMF)

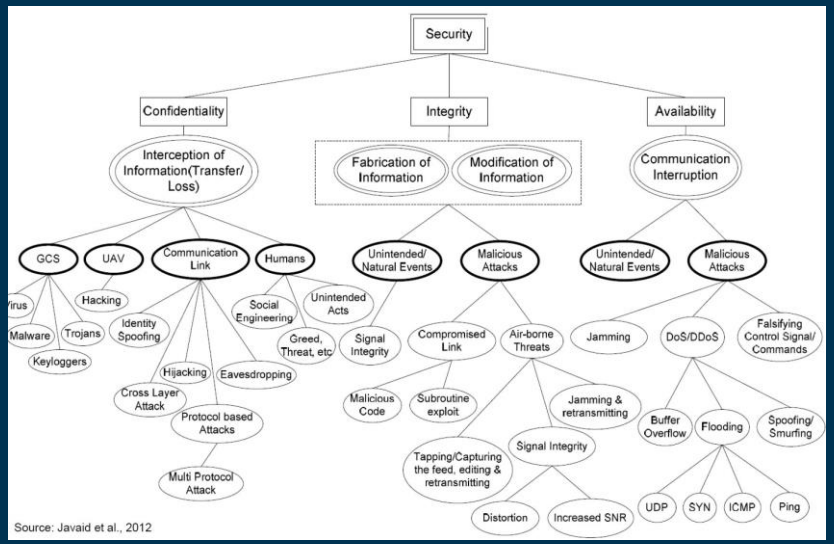
- The rigorous evaluation process which ultimately leads to the certification of the airplane's components (software and hardware) has a 'Risk Management Framework' at its core. It consists of 6 phases.
  - Categorize IT assets and equipment. This also involves setting the security boundaries and differentiating domains based on the description of the system.
  - Select the appropriate security controls and produce a security and monitoring strategy.
  - Implement the controls and the security strategy. HW/SW baseline are implemented, and verification plans are put in place.
  - Assessment is the most important phase of all. It involves threat assessment and management using various techniques and defining attack vectors and path.
  - Authorize phase is the calculation of risk that the various factors pose to the various involved entities. Also, a plan or sometimes guidelines are put in place to avoid them further.
  - Monitor is the last phase where we keep track of implemented security controls and changes. We also see the impact of the last iterations. Then the cycle starts again.



Threat Modelling

The fourth phase of RMF is security assessment to find out the vulnerabilities in the airplane's components and suggest required mitigations. Various **Tree Analysis** techniques are the most used.

- **Fault Tree Analysis (FTA)** involves Boolean logics to represent the interrelation of failure of various subsystems. It aims to find the requirements (most probable cause of failing) using such analysis. Holds similarity to Attack Trees.
- **Cyber Threat Trees** are a superset of Fault and Attack Trees and are based on multiple-valued algebras over a finite set. This allows to reduce such maps and allow more complicated interactions to be modeled.
- **Attack-Defense Trees** to find minimal cost attack paths



STRIDE Model

STRIDE Model developed by Microsoft is a threat modeling technique. It divides the vulnerabilities into 6 categories according to effects. These categories are a representation of the most crucial security aspects. These 6 parameters should be tested for each component and even for interconnections (flows). They can be defined as

- Spoofing Identity – **Authentication**
- Tampering with Data - **Integrity**
- Repudiation – **Non-Repudiation**
- Information disclosure - **Confidentiality**
- Denial of Service - **Availability**
- Elevation of Privileges – **Authorization**

Lastly, we can see which category is most likely to cause functionality failure and you apply remediation accordingly. Here it is DOS.

Threat 1: [16]

Threat ID	1
Threat Description	Attacking Critical Systems via In-Flight Entertainment Systems [16]
Threat Target	To gain control of the IFE system, or potentially crash an aircraft while in flight.
Attack Techniques	Search for security holes, install software before flight take off, and use established vulnerabilities.
Countermeasures	Secure memory, secure communication, and secure run-time environments between internal aircraft components.

Threats ID	Spoofing Identity	Tampering with data	Repudiation	Information Disclosure	Denial of Service	Elevation of Privilege
1					✓	✓
2			✓		✓	
3	✓	✓	✓	✓	✓	
4	✓	✓		✓		✓

Table (1): STRIDE threat classification for analysis

DREAD Model

- Another threat modeling model that compliments the STRIDE framework. Based on 5 threat dimensions and analyze how each affects the functionality. The final risk value is the average of each.
  - Damage potential.
  - Reproducibility.
  - Exploitability.
  - Affected users.
  - Discoverability.
- As there is no standard scale, DREAD helps measure the probability of risk using above 5 values each ranging from 1-3.
- The various final risk categories are
  - High Risk = 12 to 15
  - Medium Risk = 8 to 11
  - Low risk = 5 to 7

Risk DREAD = D+R+E+A+D / 5

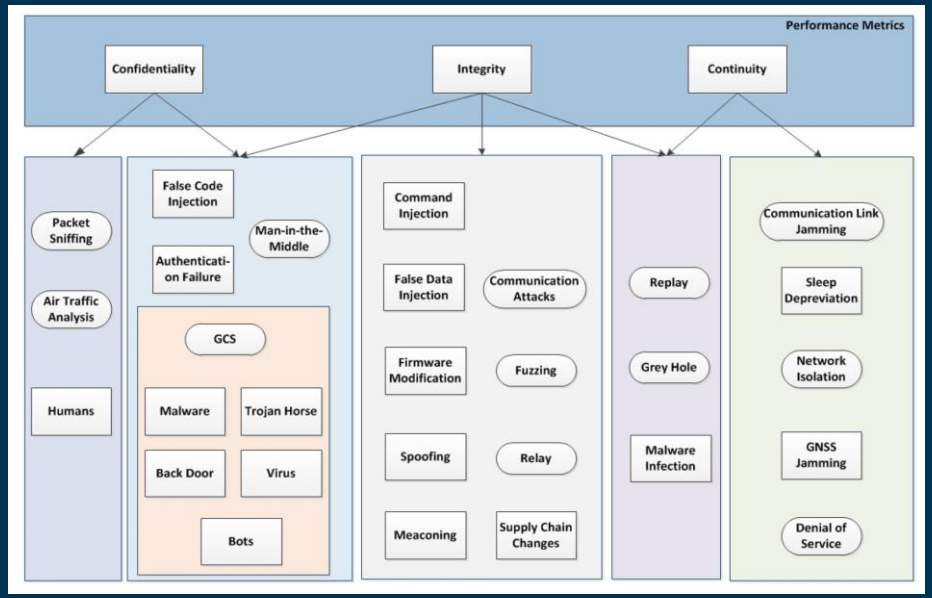
Threats ID	D	R	E	A	D	Total	Rating
Threat 1	3	3	1	2	1	10	Second: Medium Risk
Threat 2	3	3	2	3	1	12	First: High Risk
Threat 3	3	2	1	2	1	9	Second: Medium Risk
Threat 4	2	1	2	1	1	7	Last: Low Risk

Table 2: DREAD rating result for all threats

Known Vulnerabilities

The various aviation-related cyber incidents and threat modeling have shed the light on some known vulnerabilities. Some of them are

- Misinformation due to spoofing and jamming signals. Seen in positioning data on NextGen network due to lack of mutual authentication.
- Boeing 777 was susceptible to where system controls could be tampered with using a small radio device.
- Exploitation of IFE by injection through USBs and Ethernet ports. If a path to the airplane domain can be found, then it develops into an attack.
- Malware infection to the monitoring or navigation systems.



Key Issues

Cyber Security Policy

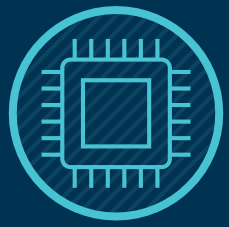
Multiple players in multi ecosystems

Commercial – AIA, ACI, A4A

Military – NIST, NSS, DoD

Drive Force

Economics have overtaken security in a commercial airline industry. Every resource dedicated to comfort.



Growth in Vulnerable Hardware Components

The integration of IoT sensors and entertainment system with core avionic system.

Malicious Software

Although very hard task to obtain access to in flight systems. But again 99% secure is still insecure. Low standard commercial products add fuel to fire. Plus Huge.

NextGen Technology

Projects like Single European Sky ATM Research (SESAR) and the Next Generation Air Systems (NextGen) are modernizing Air Traffic Management, but increased connectivity and information exchange comes increased cyber risk.

Solutions

Research

- It is a hot topic in today's industry. As the systems are becoming more complex and sophisticated by the day, there is a requirement for better security assessment techniques.
- Two of them are
  - **Hazard Space Analysis (HARA)**
  - Implementation of STRIDE approach and Hazard Analysis and Risk Assessment **(SAHARA)**

Policy Change

- Need for a unified policy rather than international standards. Would offer better interoperability.
- Setting a higher standard of security systems involved. Commercial on-flight systems should have higher accountability
- Proper Translation to the actual mechanism with high assurance sue to the critical data involved.
- One such example is the introduction of a more secure Multicore certification by EASA & FAA. It sets certain guidelines to the security standard of multicore chipsets used in avionic systems. This takes tool off from using single core chips.

Risk Management Framework

- Big players in this sector such as DoD and AIA both push for higher cyber security risk management frameworks. This provides high assurance of implementation of cyber security policy concerning aviation with a feedback mechanism.
- Certain technologies and principles that can be implemented to provide better security are
  - Distributed Security - **Modularization**
  - Security Kernel Separation
  - Utilizing secure principles in Integrated Modular Avionics such as **Abstraction and TCB.**

Thank You

Reference

- https://www.aviationtoday.com/2020/02/28/easa-and-faa-to-issue-further-guidance-on-multicore-certification-this-year/
- http://fhr.nuc.berkeley.edu/wp-content/uploads/2017/04/UCB-TH-17-001-Cybersecurity-in-Civilian-Aviation.pdf
- https://www.aia-aerospace.org/wp-content/uploads/2019/10/AIA-Civil-Aviation-Cybersecurity-Recommendations-Report-2019-Final-1.pdf
- https://commons.erau.edu/cgi/viewcontent.cgi?article=1438&context=edt
- https://law.fsu.edu/sites/g/files/upcbnu1581/files/JTLP/2019-2020/%2804%29%20JTLPv29_Feldman-Gross-Article_102820.pdf
- https://commons.erau.edu/cgi/viewcontent.cgi?article=1519&context=edt
- https://www.gao.gov/assets/720/710095.pdf