



# Security-blueprint

## AGL Security blueprint Documentation

**Version 5.0.0**  
**January 2018**

# Table of Contents

---

Introduction	1.1
Revisions	1.2
Part 1 - Hardware	1.3
Part 2 - Secure Boot	1.4
Image	1.4.1
Communication modes	1.4.2
Consoles	1.4.3
Part 3 - Hypervisor	1.5
Part 4 - Kernel	1.6
General	1.6.1
Memory	1.6.2
Consoles	1.6.3
Debug	1.6.4
File Systems	1.6.5
Part 5 - Platform	1.7
Mandatory Access Control	1.7.1
SystemD	1.7.2
SystemBus	1.7.3
System services and daemons	1.7.4
App Framework	1.7.5
Utilities	1.7.6
Users	1.7.7
Part 6 - Application	1.8
Installation	1.8.1
Privilege management	1.8.2
Signature	1.8.3
Services	1.8.4
Part 7 - Connectivity	1.9
Bus and connectors	1.9.1
Wireless	1.9.2
Cloud	1.9.3
Part 8 - Update (OTA)	1.10
FOTA	1.10.1
SOTA	1.10.2
Part 9 - Secure development	1.11
Annexes	1.12
All config notes	1.12.1
All todo notes	1.12.2

# Introduction

---

This document presents the different attacks that can be envisaged on a recent car in order to be able to create a set of tests verifying the security of Automotive Grade Linux (AGL). The more general utility behind this document is to protect the manufacturers, customers and third party from potential financial and information loss. This document is firstly based on the existing security-blueprint.

**For security to be effective, the concepts must be simple. And by default, anything that is not allowed is forbidden.**

We will cover topics starting from the lowest level (*Hardware*) up to the highest levels (*Connectivity* and *Application*). We will move quickly on *Hardware* and *Connectivity* because this is not supported at our level. Solutions of connectivity problems concern updates and secured settings while hardware securing is related to the manufacturers.

The document is filled with tags to easily identify important points:

- The *config* tag quickly identifies the configurations and the recommendations to take.

- The *note* tag allows you to notify some additional details.

- The *todo* tag shows the possible improvements.

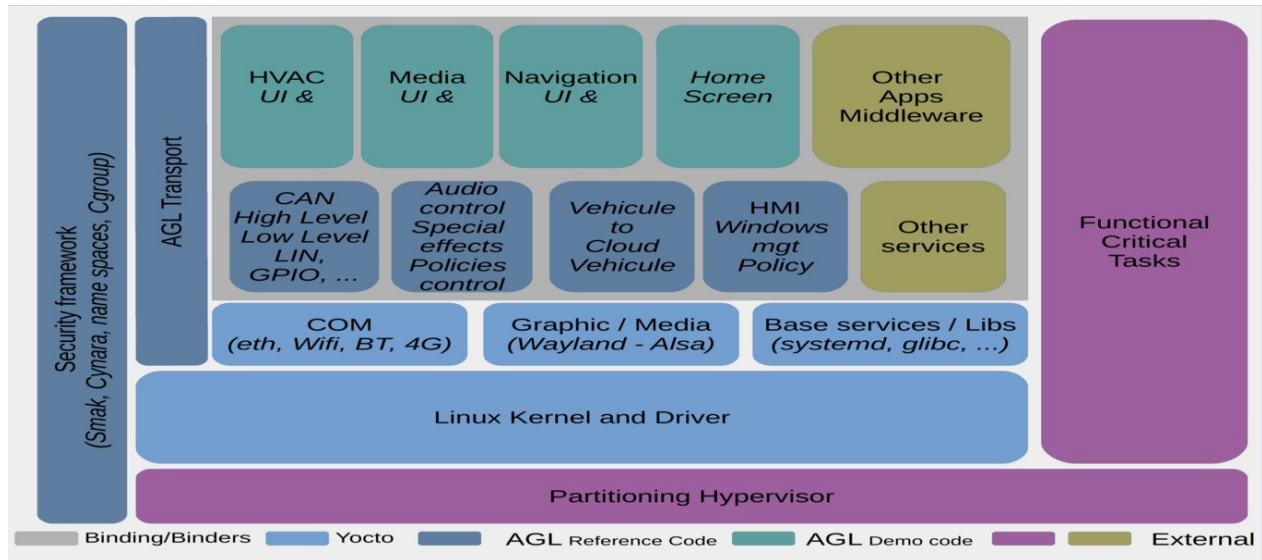
In annexes of this document, you can find all the *config* and *todo* notes.

## Hardening term

The term Hardening refers to the tools, techniques and processes required in order to reduce the attack surface on an embedded system, such as an embedded control unit (**ECU**) or other managed devices. The target for all hardening activities is to prevent the execution of invalid binaries on the device, and to prevent copying of security related data from the device.

## AGL security overview

AGL roots are based on security concepts. Those concepts are implemented by the security framework as shown in this picture:



## Acronyms and Abbreviations

The following table lists the strongest terms utilized within all this document.

Acronyms or Abbreviations	Description
AGL	<b>A</b> utomotive <b>G</b> rade <b>L</b> inux
ECU	<b>E</b> lectronic <b>C</b> ontrol <b>U</b> nit

## References

---

- [security-blueprint](#).
  - [http:// docs.automotivelinux.org/docs/architecture/en/dev/reference/security/01-overview.html](http://docs.automotivelinux.org/docs/architecture/en/dev/reference/security/01-overview.html)
- [2017] - [kernel security](#).
  - <https:// www.kernel.org/doc/Documentation/security/>
- [2017] - [Systemd integration and user management](#).
  - <http:// iot.bzh/download/public/2017/AMM-Dresden/AGL-systemd.pdf>
- [2017] - [AGL - Application Framework Documentation](#).
  - <http:// iot.bzh/download/public/2017/SDK/AppFw-Documentation-v3.1.pdf>
- [2017] - [Improving Vehicle Cybersecurity](#).
  - [https:// access.atis.org/apps/group\\_public/download.php/35648/ATIS-I-0000059.pdf](https:// access.atis.org/apps/group_public/download.php/35648/ATIS-I-0000059.pdf)
- [2016] - [AGL framework overview](#).
  - [http:// docs.automotivelinux.org/docs/apis\\_services/en/dev/reference/af-main/0-introduction.html](http:// docs.automotivelinux.org/docs/apis_services/en/dev/reference/af-main/0-introduction.html)
- [2016] - [SecureBoot-SecureSoftwareUpdates](#).
  - <http:// iot.bzh/download/public/2016/publications/SecureBoot-SecureSoftwareUpdates.pdf>
- [2016] - [Linux Automotive Security](#).
  - <http:// iot.bzh/download/public/2016/security/Linux-Automotive-Security-v10.pdf>
- [2016] - [Automotive Security Best Practices](#).
  - <https:// www.mcafee.com/it/resources/white-papers/wp-automotive-security.pdf>
- [2016] - [Gattacking Bluetooth Smart Devices](#).
  - <http:// gattack.io/whitepaper.pdf>
- [2015] - [Comprehensive Experimental Analysis of Automotive Attack Surfaces](#).
  - <http:// www.cs.wayne.edu/fengwei/15fa-csc6991/slides/8-CarHackingUsenixSecurity.pdf>
- [2015] - [Security in Automotive Bus Systems](#).
  - <http:// citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.728&rep=rep1&type=pdf>
- [2014] - [IOActive Remote Attack Surface](#).
  - [https:// www.ioactive.com/pdfs/IOActive\\_Remote\\_Attack\\_Surfaces.pdf](https:// www.ioactive.com/pdfs/IOActive_Remote_Attack_Surfaces.pdf)
- [2011] - [A practical attack against GPRS/EDGE/UMTS/HSPA mobile data communications](#).
  - [https:// media.blackhat.com/bh-dc-11/Perez-Pico/BlackHat\\_DC\\_2011\\_Perez-Pico\\_Mobile\\_Attacks-wp.pdf](https:// media.blackhat.com/bh-dc-11/Perez-Pico/BlackHat_DC_2011_Perez-Pico_Mobile_Attacks-wp.pdf)
- [2011] - [Comprehensive Experimental Analyses of Automotive Attack Surfaces](#).
  - <http:// www.autosec.org/pubs/cars-usenixsec2011.pdf>
- [2010] - [Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars](#).
  - <https:// eprint.iacr.org/2010/332.pdf>
- [2010] - [Wifi attacks wep wpa](#).
  - <https:// matthieu.io/dl/wifi-attacks-wep-wpa.pdf>
- [2008] - [SMACK](#).
  - [http:// schaufler-ca.com/yahoo\\_site\\_admin/assets/docs/SmackWhitePaper.257153003.pdf](http:// schaufler-ca.com/yahoo_site_admin/assets/docs/SmackWhitePaper.257153003.pdf)

## Document revisions

---

<i>Meta</i>	<i>Data</i>
<b>Title</b>	Security-blueprint
<b>Description</b>	This document deals with everything related to the safety of connected cars linked to the AGL project.
<b>Keywords</b>	AGL, Security, Blueprint, Iotbzh
<b>Language</b>	English
<b>Published</b>	Published January 2018 as an electronic book.
<b>Updated</b>	Fri Jan 12 2018 16:46:36 GMT+0100 (CET)
<b>Collection</b>	Open-source

---

<b>Date</b>	<b>Version</b>	<b>Designation</b>	<b>Author</b>
7 Jul 2017	-	sec-blueprint	<a href="#">Git history</a>
6 Dec 2017	5.0.0	EE.rc3 release - chapters reordering and add new parts : 3, 5, 6, 8, 9	Vincent Nieutin [Iot.bzh]

# Part 1 - Hardware

## Abstract

You will find in this first part everything that concerns the hardware security. The goal is to protect system against all attacks that are trying to gain additional privileges by recovering and/or changing cryptographic keys in order to alter the integrity of the boot. We should also prevent hardware modifications in order to achieve this goal. We will expose below some examples of possible configurations.

## Acronyms and Abbreviations

The following table lists the terms utilized within this part of the document.

Acronyms or Abbreviations	Description
<i>HSM</i>	<b>H</b> ardware <b>S</b> ecurity <b>M</b> odule
<i>NVM</i>	<b>N</b> on- <b>V</b> olatile <b>M</b> emory
<i>SHE</i>	<b>S</b> ecure <b>H</b> ardware <b>E</b> xtensions

## Integrity

The board must store hardcoded cryptographic keys in order to verify among others the *integrity* of the *bootloader*. Manufacturers can use **HSM** and **SHE** to enhance the security of their board.

Domain	Object	Recommendations
Hardware-Integrity-1	Bootloader	Must control bootloader integrity.
Hardware-Integrity-2	Board	Must use a HSM.
Hardware-Integrity-3	RTC	Must not be alterable.

## Certificates

Domain	Object	Recommendations
Hardware-Certificate-1	System	Shall allow storing dedicated certificates.
Hardware-Certificate-2	ECU	The ECU must verify the certification authority hierarchy.
Hardware-Certificate-3	System	Allow the modification of certificates only if the source can be authenticated by a certificate already stored or in the higher levels of the chain of trust.

## Memory

Domain	Object	Recommendations
Hardware-Memory-1	ECU	The ECU shall never expose the unencrypted key in RAM when using cryptographic keys.
Hardware-Memory-2	Bootloader	Internal NVM only
Hardware-Module-3	-	HSM must be used to secure keys.



## Part 2 - Secure boot

### Abstract

Domain	Improvement
Boot-Abstract-1	More generic and add examples (The chain of trust).

**Boot Hardening:** Steps/requirements to configure the boot sequence, in order to restrict the device from executing anything other than the approved software image.

In this part, we will see a series of settings that will allow us to improve security during boot phase. For the purposes of reference and explanation, we are providing guidance on how to configure an embedded device that runs with a 3.10.17 Linux kernel. If the integrity is not checked or if a critical error occurs, the system must boot on a very stable backup image.

**Requirements:** These requirements must be met even if an alternative version of the Linux kernel is chosen.

**Recommendations:** Detailed best practices that should be applied in order to secure a device. Although they are not currently listed as hard requirements, they may be upgraded to requirements status in the future. In addition, specific operators may change some of these recommendations into requirements based on their specific needs and objectives.

Domain	Improvement
Boot-Abstract-1	Review the definition of the "boot loader".

**Boot loader:** The boot loader consists of the Primary boot loader residing in **OTP** memory, sbboot, U-Boot and Secure loader residing in external flash (NAND or SPI/NOR flash memory). The CPU on power on or reset executes the primary boot loader. The **OTP** primary boot loader makes the necessary initial system configuration and then loads the secondary boot loader sbboot from external flash memory to ram memory. The sbboot then loads the U-Boot along with the Secure loader. U-Boot then verifies the Kernel/system image integrity, then loads the Kernel/system image before passing control to it.

## Acronyms and Abbreviations

The following table lists the terms utilized within this part of the document.

Acronyms or Abbreviations	Description
<i>FUSE</i>	Filesystem in <b>U</b> ser <b>S</b> pace <b>E</b>
<i>OTP</i>	<b>O</b> ne- <b>T</b> ime- <b>P</b> rogrammable
<i>DOCSIS</i>	<b>D</b> ata <b>O</b> ver <b>C</b> able <b>S</b> ervice <b>I</b> nterface <b>S</b> pecification



# Image

## Image selection

The boot process shall be uninterruptible and shall irrevocably boot the image as specified in the boot environment.

In U-Boot set the "*bootdelay*" environment variable and/or define `CONFIG_BOOTDELAY` to -2.

Domain	Variable / Config name	Value
Boot-Image-Selection-1	<code>CONFIG_BOOTDELAY</code>	-2
Boot-Image-Selection-2	<i>bootdelay</i>	-2

## Image authenticity

It shall not be possible to boot from an unverified image. The secure boot feature in U-Boot shall be enabled. The secure boot feature is available from U-Boot 2013.07 version. To enable the secure boot feature, enable the following features:

```
CONFIG_FIT: Enables support for Flat Image Tree (FIT) uImage format.
CONFIG_FIT_SIGNATURE: Enables signature verification of FIT images.
CONFIG_RSA: Enables RSA algorithm used for FIT image verification.
CONFIG_OF_CONTROL: Enables Flattened Device Tree (FDT) configuration.
CONFIG_OF_SEPARATE: Enables separate build of u-Boot from the device tree.
CONFIG_DEFAULT_DEVICE_TREE: Specifies the default Device Tree used for the run-time configuration of U-Boot.
```

Generate the U-Boot image with public keys to validate and load the image. It shall use RSA2048 and SHA256 for authentication.

Domain	Config name	State
Boot-Image-Authenticity-1	<code>CONFIG_FIT</code>	<i>Enable</i>
Boot-Image-Authenticity-2	<code>CONFIG_FIT_SIGNATURE</code>	<i>Enable</i>
Boot-Image-Authenticity-3	<code>CONFIG_RSA</code>	<i>Enable</i>
Boot-Image-Authenticity-4	<code>CONFIG_OF_CONTROL</code>	<i>Enable</i>
Boot-Image-Authenticity-5	<code>CONFIG_OF_SEPARATE</code>	<i>Enable</i>
Boot-Image-Authenticity-6	<code>CONFIG_DEFAULT_DEVICE_TREE</code>	<i>Enable</i>

## Communication modes

### Disable USB, Serial and DOCSIS Support

To disable USB support in U-Boot, following config's shall not be defined:

```
CONFIG_CMD_USB: Enables basic USB support and the usb command.
CONFIG_USB_UHCI: Defines the lowlevel part.
CONFIG_USB_KEYBOARD: Enables the USB Keyboard.
CONFIG_USB_STORAGE: Enables the USB storage devices.
CONFIG_USB_HOST_ETHER: Enables USB Ethernet adapter support.
```

In addition, disable unnecessary communication modes like Ethernet, Serial ports, DOCSIS in U-Boot and sbboot that are not necessary.

Linux Kernel support for USB should be compiled-out if not required. If it is needed, the Linux Kernel should be configured to only enable the minimum required USB devices. User-initiated USB-fileystems should be treated with special care. Whether or not the filesystems are mounted in userspace (**FUSE**), restricted mount options should be observed.

Domain	Communication modes	State
Boot-Communication-1	USB	<i>Disabled and Compiled-out</i> if not required.
Boot-Communication-2	USB	Else, Kernel should be configured to only enable the minimum required USB devices and filesystems should be treated with special care.
Boot-Communication-3	Ethernet	<i>Disabled</i>
Boot-Communication-4	U-boot and sbboot DOCSIS	<i>Disabled</i>
Boot-Communication-5	Serial ports	<i>Disabled</i>

Domain	Config name	State
Boot-Communication-USB-1	CONFIG_CMD_USB	<i>Not defined</i>
Boot-Communication-USB-2	CONFIG_USB_UHCI	<i>Not defined</i>
Boot-Communication-USB-3	CONFIG_USB_KEYBOARD	<i>Not defined</i>
Boot-Communication-USB-4	CONFIG_USB_STORAGE	<i>Not defined</i>
Boot-Communication-USB-5	CONFIG_USB_HOST_ETHER	<i>Not defined</i>

## Disable all unused Network Interfaces

Only used network interfaces should be enabled. Where possible, services should also be limited to those necessary.

Domain	Communication modes	State
Boot-Communication-1	Network interfaces	Preferably <i>no network interface is allowed</i> , otherwise, restrict the services to those used.

## Remove or Disable Unnecessary Services, Ports, and Devices

Restrict the `services` , `ports` and `devices` to those used.

Domain	Object	Recommendations
Boot-Communication-1	<code>Services</code> , <code>ports</code> and <code>devices</code>	Restrict the <code>services</code> , <code>ports</code> and <code>devices</code> to those used.

## Disable flash access

### Recommendation:

In U-Boot following flash memory commands shall be disabled:

**NAND:** Support for nand flash access available through `do_nand` has to be disabled.

Domain	Command name	State
Boot-Communication-Flash-1	<code>do_nand</code>	<i>Disable</i>

Similarly sbboot should disable flash access support through command line if any.

## Consoles

### Disable serial console

Serial console output shall be disabled. To disable console output in U-Boot, set the following macros:

Domain	Config name	Value
Boot-Consoles-Serial-1	CONFIG_SILENT_CONSOLE	Disable
Boot-Consoles-Serial-2	CONFIG_SYS_DEVICE_NULLDEV	Disable
Boot-Consoles-Serial-3	CONFIG_SILENT_CONSOLE_UPDATE_ON_RELOC	Disable

Domain	Improvement
Boot-Consoles-1	Secure loader: No reference earlier?

And set "**silent**" environment variable. For the Secure loader, disable the traces by not defining the below macro:

Domain	Environment variable name	State
Boot-Consoles-Serial-1	INC_DEBUG_PRINT	<i>Not defined</i>

For sbboot proper configuration needs to be done to disable the serial console.

## Immutable environment variables

In U-Boot, ensure Kernel command line, boot commands, boot delay and other environment variables are immutable. This will prevent side-loading of alternate images, by restricting the boot selection to only the image in FLASH.

The environment variables shall be part of the text region in U-Boot as default environment variable and not in non-volatile memory.

Remove configuration options related to non-volatile memory, such as:

Domain	Config name	State
Boot-Consoles-Variables-1	CONFIG_ENV_IS_IN_MMC	#undef
Boot-Consoles-Variables-2	CONFIG_ENV_IS_IN_EEPROM	#undef
Boot-Consoles-Variables-3	CONFIG_ENV_IS_IN_FLASH	#undef
Boot-Consoles-Variables-4	CONFIG_ENV_IS_IN_DATAFLASH	#undef
Boot-Consoles-Variables-5	CONFIG_ENV_IS_IN_FAT	#undef
Boot-Consoles-Variables-6	CONFIG_ENV_IS_IN_NAND	#undef
Boot-Consoles-Variables-7	CONFIG_ENV_IS_IN_NVRAM	#undef
Boot-Consoles-Variables-8	CONFIG_ENV_IS_IN_ONENAND	#undef
Boot-Consoles-Variables-9	CONFIG_ENV_IS_IN_SPI_FLASH	#undef
Boot-Consoles-Variables-10	CONFIG_ENV_IS_IN_REMOTE	#undef
Boot-Consoles-Variables-11	CONFIG_ENV_IS_IN_UBI	#undef
Boot-Consoles-Variables-12	CONFIG_ENV_IS_NOWHERE	#define

## (Recommendation) Removal of memory dump commands

In U-Boot, following commands shall be disabled to avoid memory dumps:

```
md : Memory Display command.
mm : Memory modify command - auto incrementing address.
nm : Memory modify command - constant address.
mw : Memory write.
cp : Memory copy.
mwc : Memory write cyclic.
mdc : Memory display cyclic.
mtest : Simple ram read/write test.
loopw : Infinite write loop on address range.
```

Domain	Command name	State
Boot-Consoles-MemDump-1	md	Disabled
Boot-Consoles-MemDump-2	mm	Disabled
Boot-Consoles-MemDump-3	nm	Disabled
Boot-Consoles-MemDump-4	mw	Disabled
Boot-Consoles-MemDump-5	cp	Disabled
Boot-Consoles-MemDump-6	mwc	Disabled
Boot-Consoles-MemDump-7	mdc	Disabled
Boot-Consoles-MemDump-8	mtest	Disabled
Boot-Consoles-MemDump-9	loopw	Disabled

Similarly, memory dump support shall be disabled from sbboot.



## Part 3 - Hypervisor

---

Definition: "A hypervisor or virtual machine monitor (VMM) is computer software, firmware or hardware that creates and runs virtual machines".

It must include a signature verification (possibly delegated).

Domain	Improvement
Hypervisor-Abstract-1	Complete Hypervisor part ( <a href="#">jailhouse</a> / <a href="#">KVM</a> / <a href="#">Xen</a> ).

### Native or Bare-metal hypervisors

These hypervisors run directly on the host's hardware to control the hardware and to manage guest operating systems. Those are the ones we're interested in.

## Part 4 - Kernel

---

### Abstract

**System Hardening:** Best practices associated with the configuration of an embedded Linux based operating system. This section includes both hardening of the kernel itself, as well as specific configurations and patches used to protect against known vulnerabilities within the build and configuration of the root filesystem.

At the Kernel level, we must ensure that no console can be launched. It could be used to change the behavior of the system or to have more information about it. Another aspect is the protection of the memory used by the Kernel.

The next sub-sections contain information on various kernel configuration options to enhance the security in the kernel (3.10.17) and also for applications compiled to take advantage of these security features. Additionally, there are also configuration options that protect from known vulnerable configuration options. Here's a high level summary of various kernel configurations that shall be required for deployment.

## General configuration

### Mandatory Access Control

Kernel should controls access with labels and policy.

Domain	Object	Recommendations
Kernel-General-MAC-1	SMACK	Must implement a Mandatory Access Control.

Domain	Improvement
Kernel-MAC-1	Add MAC config note.

### Disable kexec

This prevents someone who gets root from supplanting the kernel. This can be used as a way to bypass signed kernels.

Domain	Config name	Value
Kernel-General-kexec-1	CONFIG_KEXEC	n

### Disable kernel IP auto-configuration

It is preferable to have an IP configuration performed using a user-space tool as these tend to have more validation. We do not want the network interface coming up until the system has come up properly.

Domain	Config name	Value
Kernel-General-IPAutoConf-1	CONFIG_IP_PNP	n

### Disable Sysctl syscall support

Enabling this will result in code being included that is hard to maintain and not well tested.

Domain	Config name	Value
Kernel-General-SysCtl_SysCall-1	CONFIG_SYSCTL_SYSCALL	n

## Disable Legacy Linux Support

There are some Kernel Configs which are present only to support legacy binaries. See also "Consoles" part in order to disabling support for legacy binary formats. The `uselib` system call, in particular, has no valid use in any `libc6` or `uclibc` system in recent times. This configuration is supported in **Linux 3.15 and greater** and thus should only be disabled for such versions.

Domain	Config name	Value
Kernel-General-LegacyLinux-1	CONFIG_USELIB	n

## Disable firmware auto-loading user mode helper

The firmware auto loading helper, which is a utility executed by the kernel on `hotplug` events requiring firmware, needs to be set `setuid`. As a result of this, the helper utility is an attractive target for attackers with control of physical ports on the device. Disabling this configuration that is supported in **Linux 3.9 and greater**.

Domain	Config name	Value
Kernel-General-FirmHelper-1	CONFIG_FW_LOADER_USER_HELPER	n

## Enable Kernel Panic on OOPS

When fuzzing the kernel or attempting kernel exploits attackers are likely to trigger kernel OOPSes. Setting the behavior on OOPS to PANIC can impede their progress.

This configuration is supported in **Linux 3.5 and greater** and thus should only be enabled for such versions.

Domain	Config name	Value
Kernel-General-PanicOnOOPS-1	CONFIG_PANIC_ON_OOPS	y



## Disable socket monitoring interface

These monitors can be used to inspect shared file descriptors on Unix Domain sockets or traffic on 'localhost' which is otherwise assumed to be confidential.

The `CONFIG_PACKET_DIAG` configuration is supported in **Linux 3.7 and greater** and thus should only be disabled for such versions.

The `CONFIG_UNIX_DIAG` configuration is supported in **Linux 3.3 and greater** and thus should only be disabled for such versions.

Domain	Config name	Value
Kernel-General-SocketMon-1	<code>CONFIG_PACKET_DIAG</code>	n
Kernel-General-SocketMon-2	<code>CONFIG_UNIX_DIAG</code>	n

## Disable BPF JIT

The BPF JIT can be used to create kernel-payloads from firewall table rules.

This configuration for is supported in **Linux 3.16 and greater** and thus should only be disabled for such versions.

Domain	Config name	Value
Kernel-General-BPF_JIT-1	<code>CONFIG_BPF_JIT</code>	n

## Enable Enforced Module Signing

The kernel should never allow an unprivileged user the ability to load specific kernel modules, since that would provide a facility to unexpectedly extend the available attack surface.

To protect against even privileged users, systems may need to either disable module loading entirely, or provide signed modules (e.g. `CONFIG_MODULE_SIG_FORCE`, or dm-crypt with LoadPin), to keep from having root load arbitrary kernel code via the module loader interface.

This configuration is supported in **Linux 3.7 and greater** and thus should only be enabled for such versions.

Domain	Config name	Value
Kernel-General-ModuleSigning-1	<code>CONFIG_MODULE_SIG_FORCE</code>	y



## Disable all USB, PCMCIA (and other `hotplug` bus) drivers that aren't needed

To reduce the attack surface, the driver enumeration, probe, and operation happen in the kernel. The driver data is parsed by the kernel, so any logic bugs in these drivers can become kernel exploits.

Domain	Object	State
Kernel-General-Drivers-1	<code>USB</code>	<i>Disabled</i>
Kernel-General-Drivers-2	<code>PCMCIA</code>	<i>Disabled</i>
Kernel-General-Drivers-3	Other <code>hotplug</code> bus	<i>Disabled</i>

## Position Independent Executables

Domain	Improvement
Kernel-General-IndependentExec-1	Kernel or/and platform part ?

Domain	compiler and linker options	State
Kernel-General-IndependentExec-1	<code>-pie -fpic</code>	<i>Enable</i>

Produce a position independent executable on targets which supports it.

## Prevent Overwrite Attacks

`-z,relro` linking option helps during program load, several ELF memory sections need to be written by the linker, but can be turned read-only before turning over control to the program. This prevents some Global Offset Table GOT overwrite attacks, or in the dtors section of the ELF binary.

Domain	compiler and linker options	State
Kernel-General-OverwriteAttacks-1	<code>-z,relro</code>	<i>Enable</i>
Kernel-General-OverwriteAttacks-2	<code>-z,now</code>	<i>Enable</i>

During program load, all dynamic symbols are resolved, allowing for the complete GOT to be marked read-only (due to `-z relro` above). This prevents GOT overwrite attacks. For very large application, this can incur some performance loss during initial load while symbols are resolved, but this shouldn't be an issue for daemons.





## Library linking

Domain	Improvement
Kernel-General-LibraryLinking-1	Keep this part?

It is recommended that dynamic linking should generally not be allowed. This will avoid the user from replacing a library with malicious library. All libraries should be linked statically, but this is difficult to implement.

Domain	<code>compiler</code> and <code>linker</code> options	<i>State</i>
Kernel-General-LibraryLinking-1	<code>-static</code>	<i>Enable</i>

# Memory

## Restrict access to kernel memory

The `/dev/kmem` file in Linux systems is directly mapped to kernel virtual memory. This can be disastrous if an attacker gains root access, as the attacker would have direct access to kernel virtual memory.

To disable the `/dev/kmem` file, which is very infrequently used by applications, the following kernel option should be set in the compile-time kernel configuration:

Domain	Config name	Value
Kernel-Memory-RestrictAccess-1	CONFIG_DEVMEM	n

In case applications in userspace need `/dev/kmem` support, it should be available only for authenticated applications.

## Disable access to a kernel core dump

This kernel configuration disables access to a kernel core dump from user space. If enabled, it gives attackers a useful view into kernel memory.

Domain	Config name	Value
Kernel-Memory-CoreDump-1	CONFIG_PROC_KCORE	n

## Disable swap

If not disabled, attackers can enable swap at runtime, add pressure to the memory subsystem and then scour the pages written to swap for useful information.

Domain	Config name	Value
Kernel-Memory-Swap-1	CONFIG_SWAP	n

## Disable "Load All Symbols"

There is a `/proc/kallsyms` file which exposes the kernel memory space address of many kernel symbols (functions, variables, etc...). This information is useful to attackers in identifying kernel versions/configurations and in preparing payloads for the exploits of kernel space.

Both `KALLSYMS_ALL` and `KALLSYMS` shall be disabled;

Domain	Config name	Value
Kernel-Memory-LoadAllSymbols-1	<code>CONFIG_KALLSYMS</code>	n
Kernel-Memory-LoadAllSymbols-2	<code>CONFIG_KALLSYMS_ALL</code>	n

## Stack protection

To prevent stack-smashing, similar to the stack protector used for ELF programs in user-space, the kernel can protect its internal stacks as well.

This configuration is supported in **Linux 3.11 and greater** and thus should only be enabled for such versions.

This configuration also requires building the kernel with the **gcc compiler 4.2 or greater**.

Domain	Config name	Value
Kernel-Memory-Stack-1	<code>CONFIG_CC_STACKPROTECTOR</code>	y

Other defenses include things like shadow stacks.

## Disable access to `/dev/mem`

The `/dev/mem` file in Linux systems is directly mapped to physical memory. This can be disastrous if an attacker gains root access, as the attacker would have direct access to physical memory through this convenient device file. It may not always be possible to disable such file, as some applications might need such support. In that case, then this device file should be available only for authenticated applications.

This configuration is supported in **Linux 4.0 and greater** and thus should only be disabled for such versions.

Domain	Config name	Value
Kernel-Memory-Access-1	<code>CONFIG_DEVMEM</code>	n



## Disable cross-memory attach

Disable the `processvm*v` syscalls which allow one process to peek/poke the virtual memory of another.

This configuration is supported in **Linux 3.5 and greater** and thus should only be disabled for such versions.

Domain	Config name	Value
Kernel-Memory-CrossMemAttach-1	CROSS_MEMORY_ATTACH	n

## Stack Smashing Attacks

Domain	compiler and linker options	State
Kernel-Memory-StackSmashing-1	-fstack-protector-all	Enable

Emit extra code to check for buffer overflows, such as stack smashing attacks.

## Detect Buffer Overflows

Domain	compiler and linker options	Value
Kernel-Memory-BufferOverflows-1	-D_FORTIFY_SOURCE	2

Helps detect some buffer overflow errors.

## Serial

### Disable serial console

The serial console should be disabled to prevent an attacker from accessing this powerful interface.

Domain	Config name	Value
Kernel-Consoles-Serial-1	CONFIG_SERIAL_8250	n
Kernel-Consoles-Serial-2	CONFIG_SERIAL_8250_CONSOLE	n
Kernel-Consoles-Serial-3	CONFIG_SERIAL_CORE	n
Kernel-Consoles-Serial-4	CONFIG_SERIAL_CORE_CONSOLE	n

### Bake-in the kernel command-line

The kernel command-line is used to control many aspects of the booting kernel, and is prone to tampering as they are passed in RAM with little to no reverse validation on these parameters. To prevent this type of attack, the kernel shall be configured to ignore commands line arguments, and use pre-configured (compile time) options instead.

Set the kernel command line in the `CONFIG_CMDLINE KConfig` item and then pass no arguments from the bootloader.

Domain	Config name	Value
Kernel-Consoles-CommandLine-1	CONFIG_CMDLINE_BOOL	y
Kernel-Consoles-CommandLine-2	CONFIG_CMDLINE	"insert kernel command line here"
Kernel-Consoles-CommandLine-3	CONFIG_CMDLINE_OVERRIDE	y

It is recommended that any per-device settings (e.g: MAC addresses, serial numbers, etc.) be stored and accessed from read-only memory (or files), and that any such parameters be verified (signature checking) prior to their use.

### Disable KGDB

The Linux kernel supports KGDB over USB and console ports. These mechanisms are controlled by the `kgdbdbgp` and `kgdboc` kernel command-line parameters. It is important to ensure that no shipping product contains a kernel with KGDB compiled-in.

Domain	Config name	Value

Kernel-Consoles-KDBG-1

CONFIG\_KGDB

n

## Disable magic sysrq support

On a few architectures, you can access a powerful debugger interface from the keyboard. The same powerful interface can be present on the serial console (responding to serial break) of Linux on other architectures. Disable to avoid potentially exposing this powerful backdoor.

Domain	Config name	Value
Kernel-Consoles-SysRQ-1	CONFIG_MAGIC_SYSRQ	n

## Disable support for binary formats other than ELF

This will make possible to plug wrapper-driven binary formats into the kernel. It enables support for binary formats other than ELF. Providing the ability to use alternate interpreters would assist an attacker in discovering attack vectors.

Domain	Config name	Value
Kernel-Consoles-BinaryFormat-1	CONFIG_BINFMT_MISC	n



## Debug

No debuggers shall be present on the file system. This includes, but is not limited to, the GNU Debugger client/server (commonly known in their short form names such as the `gdb` and `gdbserver` executable binaries respectively), the `LLDB` next generation debugger or the `TCF` (Target Communications Framework) agnostic framework. Including these binaries as part of the file system will facilitate an attacker's ability to reverse engineer and debug (either locally or remotely) any process that is currently executing on the device.

## Kernel debug symbols

Debug symbols should always be removed from production kernels as they provide a lot of information to attackers.

Domain	Config name	Value
Kernel-Debug-Symbols-1	CONFIG_DEBUG_INFO	n

These kernel debug symbols are enabled by other config items in the kernel. Care should be taken to disable those also. If `CONFIG_DEBUG_INFO` cannot be disabled, then enabling `CONFIG_DEBUG_INFO_REDUCED` is second best.

## Disable Kprobes

Kprobes enables you to dynamically break into any kernel routine and collect debugging and performance information non-disruptively. You can trap at almost any kernel code address, specifying a handler routine to be invoked when the breakpoint is hit.

Domain	Config name	Value
Kernel-Debug-Kprobes-1	CONFIG_KPROBES	n

## Disable Tracing

FTrace enables the kernel to trace every kernel function. Providing kernel trace functionality would assist an attacker in discovering attack vectors.

Domain	Config name	Value
Kernel-Debug-Tracing-1	CONFIG_FTRACE	n

## Disable Profiling

Profiling and OProfile enables profiling the whole system, include the kernel, kernel modules, libraries, and applications. Providing profiling functionality would assist an attacker in discovering attack vectors.

Domain	Config name	Value
Kernel-Debug-Profiling-1	CONFIG_OPROFILE	n
Kernel-Debug-Profiling-2	CONFIG_PROFILING	n

## Disable OOPS print on BUG()

The output from OOPS print can be helpful in Return Oriented Programming (ROP) when trying to determine the effectiveness of an exploit.

Domain	Config name	Value
Kernel-Debug-OOPSONBUG-1	CONFIG_DEBUG_BUGVERBOSE	n

## Disable Kernel Debugging

There are development-only branches of code in the kernel enabled by the `DEBUG_KERNEL` conf. This should be disabled to compile-out these branches.

Domain	Config name	Value
Kernel-Debug-Dev-1	CONFIG_DEBUG_KERNEL	n
Kernel-Debug-Dev-2	CONFIG_EMBEDDED	n

In some kernel versions, disabling this requires also disabling `CONFIG_EMBEDDED`, and `CONFIG_EXPERT`. Disabling `CONFIG_EXPERT` makes it impossible to disable `COREDUMP`, `DEBUG_BUGVERBOSE`, `NAMESPACES`, `KALLSYMS` and `BUG`. In which case it is better to leave this enabled than enable the others.

## Disable the kernel debug filesystem

The kernel debug filesystem presents a lot of useful information and means of manipulation of the kernel to an attacker.

Domain	Config name	Value
Kernel-Debug-FileSystem-1	CONFIG_DEBUG_FS	n

## Disable BUG() support

The kernel will display backtrace and register information for BUGs and WARNs in kernel space, making it easier for attackers to develop exploits.

Domain	Config name	Value
Kernel-Debug-BUG-1	CONFIG_BUG	n

## Disable core dumps

Core dumps provide a lot of debug information for hackers. So disabling core dumps are recommended in production builds.

This configuration is supported in **Linux 3.7 and greater** and thus should only be disabled for such versions.

Domain	Config name	Value
Kernel-Debug-CoreDumps-1	CONFIG_COREDUMP	n

## Kernel Address Display Restriction

When attackers try to develop "run anywhere" exploits for kernel vulnerabilities, they frequently need to know the location of internal kernel structures. By treating kernel addresses as sensitive information, those locations are not visible to regular local users.

**/proc/sys/kernel/kptr\_restrict is set to "1"** to block the reporting of known kernel address leaks.

Domain	File name	Value
Kernel-Debug-AdressDisplay-1	/proc/sys/kernel/kptr_restrict	1

Additionally, various files and directories should be readable only by the root user: `/boot/vmlinuz*` , `/boot/System.map*` , `/sys/kernel/debug/` , `/proc/slabinfo`

Domain	File or Directorie name	State
Kernel-Debug-AdressDisplay-1	/boot/vmlinuz*	Readable Only for root user
Kernel-Debug-AdressDisplay-2	/boot/System.map*	Readable Only for root user
Kernel-Debug-AdressDisplay-3	/sys/kernel/debug/	Readable Only for root user
Kernel-Debug-AdressDisplay-4	/proc/slabinfo	Readable Only for root user

## DMESG Restrictions

When attackers try to develop "run anywhere" exploits for vulnerabilities, they frequently will use `dmesg` output. By treating `dmesg` output as sensitive information, this output is not available to the attacker.

**/proc/sys/kernel/dmesg\_restrict can be set to "1"** to treat dmesg output as sensitive.

Domain	File name	Value
Kernel-Debug-DMESG-1	/proc/sys/kernel/dmesg_restrict	1

Enable the below compiler and linker options when building user-space applications to avoid stack smashing, buffer overflow attacks.

## Disable /proc/config.gz

It is extremely important to not expose the kernel configuration used on a production device to a potential attacker. With access to the kernel config, it could be possible for an attacker to build a custom kernel for the device that may disable critical security features.

Domain	config name	Value
Kernel-Debug-Config-1	CONFIG_IKCONFIG	n

# File System

---

## Disable all file systems not needed

To reduce the attack surface, file system data is parsed by the kernel, so any logic bugs in file system drivers can become kernel exploits.

### Disable NFS file system

NFS FileSystems are useful during development phases, but this can be a very helpful way for an attacker to get files when you are in production mode, so we must disable them.

Domain	Config name	Value
Kernel-FileSystems-NFS-1	CONFIG_NFSD	n
Kernel-FileSystems-NFS-2	CONFIG_NFS_FS	n

## Partition Mount Options

There are several security restrictions that can be set on a filesystem when it is mounted. Some common security options include, but are not limited to:

`nosuid` - Do not allow set-user-identifier or set-group-identifier bits to take effect.

`nodev` - Do not interpret character or block special devices on the filesystem.

`noexec` - Do not allow execution of any binaries on the mounted filesystem.

`ro` - Mount filesystem as read-only.

The following flags shall be used for mounting common filesystems:

Domain	Partition	Value
Kernel-FileSystems-Mount-1	<code>/boot</code>	<code>nosuid</code> , <code>nodev</code> and <code>noexec</code> .
Kernel-FileSystems-Mount-2	<code>/var</code> & <code>/tmp</code>	In <code>/etc/fstab</code> or <code>vfstab</code> , add <code>nosuid</code> , <code>nodev</code> and <code>noexec</code> .
Kernel-FileSystems-Mount-3	<i>Non-root local</i>	If type is <code>ext2</code> or <code>ext3</code> and mount point not '/', add <code>nodev</code> .
Kernel-FileSystems-Mount-4	<i>Removable storage</i>	Add <code>nosuid</code> , <code>nodev</code> and <code>noexec</code> .
Kernel-FileSystems-Mount-5	<i>Temporary storage</i>	Add <code>nosuid</code> , <code>nodev</code> and <code>noexec</code> .
Kernel-FileSystems-Mount-6	<code>/dev/shm</code>	Add <code>nosuid</code> , <code>nodev</code> and <code>noexec</code> .
Kernel-FileSystems-Mount-7	<code>/dev</code>	Add <code>nosuid</code> and <code>noexec</code> .

If `CONFIG_DEVTMPFS_MOUNT` is set, then the kernel will mount `/dev` and will not apply the `nosuid` , `noexec` options. Either disable `CONFIG_DEVTMPFS_MOUNT` or add a remount with `noexec` and `nosuid` options to system startup.

Domain	Config name	State or Value
Kernel-FileSystems-Mount-1	<code>CONFIG_DEVTMPFS_MOUNT</code>	<i>Disabled</i> or add remount with <code>noexec</code> and <code>nosuid</code> to system startup.

## Part 5 - Platform

---

### Abstract

This part focuses on the AGL platform including all tools and techniques used to upgrade the security and downgrade the danger. It must be possible to apply the two fundamental principles written at the very beginning of the document. First of all, security management must remain simple. You must also prohibit everything by default, and then define a set of authorization rules. As cases to deal with, we must:

- Implement a **MAC** for processes and files.
- Limit communication between applications (*SystemBus* and *SystemD* part).
- Prohibit all tools used during development mode (*Utilities* and *Services* part).
- Manage user capabilities (*Users* part).
- Manage application permissions and policies (*AGLFW* part).

The tools and concepts used to meet these needs are only examples. Any other tool that meets the need can be used.

In AGL, as in many other embedded systems, different security mechanisms settle in the core layers to ensure isolation and data privacy. While the Mandatory Access Control layer (**SMACK**) provides global security and isolation, other mechanisms like **Cynara** are required to check application's permissions at runtime. Applicative permissions (also called "*privileges*") may vary depending on the user and the application being run: an application should have access to a given service only if it is run by the proper user and if the appropriate permissions are granted.

---



## Acronyms and Abbreviations

The following table lists the terms utilized within this part of the document.

Acronyms or Abbreviations	Description
<i>ACL</i>	<b>A</b> ccess <b>C</b> ontrol <b>L</b> ists
<i>alsa</i>	<b>A</b> dvanced <b>L</b> inux <b>S</b> ound <b>A</b> rchitecture
<i>API</i>	<b>A</b> pplication <b>P</b> rogramming <b>I</b> nterface
<i>AppFw</i>	<b>A</b> pplication <b>F</b> ramework
<i>Cap</i>	<b>C</b> apabilities
<i>DAC</i>	<b>D</b> iscretionary <b>A</b> ccess <b>C</b> ontrol
<i>DDOS</i>	<b>D</b> istributed <b>D</b> enial <b>O</b> f <b>S</b> ervice
<i>DOS</i>	<b>D</b> enial <b>O</b> f <b>S</b> ervice
<i>IPC</i>	<b>I</b> nter- <b>P</b> rocess <b>C</b> ommunication
<i>MAC</i>	<b>M</b> andatory <b>A</b> ccess <b>C</b> ontrol
<i>PAM</i>	<b>P</b> luggable <b>A</b> uthentication <b>M</b> odules
<i>SMACK</i>	<b>S</b> implified <b>M</b> andatory <b>A</b> ccess <b>C</b> ontrol <b>K</b> ernel

# Mandatory Access Control

---

We decided to put the **MAC** protection on the platform part despite the fact that it applies to the kernel too, since its use will be mainly at the platform level (except floor part).

**Mandatory Access Control (MAC)** is a protection provided by the Linux kernel that requires a **Linux Security Module (LSM)**. AGL uses an **LSM** called **Simplified Mandatory Access Control Kernel (SMACK)**. This protection involves the creation of **SMACK** labels as part of the extended attributes **SMACK** labels to the file extended attributes. And a policy is also created to define the behaviour of each label.

The kernel access controls is based on these labels and this policy. If there is no rule, no access will be granted and as a consequence, what is not explicitly authorized is forbidden.

There are two types of **SMACK** labels:

- **Execution SMACK** (Attached to the process): Defines how files are *accessed* and *created* by that process.
- **File Access SMACK** (Written to the extended attribute of the file): Defines *which* process can access the file.

By default a process executes with its File Access **SMACK** label unless an Execution **SMACK** label is defined.

AGL's **SMACK** scheme is based on the *Tizen 3 Q2/2015*. It divides the System into the following domains:

- Floor.
- System.
- Applications, Services and User.

See [AGL security framework review](#) and [Smack White Paper](#) for more information.

---

## Floor

The *floor* domain includes the base system services and any associated data and libraries. This data remains unchanged at runtime. Writing to floor files or directories is allowed only in development mode or during software installation or upgrade.

The following table details the *floor* domain:

Label	Name	Execution SMACK	File Access SMACK
-	Floor	r-x for all	Only kernel and internal kernel thread.
^	Hat	--- for all	rx on all domains.
*	Star	rwX for all	None

- The Hat label is Only for privileged system services (currently only systemd-journal). Useful for backup or virus scans. No file with this label should exist except in the debug log.
- The Star label is used for device files or /tmp Access restriction managed via **DAC**. Individual files remain protected by their **SMACK** label.

Domain	Label name	Recommendations
Kernel-MAC-Floor-1	^	Only for privileged system services.
Kernel-MAC-Floor-2	*	Used for device files or /tmp Access restriction via DAC.

## System

The *system* domain includes a reduced set of core system services of the OS and any associated data. This data may change at runtime.

The following table details the *system* domain:

Label	Name	Execution SMACK	File Access SMACK
System	System	None	Privileged processes
System::Run	Run	rwxtat1 for User and System label	None
System::Shared	Shared	rwxtat1 for system domain r-x for User label	None
System::Log	Log	rwa for System label xa for user label	None
System::Sub	SubSystem	Subsystem Config files	SubSystem only

Domain	Label name	Recommendations
Kernel-MAC-System-1	System	Process should write only to file with transmute attribute.
Kernel-MAC-System-2	System::run	Files are created with the directory label from user and system domain (transmute) Lock is implicit with w .
Kernel-MAC-System-3	System::Shared	Files are created with the directory label from system domain (transmute) User domain has locked privilege.
Kernel-MAC-System-4	System::Log	Some limitation may impose to add w to enable append.
Kernel-MAC-System-5	System::Sub	Isolation of risky Subsystem.

## Applications, Services and User

The *application*, *services* and *user* domain includes code that provides services to the system and user, as well as any associated data. All code running on this domain is under *Cynara* control.

The following table details the *application*, *services* and *user* domain:

Label	Name	Execution SMACK	File Access SMACK
User::Pkg::\$AppID	AppID	<code>rwx</code> (for files created by the App). <code>rx</code> for files installed by <b>AppFw</b>	\$App runtime executing \$App
User::Home	Home	<code>rwx-t</code> from System label <code>r-x-l</code> from App	None
User::App-Shared	Shared	<code>rwxt</code> from System and User domains label of \$User	None

Domain	Label name	Recommendations
Kernel-MAC-System-1	User::Pkg::\$AppID	Only one Label is allowed per App. A data directory is created by the AppFw in <code>rwx</code> mode.
Kernel-MAC-System-2	User::Home	AppFw needs to create a directory in <code>/home/\$USER/App-Shared</code> at first launch if not present with label <code>app-data</code> access is <code>User::App-Shared</code> without transmute.
Kernel-MAC-System-3	User::App-Shared	Shared space between all App running for a given user.

# SystemD

`afm-system-daemon` is used to:

- Manage users and user sessions.
- Setup applications and services (*CGroups*, *namespaces*, autostart, permissions).
- Use of `libsystemd` for its programs (event management, **D-Bus** interface).

Domain	Object	Recommendations
Platform-SystemD-1	Security model	Use Namespaces for containerization.
Platform-SystemD-2	Security model	Use CGroups to organise processes.

See [systemd integration and user management](#) for more information.

## Benefits

- Removal of one privileged process: **afm-user-daemon**
- Access and use of high level features:
  - Socket activation.
  - Management of users and integration of **PAM**.
  - Dependency resolution to services.
  - `cgroups` and resource control.
  - `Namespaces` containerization.
  - Autostart of required API.
  - Permissions and security settings.
  - Network management.

## CGroups

Control Groups offer a lot of features, with the most useful ones you can control: Memory usage, how much CPU time is allocated, how much device I/O is allowed or which devices can be accessed. **SystemD** uses *CGroups* to organise processes (each service is a *CGroups*, and all processes started by that service use that *CGroups*). By default, **SystemD** automatically creates a hierarchy of slice, scope and service units to provide a unified structure for the *CGroups* tree. With the `systemctl` command, you can further modify this structure by creating custom slices. Currently, in AGL, there are 2 slices (**user.slice** and **system.slice**).

## Namespaces

### User side

There are several ways of authenticating users (Key Radio Frequency, Phone, Gesture, ...). Each authentication provides dynamic allocation of **uids** to authenticated users. **Uids** is used to ensure privacy of users and **SMACK** for applications privacy.

First, the user initiates authentication with **PAM** activation. **PAM** Standard offers highly configurable authentication with modular design like face recognition, Voice identification or with a password. Then users should access identity services with services and applications.

## D-Bus

D-Bus is a well-known **IPC** (Inter-Process Communication) protocol (and daemon) that helps applications to talk to each other. The use of D-Bus is great because it allows to implement discovery and signaling.

The D-Bus session is by default addressed by environment variable `DBUS_SESSION_BUS_ADDRESS`. Using **systemd** variable `DBUS_SESSION_BUS_ADDRESS` is automatically set for user sessions. D-Bus usage is linked to permissions.

D-Bus has already had several [security issues](#) (mostly **DoS** issues), to allow applications to keep talking to each other. It is important to protect against this type of attack to keep the system more stable.

Domain	Object	Recommendations
Platform-DBus-1	Security model	Use D-Bus as IPC.
Platform-DBus-2	Security model	Apply D-BUS security patches: <a href="#">D-Bus CVE</a>



## System services and daemons

Domain	Improvement
Platform-Services-1	SystemD ?
Platform-Services-2	Secure daemon ?

## Tools

- **connman**: An internet connection manager designed to be slim and to use as few resources as possible. It is a fully modular system that can be extended, through plug-ins, to support all kinds of wired or wireless technologies.
- **bluez** is a Bluetooth stack. Its goal is to program an implementation of the Bluetooth wireless standards specifications. In addition to the basic stack, the `bluez-utils` and `bluez-firmware` packages contain low level utilities such as `dfutool` which can interrogate the Bluetooth adapter chipset in order to determine whether its firmware can be upgraded.
- **gstreamer** is a pipeline-based multimedia framework. It can be used to build a system that reads files in one format, processes them, and exports them in another format.
- **alsa** is a software framework and part of the Linux kernel that provides an **API** for sound card device drivers.

Domain	Tool name	State
Platform-Utilities-1	<code>connman</code>	<i>Used</i> as a connection manager.
Platform-Utilities-2	<code>bluez</code>	<i>Used</i> as a Bluetooth manager.
Platform-Utilities-3	<code>gstreamer</code>	<i>Used</i> to manage multimedia file format.
Platform-Utilities-4	<code>alsa</code>	<i>Used</i> to provides an API for sound card device drivers.

## Application framework/model (AppFw)

The application framework manages:

- The applications and services management: Installing, Uninstalling, Listing, ...
- The life cycle of applications: Start -> (Pause, Resume) -> Stop.
- Events and signals propagation.
- Privileges granting and checking.
- API for interaction with applications.

- The **security model** refers to the security model used to ensure security and to the tools that are provided for implementing that model. It's an implementation detail that should not impact the layers above the application framework.
- The **security model** refers to how **DAC** (Discretionary Access Control), **MAC** (Mandatory Access Control) and Capabilities are used by the system to ensure security and privacy. It also includes features of reporting using audit features and by managing logs and alerts.

The **AppFw** uses the security model to ensure the security and the privacy of the applications that it manages. It must be compliant with the underlying security model. But it should hide it to the applications.

Domain	Object	Recommendations
Platform-AGLFW-AppFw-1	Security model	Use the AppFw as Security model.

See [AGL AppFw Privileges Management](#) and [AGL - Application Framework Documentation](#) for more information.

## Cynara

There's a need for another mechanism responsible for checking applicative permissions: Currently in AGL, this task depends on a policy-checker service (**Cynara**).

- Stores complex policies in databases.
- "Soft" security (access is checked by the framework).

Cynara interact with **D-Bus** in order to deliver this information.

Domain	Object	Recommendations
Platform-AGLFW-Cynara-1	Permissions	Use Cynara as policy-checker service.

## Policies

- Policy rules:
  - Are simple - for pair [application context, privilege] there is straight answer (single Policy Type): [ALLOW / DENY / ...].
  - No code is executed (no script).
  - Can be easily cached and managed.
- Application context (describes id of the user and the application credentials) It is build of:
  - UID of the user that runs the application.
  - **SMACK** label of application.

## Holding policies

Policies are kept in buckets. Buckets are set of policies which have additional a property of default answer, the default answer is yielded if no policy matches searched key. Buckets have names which might be used in policies (for directions).

## Utilities

- **busybox**: Software that provides several stripped-down Unix tools in a single executable file. Of course, it will be necessary to use a "production" version of **busybox** in order to avoid all the tools useful only in development mode.

Domain	Tool name	State
Platform-Utilities-1	busybox	Used to provide a number of tools. Do not compile development tools.

## Functionalities to exclude in production mode

In production mode, a number of tools must be disabled to prevent an attacker from finding logs for example. This is useful to limit the visible surface and thus complicate the fault finding process. The tools used only in development mode are marked by an '**agl-devel**' feature. When building in production mode, these tools will not be compiled.

Domain	Utility name and normal path	State
Platform-Utilities-1	chgrp in /bin/chgrp	Disabled
Platform-Utilities-2	chmod in /bin/chmod	Disabled
Platform-Utilities-3	chown in /bin/chown	Disabled
Platform-Utilities-4	dmesg in /bin/dmesg	Disabled
Platform-Utilities-5	Dnsdomainname in /bin/dnsdomainname	Disabled
Platform-Utilities-6	dropbear , Remove "dropbear" from /etc/init.d/rcs	Disabled
Platform-Utilities-7	Editors in (vi) /bin/vi	Disabled
Platform-Utilities-8	find in /bin/find	Disabled
Platform-Utilities-9	gdbserver in /bin/gdbserver	Disabled
Platform-Utilities-10	hexdump in /bin/hexdump	Disabled
Platform-Utilities-11	hostname in /bin/hostname	Disabled
Platform-Utilities-12	install in /bin/install	Disabled
Platform-Utilities-13	iostat in /bin/iostat	Disabled
Platform-Utilities-14	killall in /bin/killall	Disabled
Platform-Utilities-15	klogd in /sbin/klogd	Disabled
Platform-Utilities-16	logger in /bin/logger	Disabled
Platform-Utilities-17	lsmod in /sbin/lsmod	Disabled
Platform-Utilities-18	pmap in /bin/pmap	Disabled
Platform-Utilities-19	ps in /bin/ps	Disabled

Platform-Utilities-20	ps in /bin/ps	Disabled
Platform-Utilities-21	rpm in /bin/rpm	Disabled
Platform-Utilities-22	SSH	Disabled
Platform-Utilities-23	stbhotplug in /sbin/stbhotplug	Disabled
Platform-Utilities-24	strace in /bin/trace	Disabled
Platform-Utilities-25	su in /bin/su	Disabled
Platform-Utilities-26	syslogd in (logger) /bin/logger	Disabled
Platform-Utilities-27	top in /bin/top	Disabled
Platform-Utilities-28	UART in /proc/tty/driver/	Disabled
Platform-Utilities-29	which in /bin/which	Disabled
Platform-Utilities-30	who and whoami in /bin/whoami	Disabled
Platform-Utilities-31	awk (busybox)	Enabled
Platform-Utilities-32	cut (busybox)	Enabled
Platform-Utilities-33	df (busybox)	Enabled
Platform-Utilities-34	echo (busybox)	Enabled
Platform-Utilities-35	fdisk (busybox)	Enabled
Platform-Utilities-36	grep (busybox)	Enabled
Platform-Utilities-37	mkdir (busybox)	Enabled
Platform-Utilities-38	mount (vfat) (busybox)	Enabled
Platform-Utilities-39	printf (busybox)	Enabled
Platform-Utilities-40	sed in /bin/sed (busybox)	Enabled
Platform-Utilities-41	tail (busybox)	Enabled
Platform-Utilities-42	tee (busybox)	Enabled
Platform-Utilities-43	test (busybox)	Enabled

The *Enabled* Unix/Linux utilities above shall be permitted as they are often used in the start-up scripts and for USB logging. If any of these utilities are not required by the device then those should be removed.

## Users

The user policy can group users by function within the car. For example, we can consider a driver and his passengers. Each user is assigned to a single group to simplify the management of space security.

## Root Access

The main applications, those that provide the principal functionality of the embedded device, should not execute with root identity or any capability.

If the main application is allowed to execute at any capability, then the entire system is at the mercy of the said application's good behaviour. Problems arise when an application is compromised and able to execute commands which could consistently and persistently compromise the system by implanting rogue applications.

It is suggested that the middleware and the UI should run in a context on a user with no capability and all persistent resources should be maintained without any capability.

One way to ensure this is by implementing a server-client paradigm. Services provided by the system's drivers can be shared this way. The other advantage of this approach is that multiple applications can share the same resources at the same time.

Domain	Object	Recommendations
Platform-Users-root-1	Main application	Should not execute as root.
Platform-Users-root-2	UI	Should run in a context on a user with no capability.

Root access should not be allowed for the following utilities:

Domain	Utility name	State
Platform-Users-root-3	login	<i>Not allowed</i>
Platform-Users-root-4	su	<i>Not allowed</i>
Platform-Users-root-5	ssh	<i>Not allowed</i>
Platform-Users-root-6	scp	<i>Not allowed</i>
Platform-Users-root-7	sftp	<i>Not allowed</i>

Root access should not be allowed for the console device. The development environment should allow users to login with pre-created user accounts.

Switching to elevated privileges shall be allowed in the development environment via `sudo`.



## Capabilities

Domain	Improvement
Platform-Users-Capabilities-1	Kernel or Platform-user?
Platform-Users-Capabilities-2	Add config note.

The goal is to restrict functionality that will not be useful in **AGL**. They are integrated into the **LSM**. Each privileged transaction is associated with a capability. These capabilities are divided into three groups:

- e: Effective: This means the capability is “activated”.
- p: Permitted: This means the capability can be used/is allowed.
- i: Inherited: The capability is kept by child/subprocesses upon `execve()` for example.



## Part 6 - Application

---

### Abstract

**Application Hardening:** Best practices to apply to the build and release of user space applications, in order to reduce the number of attack surfaces used by potential attackers.

The term of Application (App) has a very wide definition in **AGL**. Almost anything which is not in the core Operating System (OS) is an Application. Applications can be included in the base software package (image) or can be added at run-time.

---

### Acronyms and Abbreviations

The following table lists the terms utilized within this part of the document.

Acronyms or Abbreviations	Description
<i>3GPP</i>	<b>3rd Generation Partnership Project</b>
<i>CASB</i>	<b>Cloud Access Security Broker</b>
<i>DAST</i>	<b>Dynamic Application Security Testing</b>
<i>DPI</i>	<b>Deep Packet Inspection</b>
<i>IDS</i>	<b>Intrusion Detection Systems</b>
<i>IPS</i>	<b>Intrusion Prevention Systems</b>
<i>IPSec</i>	<b>Internet Protocol Security</b>
<i>LSM</i>	<b>Linux Security Module</b>
<i>MITM</i>	<b>Man In The Middle</b>
<i>OSI</i>	<b>Open Systems Interconnection</b>
<i>SATS</i>	<b>Static Application Security Testing</b>

## Local

Domain	Improvement
Application-Installation-1	Talk about AppFw offline mode.

## Installation

Applications can be delivered and installed with the base image using a special offline-mode provided by the **AppFw**. Apps can also be installed at run time.

During early release, default Apps are installed on the image at first boot.

Domain	Object	Recommendations
Application-Installation-1	AppFw	Provide offline-mode in order to install app with the base image.
Application-Installation-2	Integrity	Allow the installation of applications only if their integrity is good.

## Local

---

### Privilege Management

Application privileges are managed by **Cynara** and the security manager in the **AppFw**. For more details, please refer to the **AppFw** documentation in Platform part.

## App Signature

---

Domain	Improvement
Application-Signature-1	Add content (see secure build in Secure development part).

## Services

---

Domain	Improvement
Application-Services-1	Add content (Which services?).
Application-Services-2	Add Binder.

## Part 7 - Connectivity

---

### Abstract

This part shows different Connectivity attacks on the car.

Domain	Improvement
Connectivity-Abstract-1	Improve abstract.

## Acronyms and Abbreviations

The following table lists the terms utilized within this part of the document.

Acronyms or Abbreviations	Description
<i>ARP</i>	<b>A</b> ddress <b>R</b> esolution <b>P</b> rotocol
<i>BLE</i>	<b>B</b> luetooth <b>L</b> ow <b>E</b> nergy
<i>CAN</i>	<b>C</b> ar <b>A</b> rea <b>N</b> etwork
<i>CCMP</i>	<b>C</b> ounter-Mode/ <b>C</b> BC- <b>M</b> ac <b>P</b> rotocol
<i>EDGE</i>	<b>E</b> nhanced <b>D</b> ata <b>R</b> ates for <b>G</b> SM <b>E</b> volution - Evolution of <b>G</b> PRS
<i>GEA</i>	<b>G</b> PRS <b>E</b> ncryption <b>A</b> lgorithm
<i>GPRS</i>	<b>G</b> eneral <b>P</b> acket <b>R</b> adio <b>S</b> ervice (2,5G, 2G+)
<i>GSM</i>	<b>G</b> lobal <b>S</b> ystem for <b>M</b> obile Communications (2G)
<i>HSPA</i>	<b>H</b> igh <b>S</b> peed <b>P</b> acket <b>A</b> ccess (3G+)
<i>IMEI</i>	<b>I</b> nternational <b>M</b> obile <b>E</b> quipment <b>I</b> dentify
<i>LIN</i>	<b>L</b> ocal <b>I</b> nterconnect <b>N</b> etwork
<i>MOST</i>	<b>M</b> edia <b>O</b> riented <b>S</b> ystem <b>T</b> ransport
<i>NFC</i>	<b>N</b> ear <b>F</b> ield <b>C</b> ommunication
<i>OBD</i>	<b>O</b> n- <b>B</b> oard <b>D</b> iagnostics
<i>PATS</i>	<b>P</b> assive <b>A</b> nti- <b>T</b> heft <b>S</b> ystem
<i>PKE</i>	<b>P</b> assive <b>K</b> eyless <b>E</b> nter
<i>PSK</i>	<b>P</b> hase- <b>S</b> hift <b>K</b> eying
<i>RDS</i>	<b>R</b> adio <b>D</b> ata <b>S</b> ystem
<i>RFID</i>	<b>R</b> adio <b>F</b> requency <b>I</b> dentification
<i>RKE</i>	<b>R</b> emote <b>K</b> eyless <b>E</b> nter
<i>SDR</i>	<b>S</b> oftware <b>D</b> efined <b>R</b> adio
<i>SSP</i>	<b>S</b> ecure <b>S</b> imple <b>P</b> airing
<i>TKIP</i>	<b>T</b> emporal <b>K</b> ey <b>I</b> ntegrity <b>P</b> rotocol
<i>TPMS</i>	<b>T</b> ire <b>P</b> ressure <b>M</b> onitoring <b>S</b> ystem
<i>UMTS</i>	<b>U</b> niversal <b>M</b> obile <b>T</b> elecommunications <b>S</b> ystem (3G)
<i>USB</i>	<b>U</b> niversal <b>S</b> erial <b>B</b> us
<i>WEP</i>	<b>W</b> ired <b>E</b> quivalent <b>P</b> rivacy
<i>WPA</i>	<b>W</b> ifi <b>P</b> rotected <b>A</b> ccess

## Bus

We only speak about the **CAN** bus to take an example, because the different attacks on bus like *FlewRay*, *ByteFlight*, *Most* and *Lin* use retro engineering and the main argument to improve their security is to encrypt data packets. We just describe them a bit:

- **CAN**: Controller Area Network, developed in the early 1980s, is an event-triggered controller network for serial communication with data rates up to one MBit/s. **CAN** messages are classified over their respective identifier. **CAN** controller broadcast their messages to all connected nodes and all receiving nodes decide independently if they process the message.
- **FlewRay**: Is a deterministic and error-tolerant high-speed bus. With a data rate up to 10 MBit/s.
- **ByteFlight**: Is used for safety-critical applications in motor vehicles like air-bags. Byteflight runs at 10Mbps over 2 or 3 wires plastic optical fibers.
- **Most**: Media Oriented System Transport, is used for transmitting audio, video, voice, and control data via fiber optic cables. The speed is, for the synchronous way, up to 24 MBit/s and asynchronous way up to 14 MBit/s. **MOST** messages include always a clear sender and receiver address.
- **LIN**: Local Interconnect Network, is a single-wire subnet work for low-cost, serial communication between smart sensors and actuators with typical data rates up to 20 kBit/s. It is intended to be used from the year 2001 on everywhere in a car, where the bandwidth and versatility of a **CAN** network is not required.

Domain	Tech name	Recommendations
Connectivity-BusAndConnector-Bus-1	CAN	Implement hardware solution in order to prohibit sending unwanted signals.

See [Security in Automotive Bus Systems](#) for more information.

## Connectors

For the connectors, we supposed that they were disabled by default. For example, the **USB** must be disabled to avoid attacks like BadUSB. If not, configure the Kernel to only enable the minimum require **USB** devices. The connectors used to diagnose the car like **OBD-II** must be disabled outside garages.

Domain	Tech name	Recommendations
Connectivity-BusAndConnector-Connectors-1	USB	Must be disabled. If not, only enable the minimum require USB devices.
Connectivity-BusAndConnector-Connectors-2	USB	Confidential data exchanged with the ECU over USB must be secure.
Connectivity-BusAndConnector-Connectors-3	USB	USB Boot on a ECU must be disable.
Connectivity-BusAndConnector-Connectors-4	OBD-II	Must be disabled outside garages.





## Wireless

In this part, we talk about possible remote attacks on a car, according to the different areas of possible attacks. For each communication channels, we describe attacks and how to prevent them with some recommendations. The main recommendation is to always follow the latest updates of these remote communication channels.

Domain	Object	Recommendations
Connectivity-Wireless-1	Update	Always follow the latest updates of remote communication channels.

We will see the following parts:

- [Wifi](#)
- [Bluetooth](#)
- [Cellular](#)
- [Radio](#)
- [NFC](#)

Domain	Improvement
Connectivity-Wireless-1	Add communication channels (RFID, ZigBee?).

For existing automotive-specific means, we take examples of existing system attacks from the *IOActive* document ([A Survey of Remote Automotive Attack Surfaces](#)) and from the ETH document ([Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars](#)).

- [Telematics](#)
- [Passive Anti-Theft System \(PATS\)](#)
- [Tire Pressure Monitoring System \(TPMS\)](#)
- [Remote Keyless Entry/Start \(RKE\)](#)
- [Passive Keyless Entry \(PKE\)](#)

# Wifi

## Attacks

We can differentiate existing attacks on wifi in two categories: Those on **WEP** and those on **WPA**.

- **WEP** attacks:
  - **FMS**: (Fluhrer, Mantin and Shamir attack) is a "Stream cipher attack on the widely used RC4 stream cipher. The attack allows an attacker to recover the key in an RC4 encrypted stream from a large number of messages in that stream."
  - **KoreK**: "Allows the attacker to reduce the key space".
  - **PTW**: (Pyshkin Tews Weinmann attack).
  - **Chopchop**: Found by KoreK, "Weakness of the CRC32 checksum and the lack of replay protection."
  - **Fragmentation**
- **WPA** attacks:
  - **Beck and Tews**: Exploit weakness in **TKIP**. "Allow the attacker to decrypt **ARP** packets and to inject traffic into a network, even allowing him to perform a **DoS** or an **ARP** poisoning".
  - **KRACK**: (K)ey (R)einstallation (A)ttack ([jira AGL SPEC-1017](#)).

## Recommendations

- Do not use **WEP**, **PSK** and **TKIP**.
- Use **WPA2** with **CCMP**.
- Should protect data sniffing.

Domain	Tech name or object	Recommendations
Connectivity-Wireless-Wifi-1	WEP, PSK, TKIP	Disabled
Connectivity-Wireless-Wifi-2	WPA2 and AES-CCMP	Used
Connectivity-Wireless-Wifi-3	WPA2	Should protect data sniffing.
Connectivity-Wireless-Wifi-4	PSK	Changing regularly the password.
Connectivity-Wireless-Wifi-5	Device	Upgraded easily in software or firmware to have the last security update.

See [Wifi attacks WEP WPA](#) and [Breaking wep and wpa \(Beck and Tews\)](#) for more information.

# Bluetooth

## Attacks

- **Bluesnarfing** attacks involve an attacker covertly gaining access to your Bluetooth-enabled device for the purpose of retrieving information, including addresses, calendar information or even the device's **I**nternational **M**obile **E**quipment **I**dentify. With the **IMEI**, an attacker could route your incoming calls to his cell phone.
- **Bluebugging** is a form of Bluetooth attack often caused by a lack of awareness. Similar to bluesnarfing, bluebugging accesses and uses all phone features but is limited by the transmitting power of class 2 Bluetooth radios, normally capping its range at 10-15 meters.
- **Bluejacking** is the sending of unsolicited messages.
- **BLE: Bluetooth Low Energy attacks.**
- **DoS:** Drain a device's battery or temporarily paralyze the phone.

## Recommendations

- Not allowing Bluetooth pairing attempts without the driver's first manually placing the vehicle in pairing mode.
- Monitoring.
- Use **BLE** with caution.
- For v2.1 and later devices using **Secure Simple Pairing (SSP)**, avoid using the "Just Works" association model. The device must verify that an authenticated link key was generated during pairing.

Domain	Tech name	Recommendations
Connectivity-Wireless-Bluetooth-1	BLE	Use with caution.
Connectivity-Wireless-Bluetooth-2	Bluetooth	Monitoring
Connectivity-Wireless-Bluetooth-3	SSP	Avoid using the "Just Works" association model.
Connectivity-Wireless-Bluetooth-4	Visibility	Configured by default as undiscoverable. Except when needed.
Connectivity-Wireless-Bluetooth-5	Anti-scanning	Used, inter alia, to slow down brute force attacks.

See [Low energy and the automotive transformation](#), [Gattacking Bluetooth Smart Devices](#), [Comprehensive Experimental Analyses of Automotive Attack Surfaces](#) and [With Low Energy comes Low Security](#) for more information.

## Cellular

### Attacks

- **IMSI-Catcher**: Is a telephone eavesdropping device used for intercepting mobile phone traffic and tracking location data of mobile phone users. Essentially a "fake" mobile tower acting between the target mobile phone and the service provider's real towers, it is considered a man-in-the-middle (**MITM**) attack.
- Lack of mutual authentication (**GPRS/EDGE**) and encryption with **GEA0**.
- **Fall back** from **UMTS/HSPA** to **GPRS/EDGE** (Jamming against **UMTS/HSPA**).
- 4G **DoS** attack.

### Recommendations

- Check antenna legitimacy.

Domain	Tech name	Recommendations
Connectivity-Wireless-Cellular-1	GPRS/EDGE	Avoid
Connectivity-Wireless-Cellular-2	UMTS/HSPA	Protected against Jamming.

See [A practical attack against GPRS/EDGE/UMTS/HSPA mobile data communications](#) for more information.

## Radio

### Attacks

- Interception of data with low cost material (**SDR** with hijacked DVB-T/DAB for example).

### Recommendations

- Use the **Radio Data System (RDS)** only to send signals for audio output and meta concerning radio.

Domain	Tech name	Recommendations
Connectivity-Wireless-Radio-1	RDS	Only audio output and meta concerning radio.

# NFC

## Attacks

- **MITM:** Relay and replay attack.

## Recommendations

- Should implements protection against relay and replay attacks (Tokens, etc...).
- Disable unneeded and unapproved services and profiles.
- NFC should be use encrypted link (secure channel). A standard key agreement protocol like Diffie-Hellmann based on RSA or Elliptic Curves could be applied to establish a shared secret between two devices.
- Automotive NFC device should be certified by NFC forum entity: The NFC Forum Certification Mark shows that products meet global interoperability standards.
- NFC Modified Miller coding is preferred over NFC Manchester coding.

Domain	Tech name	Recommendations
Connectivity-Wireless-NFC-1	NFC	Protected against relay and replay attacks.
Connectivity-Wireless-NFC-2	Device	Disable unneeded and unapproved services and profiles.

# Cloud

## Download

- **authentication:** Authentication is the security process that validates the claimed identity of a device, entity or person, relying on one or more characteristics bound to that device, entity or person.
- **Authorization:** Parses the network to allow access to some or all network functionality by providing rules and allowing access or denying access based on a subscriber's profile and services purchased.

Domain	Object	Recommendations
Application-Cloud-Download-1	authentication	Must implement authentication process.
Application-Cloud-Download-2	Authorization	Must implement Authorization process.

## Infrastructure

- **Deep Packet Inspection: DPI** provides techniques to analyze the payload of each packet, adding an extra layer of security. **DPI** can detect and neutralize attacks that would be missed by other security mechanisms.
- A **DoS** protection in order to avoid that the Infrastructure is no more accessible for a period of time.
- **Scanning tools** such as **SATS** and **DAST** assessments perform vulnerability scans on the source code and data flows on web applications. Many of these scanning tools run different security tests that stress applications under certain attack scenarios to discover security issues.
- **IDS & IPS:** **IDS** detect and log inappropriate, incorrect, or anomalous activity. **IDS** can be located in the telecommunications networks and/or within the host server or computer. Telecommunications carriers build intrusion detection capability in all network connections to routers and servers, as well as offering it as a service to enterprise customers. Once **IDS** systems have identified an attack, **IPS** ensures that malicious packets are blocked before they cause any harm to backend systems and networks. **IDS** typically functions via one or more of three systems:
  1. Pattern matching.
  2. Anomaly detection.
  3. Protocol behavior.

Domain	Object	Recommendations
Application-Cloud-Infrastructure-1	Packet	Should implement a DPI.
Application-Cloud-Infrastructure-2	DoS	Must implement a DoS protection.
Application-Cloud-Infrastructure-3	Test	Should implement scanning tools like SATS and DAST.
Application-Cloud-Infrastructure-4	Log	Should implement security tools (IDS and IPS).
Application-Cloud-Infrastructure-5	App integrity	Applications must be signed by the code signing authority.

## Transport

For data transport, it is necessary to **encrypt data end-to-end**. To prevent **MITM** attacks, no third party should be able to interpret transported data. Another aspect is the data anonymization in order to protect the leakage of private information on the user or any other third party.

The use of standards such as **IPSec** provides *"private and secure communications over IP networks, through the use of cryptographic security services, is a set of protocols using algorithms to transport secure data over an IP network."*. In addition, **IPSec** operates at the network layer of the **OSI** model, contrary to previous standards that operate at the application layer. This makes its application independent and means that users do not need to configure each application to **IPSec** standards.

**IPSec** provides the services below :

- Confidentiality: A service that makes it impossible to interpret data if it is not the recipient. It is the encryption function that provides this service by transforming intelligible (unencrypted) data into unintelligible (encrypted) data.
- Authentication: A service that ensures that a piece of data comes from where it is supposed to come from.
- Integrity: A service that consists in ensuring that data has not been tampered with accidentally or fraudulently.
- Replay Protection: A service that prevents attacks by re-sending a valid intercepted packet to the network for the same authorization. This service is provided by the presence of a sequence number.
- Key management: Mechanism for negotiating the length of encryption keys between two **IPSec** elements and exchange of these keys.

An additional means of protection would be to do the monitoring between users and the cloud as a **CASB** will provide.

Domain	Object	Recommendations
Application-Cloud-Transport-1	Integrity, confidentiality and legitimacy	Should implement IPSec standards.





## Part 8 - Update (OTA)

---

### Abstract

Updating applications and firmware is essential for the development of new features and even more to fix security bugs. However, if a malicious third party manages to divert its first use, it could alter the functioning of the system and/or applications. The security of the updates is therefore a critical point to evaluate in order to guarantee the integrity, the confidentiality and the legitimacy of the transmitted data.

---

### Acronyms and Abbreviations

The following table lists the terms utilized within this part of the document.

Acronyms or Abbreviations	Description
<i>FOTA</i>	<b>F</b> irmware <b>O</b> ver <b>T</b> he <b>A</b> ir
<i>OTA</i>	<b>O</b> ver <b>T</b> he <b>A</b> ir
<i>SOTA</i>	<b>S</b> oftware <b>O</b> ver <b>T</b> he <b>A</b> ir

## Firmware Over The Air

The firmware update is critical since its alteration back to compromise the entire system. It is therefore necessary to take appropriate protective measures. The principle of verifying chain integrity fulfills much of AGL's security. During a firmware update, it is necessary to update the different signatures to check the integrity of the system.

There is also the constraint of the update time: The system must start quickly and therefore, update itself as quickly. We imagine that the **FOTA** is mainly used in the vehicle maintenance session (e.g. Garage). We will then use no more **FOTA** but a wired update. There is a limit to what can be updated wirelessly. This maintenance update could solve these problems.

Field upgrades can be achieved securely by using a Secure Loader. This loader will authenticate an incoming image (USB, Serial, Network) prior to writing it to the flash memory on the device. It should not be possible to write to flash from bootloader (U-Boot). Note that because USB support is to be disabled within the sbboot/U-Boot code, the board specific implementation of the Secure Loader will have to manage the entire USB initialization, enumeration, and read/write access to the mass storage device.

Domain	Object	Recommendations
Update-FOTA-1	Integrity, confidentiality and legitimacy	Must be secure.

Different possible type of **FOTA**:

- Package-based like rpm, dpkg:
  - + Simple.
  - - Power-off.
  - - Dependency.
- Full file system updates:
  - + Robust.
  - - Tends device-specific.
  - - Need rsync or similar.
- Atomic differential:
  - + Robust.
  - + Minimal bandwidth consumption.
  - + Easy reusable.
  - - Physically one file system (Corruption -> unbootable system).
  - - No rollback logic.

## Software Over The Air

---

**SOTA** is made possible by **AppFw** (See Platform part). It will be possible to manage in a simple way the packets (i.g. Android like).

Domain	Improvement
Update-SOTA-1	Part to complete.

## Part 9 - Secure development

In order to save a lot of time in code auditing, developers must follow coding guidelines.

### Secure build

#### Kernel build

Tools like:

- [Code optimisation](#).
- [Kernel Drivers test](#) with [docs](#).

Domain	Improvement
SecureDev-SecureBuild-1	Add content.

### App/Widget signatures

Domain	Improvement
SecureDev-Signatures-1	Add content.

### Code audit

These tools are used to check the correct implementation of functionalities and compliance with related good practices.

- [Continuous Code Quality](#).

Domain	Improvement
SecureDev-CodeAudit-1	Add CVE analyser.
SecureDev-CodeAudit-2	<a href="#">OSSTMM</a> .

### SATS

- [RATS](#) (Maybe to old).
- [Flaw Finder](#).
- [wiki list](#).
- [Mathematical approach](#).

It is necessary to verify that the application code does not use functions that are depreciated and recognized as unsecured or cause problems.

## DATS

- [wiki list](#).

## Annexes

---

The first part resumed all the configurations you must implement without any explications since all the explanations are given as and when in the document.

The second one allows to visualize all the todo notes in order to have a global vision of the possible improvements of the document.

## Config notes

Domain	Object	Recommendations
Hardware-Integrity-1	Bootloader	Must control bootloader integrity.
Hardware-Integrity-2	Board	Must use a HSM.
Hardware-Integrity-3	RTC	Must not be alterable.

Domain	Object	Recommendations
Hardware-Certificate-1	System	Shall allow storing dedicated certificates.
Hardware-Certificate-2	ECU	The ECU must verify the certification authority hierarchy.
Hardware-Certificate-3	System	Allow the modification of certificates only if the source can be authenticated by a certificate already stored or in the higher levels of the chain of trust.

Domain	Object	Recommendations
Hardware-Memory-1	ECU	The ECU shall never expose the unencrypted key in RAM when using cryptographic keys.
Hardware-Memory-2	Bootloader	Internal NVM only
Hardware-Module-3	-	HSM must be used to secure keys.

Domain	Variable / Config name	Value
Boot-Image-Selection-1	CONFIG_BOOTDELAY	-2
Boot-Image-Selection-2	bootdelay	-2

Domain	Config name	State
Boot-Image-Authenticity-1	CONFIG_FIT	Enable
Boot-Image-Authenticity-2	CONFIG_FIT_SIGNATURE	Enable
Boot-Image-Authenticity-3	CONFIG_RSA	Enable
Boot-Image-Authenticity-4	CONFIG_OF_CONTROL	Enable
Boot-Image-Authenticity-5	CONFIG_OF_SEPARATE	Enable
Boot-Image-Authenticity-6	CONFIG_DEFAULT_DEVICE_TREE	Enable

Domain	Communication modes	State
Boot-Communication-1	USB	Disabled and Compiled-out if not required.



Boot-Communication-2	USB	Else, Kernel should be configured to only enable the minimum required USB devices and filesystems should be treated with special care.
Boot-Communication-3	Ethernet	<i>Disabled</i>
Boot-Communication-4	U-boot and sboot DOCSIS	<i>Disabled</i>
Boot-Communication-5	Serial ports	<i>Disabled</i>

Domain	Config name	State
Boot-Communication-USB-1	CONFIG_CMD_USB	<i>Not defined</i>
Boot-Communication-USB-2	CONFIG_USB_UHCI	<i>Not defined</i>
Boot-Communication-USB-3	CONFIG_USB_KEYBOARD	<i>Not defined</i>
Boot-Communication-USB-4	CONFIG_USB_STORAGE	<i>Not defined</i>
Boot-Communication-USB-5	CONFIG_USB_HOST_ETHER	<i>Not defined</i>

Domain	Communication modes	State
Boot-Communication-1	Network interfaces	Preferably <i>no network interface is allowed</i> , otherwise, restrict the services to those used.

Domain	Object	Recommendations
Boot-Communication-1	Services , ports and devices	Restrict the services , ports and devices to those used.

Domain	Command name	State
Boot-Communication-Flash-1	do_nand	<i>Disable</i>

Domain	Config name	Value
Boot-Consoles-Serial-1	CONFIG_SILENT_CONSOLE	<i>Disable</i>
Boot-Consoles-Serial-2	CONFIG_SYS_DEVICE_NULLDEV	<i>Disable</i>
Boot-Consoles-Serial-3	CONFIG_SILENT_CONSOLE_UPDATE_ON_RELOC	<i>Disable</i>

Domain	Environment variable name	State
Boot-Consoles-Serial-1	INC_DEBUG_PRINT	<i>Not defined</i>

Domain	Config name	State
Boot-Consoles-Variables-1	CONFIG_ENV_IS_IN_MMC	<i>#undef</i>
Boot-Consoles-Variables-2	CONFIG_ENV_IS_IN_EEPROM	<i>#undef</i>
Boot-Consoles-Variables-3	CONFIG_ENV_IS_IN_FLASH	<i>#undef</i>
Boot-Consoles-Variables-4	CONFIG_ENV_IS_IN_DATAFLASH	<i>#undef</i>

Boot-Consoles-Variables-5	CONFIG_ENV_IS_IN_FAT	#undef
Boot-Consoles-Variables-6	CONFIG_ENV_IS_IN_NAND	#undef
Boot-Consoles-Variables-7	CONFIG_ENV_IS_IN_NVRAM	#undef
Boot-Consoles-Variables-8	CONFIG_ENV_IS_IN_ONENAND	#undef
Boot-Consoles-Variables-9	CONFIG_ENV_IS_IN_SPI_FLASH	#undef
Boot-Consoles-Variables-10	CONFIG_ENV_IS_IN_REMOTE	#undef
Boot-Consoles-Variables-11	CONFIG_ENV_IS_IN_UBI	#undef
Boot-Consoles-Variables-12	CONFIG_ENV_IS_NOWHERE	#define

Domain	Command name	State
Boot-Consoles-MemDump-1	md	Disabled
Boot-Consoles-MemDump-2	mm	Disabled
Boot-Consoles-MemDump-3	nm	Disabled
Boot-Consoles-MemDump-4	mw	Disabled
Boot-Consoles-MemDump-5	cp	Disabled
Boot-Consoles-MemDump-6	mwc	Disabled
Boot-Consoles-MemDump-7	mdc	Disabled
Boot-Consoles-MemDump-8	mtest	Disabled
Boot-Consoles-MemDump-9	loopw	Disabled

Domain	Object	Recommendations
Kernel-General-MAC-1	SMACK	Must implement a Mandatory Access Control.

Domain	Config name	Value
Kernel-General-kexec-1	CONFIG_KEXEC	n

Domain	Config name	Value
Kernel-General-IPAutoConf-1	CONFIG_IP_PNP	n

Domain	Config name	Value
Kernel-General-SysCtl_SysCall-1	CONFIG_SYSCTL_SYSCALL	n

Domain	Config name	Value
Kernel-General-LegacyLinux-1	CONFIG_USELIB	n

Domain	Config name	Value
Kernel-General-FirmHelper-1	CONFIG_FW_LOADER_USER_HELPER	n

Domain	Config name	Value
Kernel-General-PanicOnOOPS-1	CONFIG_PANIC_ON_OOPS	y

Domain	Config name	Value
--------	-------------	-------

Kernel-General-SocketMon-1	CONFIG_PACKET_DIAG	n
Kernel-General-SocketMon-2	CONFIG_UNIX_DIAG	n

Domain	Config name	Value
Kernel-General-BPF_JIT-1	CONFIG_BPF_JIT	n

Domain	Config name	Value
Kernel-General-ModuleSigning-1	CONFIG_MODULE_SIG_FORCE	y

Domain	Object	State
Kernel-General-Drivers-1	USB	Disabled
Kernel-General-Drivers-2	PCMCIA	Disabled
Kernel-General-Drivers-3	Other hotplug bus	Disabled

Domain	compiler and linker options	State
Kernel-General-IndependentExec-1	-pie -fpic	Enable

Domain	compiler and linker options	State
Kernel-General-OverwriteAttacks-1	-z,relro	Enable
Kernel-General-OverwriteAttacks-2	-z,nov	Enable

Domain	compiler and linker options	State
Kernel-General-LibraryLinking-1	-static	Enable

Domain	Config name	Value
Kernel-Memory-RestrictAccess-1	CONFIG_DEVKMEM	n

Domain	Config name	Value
Kernel-Memory-CoreDump-1	CONFIG_PROC_KCORE	n

Domain	Config name	Value
Kernel-Memory-Swap-1	CONFIG_SWAP	n

Domain	Config name	Value
Kernel-Memory-LoadAllSymbols-1	CONFIG_KALLSYMS	n
Kernel-Memory-LoadAllSymbols-2	CONFIG_KALLSYMS_ALL	n

Domain	Config name	Value
Kernel-Memory-Stack-1	CONFIG_CC_STACKPROTECTOR	y

Other defenses include things like shadow stacks.

Domain	Config name	Value
Kernel-Memory-Access-1	CONFIG_DEVKMEM	n

Domain	Config name	Value
Kernel-Memory-CrossMemAttach-1	CROSS_MEMORY_ATTACH	n

Domain	compiler and linker options	State
Kernel-Memory-StackSmashing-1	-fstack-protector-all	Enable

Domain	compiler and linker options	Value
Kernel-Memory-BufferOverflows-1	-D_FORTIFY_SOURCE	2

Domain	Config name	Value
Kernel-Consoles-Serial-1	CONFIG_SERIAL_8250	n
Kernel-Consoles-Serial-2	CONFIG_SERIAL_8250_CONSOLE	n
Kernel-Consoles-Serial-3	CONFIG_SERIAL_CORE	n
Kernel-Consoles-Serial-4	CONFIG_SERIAL_CORE_CONSOLE	n

Domain	Config name	Value
Kernel-Consoles-CommandLine-1	CONFIG_CMDLINE_BOOL	y
Kernel-Consoles-CommandLine-2	CONFIG_CMDLINE	"insert kernel command line here"
Kernel-Consoles-CommandLine-3	CONFIG_CMDLINE_OVERRIDE	y

Domain	Config name	Value
Kernel-Consoles-KDBG-1	CONFIG_KGDB	n

Domain	Config name	Value
Kernel-Consoles-SysRQ-1	CONFIG_MAGIC_SYSRQ	n

Domain	Config name	Value
Kernel-Consoles-BinaryFormat-1	CONFIG_BINFMT_MISC	n

Domain	Config name	Value
Kernel-Debug-Symbols-1	CONFIG_DEBUG_INFO	n

Domain	Config name	Value
Kernel-Debug-Kprobes-1	CONFIG_KPROBES	n

Domain	Config name	Value
Kernel-Debug-Tracing-1	CONFIG_FTRACE	n

Domain	Config name	Value
Kernel-Debug-Profiling-1	CONFIG_OPROFILE	n
Kernel-Debug-Profiling-2	CONFIG_PROFILING	n

Domain	Config name	Value
Kernel-Debug-OOPSONBUG-1	CONFIG_DEBUG_BUGVERBOSE	n

Domain	Config name	Value
Kernel-Debug-Dev-1	CONFIG_DEBUG_KERNEL	n
Kernel-Debug-Dev-2	CONFIG_EMBEDDED	n

Domain	Config name	Value
Kernel-Debug-FileSystem-1	CONFIG_DEBUG_FS	n

Domain	Config name	Value
Kernel-Debug-BUG-1	CONFIG_BUG	n

Domain	Config name	Value
Kernel-Debug-CoreDumps-1	CONFIG_COREDUMP	n

Domain	File name	Value
Kernel-Debug-AdressDisplay-1	/proc/sys/kernel/kptr_restrict	1

Domain	File or Directorie name	State
Kernel-Debug-AdressDisplay-1	/boot/vmlinuz*	Readable Only for root user
Kernel-Debug-AdressDisplay-2	/boot/System.map*	Readable Only for root user
Kernel-Debug-AdressDisplay-3	/sys/kernel/debug/	Readable Only for root user
Kernel-Debug-AdressDisplay-4	/proc/slabinfo	Readable Only for root user

Domain	File name	Value
Kernel-Debug-DMESG-1	/proc/sys/kernel/dmesg_restrict	1

Domain	Config name	Value
Kernel-Debug-Config-1	CONFIG_IKCONFIG	n

Domain	Config name	Value
Kernel-FileSystems-NFS-1	CONFIG_NFSD	n
Kernel-FileSystems-NFS-2	CONFIG_NFS_FS	n

Domain	Partition	Value
Kernel-FileSystems-Mount-1	/boot	nosuid , nodev and noexec .
Kernel-FileSystems-Mount-2	/var & /tmp	In /etc/fstab or vfstab , add nosuid , nodev and noexec .
Kernel-FileSystems-Mount-3	Non-root local	If type is ext2 or ext3 and mount point not '/', add nodev .
Kernel-FileSystems-Mount-4	Removable storage	Add nosuid , nodev and noexec .
Kernel-FileSystems-Mount-5	Temporary storage	Add nosuid , nodev and noexec .

Kernel-FileSystems-Mount-6	/dev/shm	Add <code>nosuid</code> , <code>nodev</code> and <code>noexec</code> .
Kernel-FileSystems-Mount-7	/dev	Add <code>nosuid</code> and <code>noexec</code> .

Domain	Config name	State or Value
Kernel-FileSystems-Mount-1	CONFIG_DEVTMPFS_MOUNT	<i>Disabled</i> or add remount with <code>noexec</code> and <code>nosuid</code> to system startup.

Domain	Label name	Recommendations
Kernel-MAC-Floor-1	^	Only for privileged system services.
Kernel-MAC-Floor-2	*	Used for device files or <code>/tmp</code> Access restriction via DAC.

Domain	Label name	Recommendations
Kernel-MAC-System-1	System	Process should write only to file with transmute attribute.
Kernel-MAC-System-2	System::run	Files are created with the directory label from user and system domain (transmute) Lock is implicit with <code>w</code> .
Kernel-MAC-System-3	System::Shared	Files are created with the directory label from system domain (transmute) User domain has locked privilege.
Kernel-MAC-System-4	System::Log	Some limitation may impose to add <code>w</code> to enable append.
Kernel-MAC-System-5	System::Sub	Isolation of risky Subsystem.

Domain	Label name	Recommendations
Kernel-MAC-System-1	User::Pkg::\$AppID	Only one Label is allowed per App. A data directory is created by the AppFw in <code>rwX</code> mode.
Kernel-MAC-System-2	User::Home	AppFw needs to create a directory in <code>/home/\$USER/App-Shared</code> at first launch if not present with label <code>app-data</code> access is <code>User::App-Shared</code> without transmute.
Kernel-MAC-System-3	User::App-Shared	Shared space between all App running for a given user.

Domain	Object	Recommendations
Platform-SystemD-1	Security model	Use Namespaces for containerization.
Platform-SystemD-2	Security model	Use CGroups to organise processes.

Domain	Object	Recommendations
Platform-DBus-1	Security model	Use D-Bus as IPC.
Platform-DBus-2	Security model	Apply D-BUS security patches: <a href="#">D-Bus CVE</a>

Domain	Tool name	State

Platform-Utilities-1	connman	Used as a connection manager.
Platform-Utilities-2	bluez	Used as a Bluetooth manager.
Platform-Utilities-3	gststreamer	Used to manage multimedia file format.
Platform-Utilities-4	alsa	Used to provides an API for sound card device drivers.

Domain	Object	Recommendations
Platform-AGLFW-AppFw-1	Security model	Use the AppFw as Security model.

Domain	Object	Recommendations
Platform-AGLFW-Cynara-1	Permissions	Use Cynara as policy-checker service.

Domain	Tool name	State
Platform-Utilities-1	busybox	Used to provide a number of tools. Do not compile development tools.

Domain	Utility name and normal path	State
Platform-Utilities-1	chgrp in /bin/chgrp	Disabled
Platform-Utilities-2	chmod in /bin/chmod	Disabled
Platform-Utilities-3	chown in /bin/chown	Disabled
Platform-Utilities-4	dmesg in /bin/dmesg	Disabled
Platform-Utilities-5	Dnsdomainname in /bin/dnsdomainname	Disabled
Platform-Utilities-6	dropbear , Remove "dropbear" from /etc/init.d/rcs	Disabled
Platform-Utilities-7	Editors in (vi) /bin/vi	Disabled
Platform-Utilities-8	find in /bin/find	Disabled
Platform-Utilities-9	gdbserver in /bin/gdbserver	Disabled
Platform-Utilities-10	hexdump in /bin/hexdump	Disabled
Platform-Utilities-11	hostname in /bin/hostname	Disabled
Platform-Utilities-12	install in /bin/install	Disabled
Platform-Utilities-13	iostat in /bin/iostat	Disabled
Platform-Utilities-14	killall in /bin/killall	Disabled
Platform-Utilities-15	klogd in /sbin/klogd	Disabled
Platform-Utilities-16	logger in /bin/logger	Disabled
Platform-Utilities-17	lsmod in /sbin/lsmod	Disabled
Platform-Utilities-18	pmap in /bin/pmap	Disabled
Platform-Utilities-19	ps in /bin/ps	Disabled
Platform-Utilities-20	ps in /bin/ps	Disabled
Platform-Utilities-21	rpm in /bin/rpm	Disabled
Platform-Utilities-22	SSH	Disabled
Platform-Utilities-23	stbhotplug in /sbin/stbhotplug	Disabled

Platform-Utilities-24	strace in /bin/trace	Disabled
Platform-Utilities-25	su in /bin/su	Disabled
Platform-Utilities-26	syslogd in (logger) /bin/logger	Disabled
Platform-Utilities-27	top in /bin/top	Disabled
Platform-Utilities-28	UART in /proc/tty/driver/	Disabled
Platform-Utilities-29	which in /bin/which	Disabled
Platform-Utilities-30	who and whoami in /bin/whoami	Disabled
Platform-Utilities-31	awk (busybox)	Enabled
Platform-Utilities-32	cut (busybox)	Enabled
Platform-Utilities-33	df (busybox)	Enabled
Platform-Utilities-34	echo (busybox)	Enabled
Platform-Utilities-35	fdisk (busybox)	Enabled
Platform-Utilities-36	grep (busybox)	Enabled
Platform-Utilities-37	mkdir (busybox)	Enabled
Platform-Utilities-38	mount (vfat) (busybox)	Enabled
Platform-Utilities-39	printf (busybox)	Enabled
Platform-Utilities-40	sed in /bin/sed (busybox)	Enabled
Platform-Utilities-41	tail (busybox)	Enabled
Platform-Utilities-42	tee (busybox)	Enabled
Platform-Utilities-43	test (busybox)	Enabled

Domain	Object	Recommendations
Platform-Users-root-1	Main application	Should not execute as root.
Platform-Users-root-2	UI	Should run in a context on a user with no capability.

Domain	utility name	State
Platform-Users-root-3	login	Not allowed
Platform-Users-root-4	su	Not allowed
Platform-Users-root-5	ssh	Not allowed
Platform-Users-root-6	scp	Not allowed
Platform-Users-root-7	sftp	Not allowed

Domain	Object	Recommendations
Application-Installation-1	AppFw	Provide offline-mode in order to install app with the base image.
Application-Installation-2	Integrity	Allow the installation of applications only if their integrity is good.



Domain	Tech name	Recommendations
Connectivity-BusAndConnector-Bus-1	CAN	Implement hardware solution in order to prohibit sending unwanted signals.

Domain	Tech name	Recommendations
Connectivity-BusAndConnector-Connectors-1	USB	Must be disabled. If not, only enable the minimum require USB devices.
Connectivity-BusAndConnector-Connectors-2	USB	Confidential data exchanged with the ECU over USB must be secure.
Connectivity-BusAndConnector-Connectors-3	USB	USB Boot on a ECU must be disable.
Connectivity-BusAndConnector-Connectors-4	OBD-II	Must be disabled outside garages.

Domain	Object	Recommendations
Connectivity-Wireless-1	Update	Always follow the latest updates of remote communication channels.

Domain	Tech name or object	Recommendations
Connectivity-Wireless-Wifi-1	WEP, PSK, TKIP	Disabled
Connectivity-Wireless-Wifi-2	WPA2 and AES-CCMP	Used
Connectivity-Wireless-Wifi-3	WPA2	Should protect data sniffing.
Connectivity-Wireless-Wifi-4	PSK	Changing regularly the password.
Connectivity-Wireless-Wifi-5	Device	Upgraded easily in software or firmware to have the last security update.

Domain	Tech name	Recommendations
Connectivity-Wireless-Bluetooth-1	BLE	Use with caution.
Connectivity-Wireless-Bluetooth-2	Bluetooth	Monitoring
Connectivity-Wireless-Bluetooth-3	SSP	Avoid using the "Just Works" association model.
Connectivity-Wireless-Bluetooth-4	Visibility	Configured by default as undiscoverable. Except when needed.
Connectivity-Wireless-Bluetooth-5	Anti-scanning	Used, inter alia, to slow down brute force attacks.

Domain	Tech name	Recommendations
Connectivity-Wireless-Cellular-1	GPRS/EDGE	Avoid
Connectivity-Wireless-Cellular-2	UMTS/HSPA	Protected against Jamming.

Domain	Tech name	Recommendations
Connectivity-Wireless-Radio-1	RDS	Only audio output and meta concerning radio.

Domain	Tech name	Recommendations
Connectivity-Wireless-NFC-1	NFC	Protected against relay and replay attacks.
Connectivity-Wireless-NFC-2	Device	Disable unneeded and unapproved services and profiles.

Domain	Object	Recommendations
Application-Cloud-Download-1	authentication	Must implement authentication process.
Application-Cloud-Download-2	Authorization	Must implement Authorization process.

Domain	Object	Recommendations
Application-Cloud-Infrastructure-1	Packet	Should implement a DPI.
Application-Cloud-Infrastructure-2	DoS	Must implement a DoS protection.
Application-Cloud-Infrastructure-3	Test	Should implement scanning tools like SATS and DAST.
Application-Cloud-Infrastructure-4	Log	Should implement security tools (IDS and IPS).
Application-Cloud-Infrastructure-5	App integrity	Applications must be signed by the code signing authority.

Domain	Object	Recommendations
Application-Cloud-Transport-1	Integrity, confidentiality and legitimacy	Should implement IPSec standards.

Domain	Object	Recommendations
Update-FOTA-1	Integrity, confidentiality and legitimacy	Must be secure.

## Todo notes

Domain	Improvement
Boot-Abstract-1	More generic and add examples (The chain of trust).

Domain	Improvement
Boot-Abstract-1	Review the definition of the "boot loader".

Domain	Improvement
Boot-Consoles-1	Secure loader: No reference earlier?

Domain	Improvement
Hypervisor-Abstract-1	Complete Hypervisor part ( <a href="#">jailhouse</a> / <a href="#">KVM</a> / <a href="#">Xen</a> ).

Domain	Improvement
Kernel-MAC-1	Add MAC config note.

Domain	Improvement
Kernel-General-IndependentExec-1	Kernel or/and platform part ?

Domain	Improvement
Kernel-General-LibraryLinking-1	Keep this part?

Domain	Improvement
Platform-Services-1	SystemD ?
Platform-Services-2	Secure daemon ?

Domain	Improvement
Platform-Users-Capabilities-1	Kernel or Platform-user?
Platform-Users-Capabilities-2	Add config note.

Domain	Improvement
Application-Installation-1	Talk about AppFw offline mode.

Domain	Improvement
Application-Signature-1	Add content (see secure build in Secure development part).

Domain	Improvement
Application-Services-1	Add content (Which services?).
Application-Services-2	Add Binder.

Domain	Improvement
--------	-------------

Connectivity-Abstract-1

Improve abstract.

Domain	Improvement
Connectivity-Wireless-1	Add communication channels (RFID, ZigBee?).

Domain	Improvement
Update-SOTA-1	Part to complete.

Domain	Improvement
SecureDev-SecureBuild-1	Add content.

Domain	Improvement
SecureDev-Signatures-1	Add content.

Domain	Improvement
SecureDev-CodeAudit-1	Add CVE analyser.
SecureDev-CodeAudit-2	<a href="#">OSSTMM</a> .