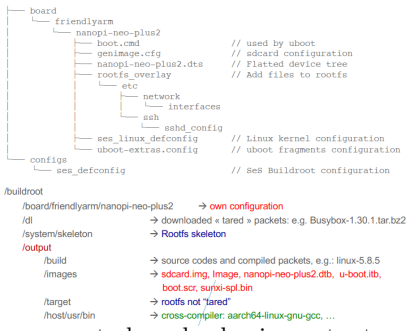
Buildroot

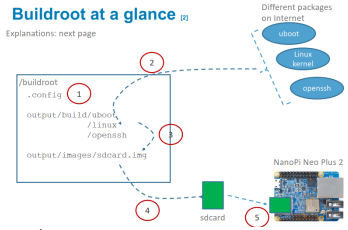
1. D'expliquer les principaux répertoires de buildroot



Ce qui est manquant dans le dossier output sera recompilé lorsque la commande make est lancée (ou alors en faisant la commande make <package>-rebuild

2. D'expliquer le principe de fonctionnement de buildroot

Générateur de Linux embarqué avec le système de cross-compilation



1) The /buildroot/.config file contains the NanoPi Neo Plus 2 buildroot configuration
2) During the make, the configured packages sources files are downloaded to the directories output/uboot, ...
3) At the end of the make, the sdcard.img file is created. This file containts the bootloader, kernel, rootfs, ...
4) The sdcard is flashed with the sdcard.img file
5) The sdcard is put in the NanoPi Neo Plus 2

3. D'expliquer la configuration de buildroot pour un hardware donné

The Buildroot configuration is contained in two files : .config and xxx_defconfig
- Buildroot uses Kconfig like the Linux kernel
- The .config file is a full default config file with more 4000 lines
- The defconfig stores only the values for options for which the non-default value is chosen. It is a small file

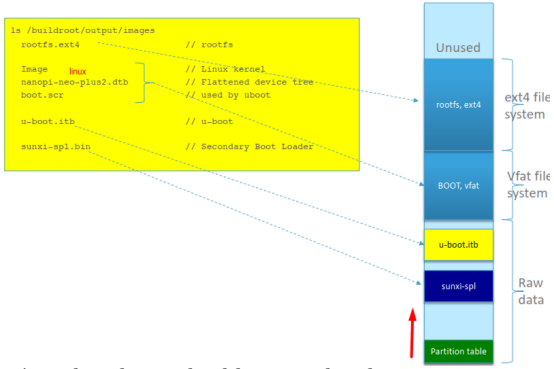
4. D'expliquer comment faire un patch et appliquer ce patch dans buildroot

Initialiser un repo git dans le folder où on va faire les modifs
Commit
Faire les modifs puis commit
Demander la diff entre les commit
git format-patch main –o my_patches
Faire un dossier my_patches dans board .. nano et quand on fait git format-patch .. il va enregistrer le patch dans notre dossier

5. D'expliquer comment configurer, compiler buildroot, u-boot, kernel

make menuconfig to config buildroot
make linux-menuconfig to config linux kernel
make uboot-menuconfig to config uboot
La commande make permet de compiler u-boot et buildroot
make linux-rebuild compile le linux

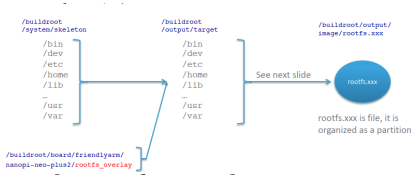
6. D'expliquer comment la SD-Card est générée



Pour créer sdcard.img, buildroot utilise le script genimage.sh fichier : /buildroot/board/friendlyarm/scripts/genimages.cfg file

7. D'expliquer comment le rootfs est généré

- A rootfs skeleton is in the directory /buildroot/system/skeleton
- This skeleton is copied to the pseudo rootfs directory /buildroot/output/target
- It is possible to add files, directories with rootfs_overlay
- After the make command, the pseudo rootfs is populated and copied to one file in this directory : /buildroot/output/image/rootfs.xxx (xxx can be ext4, squashfs, ...)



8. D'expliquer le rootfs_overlay

Le dossier rootfs_overlay permet d'ajouter des fichiers au rootfs (executables cross-compilé, fichiers de configuration)

9. Savoir installer un nouveau package dans buildroot

(info dans LinuxHardening)
Known buildroot packages are in the directory /buildroot/packages
- Example with a generic foo package
- The directory /buildroot/packages/foo contains mainly these files :

— Config.in file, written in kconfig language, describing the configuration options for the package

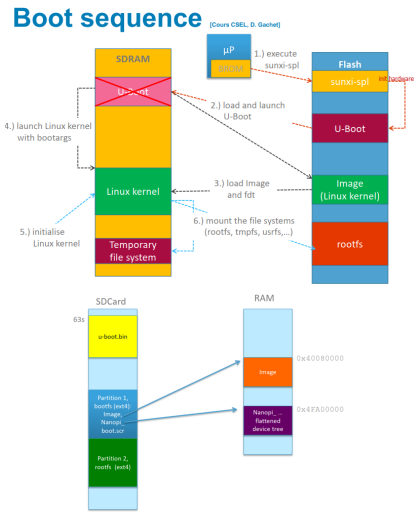
— foo.mk file describing where to fetch the source, how to build and install it, etc.
Optional foo.hash file, providing hashes to check the integrity of the downloaded tarballs and license files

— Sxx_foo file, it is the start script for the foo package

U-boot

10.D'expliquer le démarrage du NanoPi

Le démarrage du NanoPi NEO Plus2 se décompose en 6 phases :
- Lorsque le μP est mis sous tension, le code stocké dans son BROM va charger dans ses 32KiB de SRAM interne le firmware « sunxi-spl » stocké dans le secteur no 16 de la carte SD / eMMC et l'exécuter.
- Le firmware « sunxi-spl » (Secondary Program Loader) initialise les couches basses du μP, puis charge l'U-Boot dans la RAM du μP avant de le lancer.
- L'U-Boot va effectuer les initialisations hardware nécessaires (horloges, contrôleurs, ...) avant de charger l'image non compressées du noyau Linux dans la RAM, le fichier «Image», ainsi que le fichier de configuration FDT (flattened device tree).
- L'U-Boot lancera le noyau Linux en lui passant les arguments de boot (bootargs).
- Le noyau Linux procédera à son initialisation sur la base des bootargs et des éléments de configuration contenus dans le fichier FDT (sun50i-h5-nanopi-neoplus2.dtb).
- Le noyau Linux attachera les systèmes de fichiers (rootfs, tmpfs, usrfs, ...) et poursuivra son exécution.



11. De connaître, expliquer les principales commandes de u-boot utilisées durant le démarrage

Si on appuie sur une touche pendant le démarrage, on entre dans le mode u-boot

boot load the Linux kernel, Image file, the FDT (Flattened Device Tree) and start Linux *booti* permet de lancer d'image linux.

mmc mmc(MultiMediaCard) sub system

printenv print environement variables

```
ext2load- load binary file from a Ext2 filesystem
ext2ls  - list files in a directory (default /)
ext4load- load binary file from a Ext4 filesystem
ext4ls  - list files in a directory (default /)
ext4size- determine a file's size

fatinfo - print information about filesystem
fatload - load binary file from a dos filesystem
fatls   - list files in a directory (default /)
fatmkdir- create a directory
fatrm   - delete a file
fatsize - determine a file's size
fatwrite- write file into a dos filesystem
```

12. De savoir comment configurer u-boot

On configure avec make uboot-menuconfig puis on effectue la compilation avec une des deux manières :

1. make uboot-rebuild
2. supprimer les fichiers puis make

La configuration de u-boot est stockée dans .config

13. D'expliquer comment améliorer la sécurité de u-boot

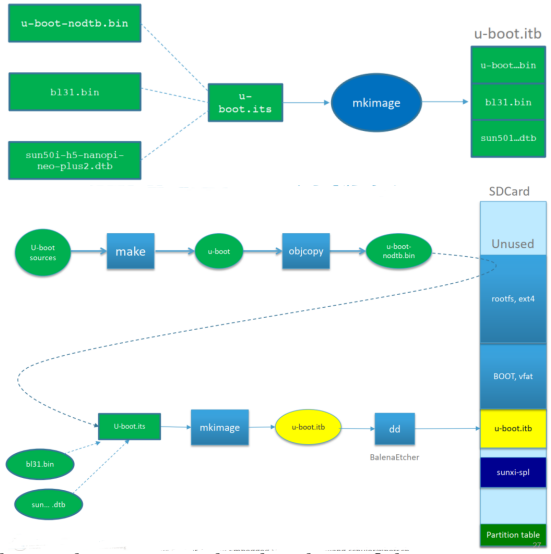
Il est possible d'ajouter deux options :

- suppress the –g option : delete the debug information
- -fstack-protector-all : adds extra code to check buffer overflows, such as stack smashing attacks. This is done by adding a guard variable to functions. This variable is called Canary

14. De connaître les différentes étapes pour la création de l'image de u-boot.itb

mkimage reads u-boot.its and builds u-boot.itb

- u-boot-nodtb.bin : u-boot code. Created during command make. Raw file with only the loadable sections of the u-boot file
- bl31.bin : trust zone.
- sun501-h5 . . . .dtb : Flattened device tree (device Tree Blob)



Le fichier u-boot.its va chercher les 3 fichiers mentionnés ci-dessus :

```
cat u-boot.its
   /dts-v1/;
   / {
      description = "Configuration to load ATF before U-Boot";
      #address-cells = <1>;
      images {
            uboot {
               description = "U-Boot (64-bit)";
               data = /incbin/("u-boot-nodtb.bin");
               type = "standalone";
               arch = "arm64";
               compression = "none";
               load = <0x4a000000>;
            };
            atf {
               description = "ARM Trusted Firmware";
               data = /incbin/("/buildroot/output/images/bl31.bin");
               type = "firmware";
               arch = "arm64";
               compression = "none";
               load = <0x44000>;
               entry = <0x44000>;
            };
         fdt_1 {
            description = "sun50i-h5-nanopi-neo-plus2";
            data = /incbin/("arch/arm/dts/sun50i-h5-nanopi-neo-plus2.dtb");
            type = "flat_dt";
            compression = "none";
         }
   };
   configurations {
         default = "config_1";

         config_1 {
            description = "sun50i-h5-nanopi-neo-plus2";
            firmware = "uboot";
            loadables = "atf";
            fdt = "fdt_1";
         };
```

15. Savoir ce que fait la commande strip sur un fichier elf

Un fichier elf c'est un fichier executable linux dans lequel il y a des informations comme une table de symbole et des informations de débugging. Quand on strip un elf, on enlève ses infos ce qui rend le fichier plus petit mais surtout beaucoup plus dur a reverse engineer (++ sécurité)

16. De connaître les différentes étapes pour la création de uImage

Préparation des fichiers nécessaires : Il est nécessaire de disposer des fichiers suivants : Le noyau Linux compilé (vmlinux ou zImage), le système de fichiers racine (rootfs), le script de démarrage (bootscript)

Utilisation de l'outil mkimage : Cet outil est fourni avec le noyau Linux et permet de créer l'image uImage à partir des fichiers précédemment préparés.

La commande générale pour créer une image uImage est :

$mkimage -A < architecture > -O < os > -T < type > -C < compression > -a < load_address > -e < entry\_point > -n < image\_name > -d < image\_file >< uImage\_file >$

Une fois l'image uImage créée, elle doit être copiée sur la mémoire flash du système embarqué, généralement en utilisant l'outil de programmation de mémoire flash spécifique à la plateforme cible.

17. De connaître l'utilité du Flattened Device Tree

It is a file which contains the hardware description. Linux uses it for its configuration

FDT has two files :

- .dts : Device Tree Source, it is an ascii file
- .dtb : Device Tree Blob, it is a binary file

After the introduction of FTD with the kernel 2.6, a new binary file format was created : FIT (Flattened Image Tree). This format allows to insert different files into a single file

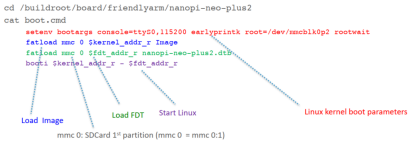
18. De connaître de manière générale le mapping de la SDCard

19. D'expliquer le fichier boot.scr

Le fichier boot.scr est utilisé par u-boot pour charger le kernel Linux. Il est créé avec la commande :

mkimage -C none -A arm64 -T script -d board/friendlyarm/nanopi-neoplus2/boot.cmd out-

put/images/boot.scr



Compilation du noyau

20. De connaître les principaux répertoires du noyau Linux

This directory has these main sub-directories :
arch Hardware dependent code
block Generic functions for the block devices
crypto Cryptographic algo. used in the kernel
Documentation Documentation about the kernel
drivers All drivers known by the kernel
fs All filesystem know by the kernel
include kernel include files

init Init code (function start_kernel)
ipc Interprocess communication
kernel Kernel code, scheduler, mutex, ...
lib different libraries used by the kernel
mm Memory management
net Different protocols, IPv4, IPv6, bluetooth, ...
samples Different examples, kobject, kfifo, ...
security Encrypted keys, SELinux, ...
sound Sound support for Linux kernel
virt Kernel-based virtual machine
This directory has these main files :
vmlinux Linux kernel, ELF format, ARM aarch64
.config Linux kernel configuration
.config.old Old Linux kernel configuration
Kconfig Configuration for the make linux-xconfig
Makefile makefile

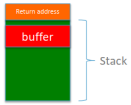
21. De connaître les principales méthodes pour sécuriser le noyau Linux

- -Overflow detection : activer -fstack-protector. Protection contre erreurs de overflow de mémoire.
- Platform selection : La désactivation des platforms non utilisés permettent de restreindre d'accès au linux embarqué
- Kernel .config support : Désactiver cette option va faire que le fichier .config n'ira pas dans l'image envoyée sur la cible. Les configurations du linux embarquée ne seront donc pas disponibles sur la cible
- Randomize espace : activer, rendre le système imprédictible
- Randomize SLAB Allocator : gestion de la mémoire de façon plus efficace et aussi de le rendre moins prédictible
- Randomize address of kernel Image : adressage de la mémoire moins prédictible
- Write protect kernel text and module : empêche le code d'être modifié. Le code du kernel sera en mode read-only
- Enable random number generator : Permet au système de générer des nombres aléatoires ce qui est utile lors des processus d'encryptage (génération des clés)
- Filter access : restreit l'accès à la mémoire qu'au user root
- Restrict unprivileged access to the kernel syslog : restreit l'accès des logs du système qu'au user root
- Kernel Memory Initialization : initialise toutes les valeur de la mémoire à zéro évitant ainsi les problème de heap memory non initialisée
- File system : active les labels de securité pour les différentes évolution de format de partition

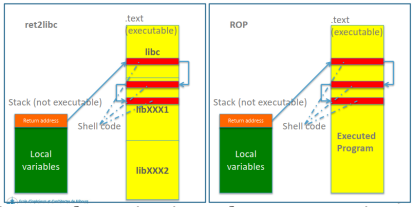
22. D'expliquer le principe des software attacks : buffer overflow, ret2libc, ROP

A buffer overflow attack can insert and executes a shell code in an executable stack.

```
void main () {
char buffer [4];
    strcpy (&buffer[0], "123456");
}
```



Now, the stack memory is no longer executable
- ret2libc could be used to bypass non executable stack memory. The main idea is to execute code in an executable memory like libc() or other libraries.
- ROP, or Return-Oriented Programming allows also to bypass non executable stack memory. The main idea is to execute code in the program itself



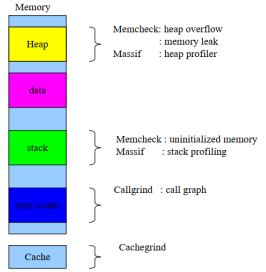
23. D'expliquer le principe des protections contre les softwares attacks : ASLR, PIE, canary

ASLR (Address Space Layout Randomization) randomize_va_space randomizes the stack and heap addresses
The PIE (Position Independent Executable) : Functions and variables addresses of an executable are not static and are computed during the program execution. This option avoids the ROP problem.

Valgrind

24. De connaître les différents outils de Valgrind et leur utilisation

1. Memcheck is a memory error detector. It helps you make your programs, particularly those written in C and C++, more correct.
2. Cachegrind is a cache and branch-prediction profiler. It helps you make your programs run faster. Comment j'utilise la cache -> optimizer lecture d'un gros fichier.
3. Callgrind is a call-graph generating cache profiler. It has some overlap with Cachegrind, but also gathers some information that Cachegrind does not. Comment j'utilise le cpu.
4. Helgrind is a thread error detector. It helps you make your multi-threaded programs more correct. Un soft est bloqué par exemple, on peut utiliser helgrind pour essayer de trouver ou ça bloque ou si ça risque de bloquer quelque part.
5. Massif is a heap and stack profiler. It helps you make your programs use less memory. Permet de vérifier d'utilisation de la heap -> optimiser la heap, éviter l'overflow



25. Pour un code donné avec des erreurs, savoir quel-s outil-s de Valgrind utiliser

voir point 24

Hardening Linux

26. De contrôler l'intégrité d'un package, d'un programme

??????????????

27. De configurer un nouveau package, programme

```
tar xvzf package1.tar.gz
cd package1
less INSTALL or less README     // Analyze the different options
./configure --help              // Analyze the different options
```

28. De cross-compiler un programme

??????????????

29. De contrôler les services, les ports ouverts

netstat command shows the open tcp and udp ports
$netstat - atun$
lsof command shows open files
$lsof|grepIP$
nmapssss command scans open ports for another hosts
$nmap - Pn - p1 - 65535192.168.0.11$ (scan all tcp ports fot the host 192.168.0.11)

30. De contrôler les « file systems »

By separating file systems into various partitions, it is possible to fine tune permissions and functionalities

31. De contrôler les permissions des fichiers, répertoires

By separating file systems into various partitions, it is possible to fine tune permissions and functionalities.

Ajournaling file system must be installed (e.g. ext3, ext4) and activated

Areas where users have “write privileges” should be kept on their own partition.

LVM (Logical Volume Manager) can be used when more than four partitions are required.

- read access (r)
- write access (w)
- execute access (x)

32. De sécuriser le réseau

- Disable IPv6 :

sysctl –w /net/ipv6/conf/default/disable_ipv6=1

- IP source routing must be disabled :

sysctl –w net/ipv4/conf/all/accept_source_route=0

- Forwarding (Routing) of normal and multicast packets should also be deactivated unless expressively needed :

sysctl –w net/ipv4/conf/all/forwarding=0
sysctl –w net/ipv6/conf/all/forwarding=0
sysctl –w net/ipv4/conf/all/mc_forwarding=0
sysctl –w net/ipv6/conf/all/mc_forwarding=0

- Block ICMP redirect messages :

sysctl –w net/ipv4/conf/all/accept_redirects=0
sysctl –w net/ipv6/conf/all/accept_redirects=0
sysctl –w net/ipv4/conf/all/secure_redirects=0
sysctl –w net/ipv4/conf/all/send_redirects =0

- Enable source route verification in order to prevent spoofing (attaque, usurpation d’identité) :

sysctl –w net/ipv4/conf/default/rp_filter=0

- Log all malformed packed and ignore icmp bogus ones :

sysctl –w net/ipv4/conf/all/log_martians=1
sysctl –w net/ipv4/icmp_ignore_bogus_error_responses=1

- Disable ICMP echo and timestamp responses sent via broadcast or multicast :

sysctl –w net/ipv4/icmp_echo_ignore_broadcasts=1

- Increase resilience under heavy TCP load by increasing backlog buffer and by enabling syn cookies :

sysctl –w net/ipv4/tcp_max_syn_backlog=4096
sysctl –w net/ipv4/tcp_syncookies=1

- *sysctl –p* command reads the file /etc/sysctl.conf and configure kernel parameters

```
cat /etc/sysctl.conf
  net/ipv6/conf/default/disable_ipv6=1
  net/ipv4/conf/all/accept_source_route=0
  net/ipv4/conf/all/forwarding=0
  net/ipv6/conf/all/forwarding=0
  net/ipv4/conf/all/mc_forwarding=0
  net/ipv6/conf/all/mc_forwarding=0
  net/ipv4/conf/all/accept_redirects=0
  net/ipv6/conf/all/accept_redirects=0
  net/ipv4/conf/all/secure_redirects=0
  net/ipv4/conf/all/send_redirects=0
  net/ipv4/conf/default/rp_filter=0
  net/ipv4/conf/all/log_martians=1
  net/ipv4/icmp_ignore_bogus_error_responses=1
  net/ipv4/icmp_echo_ignore_broadcasts=1
  net/ipv4/tcp_max_syn_backlog=4096
  net/ipv4/tcp_syncookies=1
```

33. De contrôler-sécuriser les comptes utilisateurs

The umask is the user file creation right

#umask 027 (all permissions)

- Le premier chiffre contrôle les droits d’accès de l’utilisateur propriétaire.
- Le deuxième chiffre contrôle les droits d’accès d’un groupe d’utilisateurs.
- Le troisième chiffre contrôle les droits d’accès des autres utilisateurs.

0 : rwx ; 1 : rw- ; 2 : r-x ; 3 : r– ; 4 : -wx ; 5 : -w- ; 6 : –x ; 7 : - - -(aucun)

34. De limiter le login root

- Direct root logins must be restricted to the console (emergency situations usually require hands at the console). This can be achieved by modifying the /etc/securetty file. Example for NanoPi : echo “ttyS0” > /etc/securetty
- Only root can access to the root directory : chmod 700 /root
- Use su or sudo commands in order to have the root rights
- The root PATH must not contain the current directory or writable directories

35. De sécuriser le noyau

- ASLR (Adresse Space Layout Randomization) randomize_va_space : 1 : shared libraries will be loaded to random addresses, echo 2 = echo1 plus heap randomization

```
make linux-menuconfig:
        General Setup → [ ] Disable Heap randomization
        Kernel feature → [*] Randomize the address of the kernel image
echo 1 > /proc/sys/kernel/randomize_va_space
echo 2 > /proc/sys/kernel/randomize_va_space
Or
sysctl -w kernel.randomize_va_space=1
sysctl -w kernel.randomize_va_space=2
```

- Write protect kernel text section, kernel configuration :

```
make linux-xconfig:
    Kernel Feature → [*] Apply r/o permissions of VM areas also to their
                         linear aliases
    General Setup → [*] Set loadable kernel module data as NX and text as RO
```

- Strip assembler-generated symbols during link, kernel configuration

```
make linux-menuconfig:
    Kernel Hacking→ Compile time check and compiler options
          → [ ] Compile the kernel with debug info
          → [*] Strip assembler-generated symbols during link
```

- Enable -fstack-protector buffer overflow detection , kernel configuration

```
make linux-menuconfig:
   General architecture-dependent options -> [*] Stack Protector buffer
                                             overflow detection
                                          [*] Strong Stack Protector
```

- Only root can access to the kernel system logs (through dmesg).

```
make linux-xconfig: Security options → [*] Restrict unprivileged access to
the kernel syslog
```

36. De sécuriser une application

Dans le makefile :

CFLAGS="-fPIE -fstack-protector-all -D _FORTIFY _SOURCE=2 -ftrapv"

LDFLAGS="-Wl,-z,now,-z,relro -z,noexecstack, -pie "

Ou en ligne de commande :

gcc –Wall -Wextra -z noexecstack -pie -fPIE -fstack-protector-all - Wl,-z,relro,-z,now –O –D_FORTIFY_SOURCE=2 –ftrapv -o test test.c

- noexecstack : no-execute stack. Do not allow execution of instructions stored on the stack. An operating system and/or hardware which supports the NX bit may mark certain areas of memory as non-executable (stack, heap)
- pie : voir question 23
- stack-protector-all : Stack smashing protection is a way to protect programs from stack buffer overflows by adding random values (canaries) between the function’s local variables and the saved instruction pointer
- relro, now : All dynamic symbols must be resolved during the program startup. The GOT (Global Offset Table) table is initialized and next it is marked read-only. This prevents GOT overwrite attacks.
- FORTIFY_SOURCE : this macros provide buffer overflow checks for the following functions
  memcpy, mempcpy, memmove, memset, strcpy, stpcpy, strncpy, strcat, strncat, sprintf, vsprintf, snprintf, vsnprintf, gets
  Set to 1, with compiler optimization level 1 (gcc -O1) and above, checks that shouldn’t change the behavior of conforming programs are performed
  Set to 2 some more checking is added, but some conforming programs might fail
  Some of the checks can be performed at compile time, the results are in compiler warnings ; other checks take place at run time, the results are in a run-time error if the check fails.
- ftrapv : Integer overflow. not short or unsigned int.
- strnXXX functions : Replace strcat, strcpy by strn.... functions.
  char *strncpy(char *dest, const char *src, size_t n)

This function is similar to strcpy(), except that at most n bytes of src are copied. If the length of src is less than n, strncpy() pads the remainder of dest with null bytes.
Some C functions such as gets(), strcpy(), strcat(), printf() are known to be insecure because they don't check the size of the destination buffers

37. De contrôler le démarrage de Linux

??????????????

38. D'appliquer la méthodologie OSSTMM simplifiée

Open Source Security Testing Methodology Manual (OSSTMM) provides a methodology for a thorough security test
A set of security metrics, called Risk Assessment Values (RAVs), provides a powerful tool that can provide a graphical representation of state, and shows changes in state over the time

RAVs

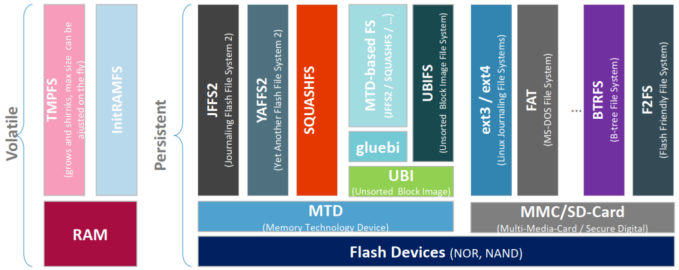
Category		Input
Operational Security		Visibility
		Access
		Trust
Loss Controls	Class A	Authentication
		Indemnification
		Resistance
		Subjugation
		Continuity
	Class B	Non-Repudiation
		Confidentiality
		Privacy
		Integrity
		Alarm
Security Limitations		Vulnerability
		Weakness
		Concern
		Exposure
		Anomaly

- Access : Count each access which potentially allow interaction with the embedded system. Count each TCP/UDP ports really open
  nmap -Pn –n -p 1-65535 IP
  nmap –Pn –n –sU -p 1-65535 IP
  netstat -atunp
- Authentication : Count each connection to a system (ssh, console, . . .) which asks a username and password
- Confidentiality : Count each connection where the data is encrypted. Example : it is better to have only a ssh connnection than a telnet connection to a system
- Vulnerability, Weekness, Concern :
  Check if a version of a program has vulnerabilities
  Check the authentication passwords. The passwords are stored in /etc/shadow, you can use hashcat, John the Ripper, hydra programs in order to check the passwords Check the robustness of the cryptographic algorithms
- Exposure : Give direct or indirect unjustified visibility of targets
  nmap –Pn –n –p 22 –sV IP // indicate the service version (Openssh 8.1p)

Filesystem

39. De connaître les différents types de systèmes de fichiers ainsi que leurs applications

Pour les systèmes embarqués, il existe deux catégories de systèmes de fichiers, les volatiles en RAM et les persitents sur des Flash (NOR et de plus en plus NAND)
Deux technologies principales sont disponible sur les Flash, soit les MTD (Memory Technology Device) ou les MMC/SD-Card (Multi-Media-Card / Secure Digital Card)



40. De connaître les caractéristiques des filesystems ext2-3-4, ainsi que les commandes associées

- EXT2 : file system for the Linux kernel. Not a journaled file system. Uses block mapping in order to reduce file fragmentation (it allocates several free blocks).
  After an unexpected power failure or system crash (also called an unclean system shutdown), each mounted ext2 file system on the machine must be checked for consistency with the e2fsck program
- EXT3 : replaces ext2. Is compatible with ext2. Is a journaled file system. The ext3 file system prevents loss of data integrity even when an unclean system shutdown occurs.
- EXT4 : Compatible with ext3 and ext2, making it possible to mount ext3 and ext2 as ext4. Supports Large file system (volume max : 2^{60} bytes, file max : 2^{40} bytes). Ext4 uses extents (zone de stockage contiguë réservée pour un fichier sur le système de fichiers d'un ordinateur) (as opposed to the traditional block mapping scheme used by ext2 and ext3), which improves performance when using large files and reduces metadata overhead for large files
  Commands :

```
# Create a partition (rootfs), start 64MB, length 256MB
  sudo parted /dev/sdb mkpart primary ext4 131072s  655359s

# Format the partition with the volume label = rootfs
  sudo mkfs.ext4 /dev/sdb1 -L rootfs

# Modify (on the fly) the ext4 configuration
  sudo tune2fs <options> /dev/sdb1

# check the ext4 configuration
  mount
  sudo tune2fs -l /dev/sdb1
  sudo dumpe2fs /dev/sdb1

# mount an ext4 file system
  mount -t ext4 /dev/sdb1  /mnt/test    // with default options
  mount -t ext4 -o defaults,noatime,discard,nodiratime,data=writeback,acl,user_xattr
  /dev/sdb1    /mnt/test
```

filesystem options can be activated with the mount command (or to the /etc/fstab file)
Journaling : A journaling file system is a file system that keeps track of changes not yet committed to the file system's. The journaling guarantees the data consistency, but it reduces the file system performances
MMC/SD-Card constraints : In order to improve the longevity of MMC/SDCard, it is necessary to reduce the unnecessary writes
Mount options :

- noatime : Do not update inode (data structure in a Unix-style file system that describes a file-system object such as a file or a directory) access times on this filesystem (e.g., for faster access on the news spool to speed up news servers)
- nodiratime : Do not update directory inode access times on this filesystem
- relatime : this option can replace the noatime and nodiratime if an application needs the access time information (like mutt)

Mount options fot the journaling (man ext4) :

- Data=journal : All data is committed into the journal prior to being written into the main filesystem (It is the safest option in terms of data integrity and reliability, though maybe not so much for performance)
- Data=ordered : This is the default mode. All data is forced directly out to the main file system before the metadata being committed to the journal
- Data=writeback : Data ordering is not preserved - data may be written into the main filesystem after its metadata has been committed to the journal

Autres commandes :

Discard: Use discard requests to inform the storage that a given range of blocks is no longer in use. A MMC/SD-Card can use this information to free up space internally, using the free blocks for wear-levelling.

acl: Support POSIX Access Control Lists

default: rw, suid, dev, exec, auto, nouser, and async

- rw: read-write
- suid: Allow set-user-identifier or set-group-identifier bits
- dev: Interpret character or block special devices on the filesystem
- exec: Permit execution of binaries
- auto: Can be mounted with the -a option (mount –a)
- nouser: Forbid an ordinary (i.e., non-root) user to mount the filesystem
- async: All I/O to the filesystem should be done asynchronously

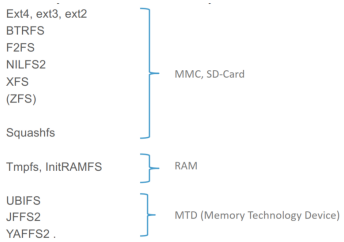
File /etc/fstab contains descriptive information about the filesystems the system can mount

41. D'expliquer les différents « files systems » utilisés dans les systèmes embarqués (ext2-3-4,BTRFS, F2FS, NILFS2, XFS, ZFS, ...)

- EXT2,3,4 : point 40

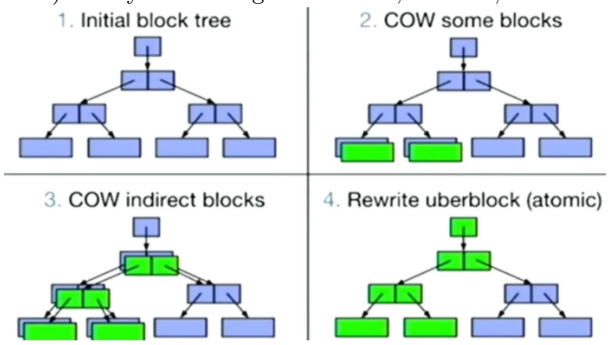
— BTRFS : point 42
— F2FS : Flash-Friendly File System. It is a log filesystem. It can be tuned using many parameters to allow best handling on different supports. F2FS features : Atomic operations, Defragmentation, TRIM support (reporting free blocks for reuse)
— NILFS2 : New Implementation of a Log-structured File System. Uses log filesystem and B-Tree technologies. Userspace garbage collector.
— XFS : Journaling filesystem. Supports huge filesystems. Designed for scalability (ability to increase or decrease). Does not seem to be handling power loss (standby state) well.
— ZFR : Zettabyte (10^{21})File System. B-Tree file system. Provides strong data integrity. Supports huge filesystems. Not intended for embedded systems (requires RAM).
— Squashfs : compressed read-only filesystem for Linux. uses gzip, lzma, lzo, lz4 and xz compression to compress files, inodes (data structure in a Unix-style file system that describes a file-system object such as a file or a directory) and directories. Intended for general read-only filesystem use, for archival use, and in embedded systems with small processors where low overhead is needed.

Performances : EXT4 is currently the best solution for embedded systems using MMC ; F2FS and NILFS2 show very good write performances
Features : BTRFS is a next generation filesystem ; NILFS2 provides simpler but similar features
Scalability : EXT4 clearly doesn't scale as well as BTRFS and F2FS

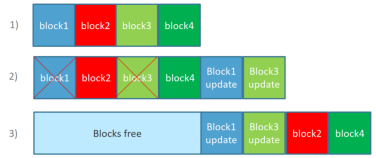


42. Expliquer les files system de type Journal, B_Tree/CoW, log filesystem

— File system : soit l'organisation hiérarchique des fichiers au sein d'un système d'exploitation. Soit l'organisation des fichiers au sein d'un volume physique ou logique, qui peut être de différents types (par exemple NTFS, FAT, FAT32, ext2fs, ext3fs, ext4fs, zfs, btrfs, etc.)
— Journalized file system : keeps track of every modification in a journal in a dedicated area. Allows to restore a corrupted filesystem. Modifications are first recorded in the journal then applied on the disk. If a corruption occurs : The File System will either keep or drop the modifications. Journalized filesystems : EXT3, EXT4, XFS, Reiser4
— B-TREE/CoW : B+ tree is a data structure that generalized binary trees. CoW (Copy on Write) is used to ensure no corruption occurs at runtime (the original storage is never modified. When a write request is made, data is written to a new storage area ; original storage is preserved until modifications are committed ; if an interruption occurs during writing the new storage area, original storage can be used). Filesystems using CoW : ZFS, BTRFS, NILFS2



— Log filesystem : Log-structured filesystems use the storage medium as circular buffer and new blocks are always written to the end. Log-structured filesystems are often used for flash media since they will naturally perform wear-levelling (extend the life of solid-state). The log-structured approach is a specific form of copy-on-write behavior. Log filesystems : F2FS, NILFS2, JFFS2, UBIFS



1) Initial state
2) Block 1-3 are updated, old blocks 1-3 are not used
3) Garbage copies block2 and 4, and frees old block1-2-3-4

— BTRFS (B-Tree filesystem) : BTRFS is a "new" file system compared to EXT. It is a B-Tree filesystem. ? ? ? ? ? ? ? ? ? ? ? ? ? ?

43. De connaître les caractéristiques du filesystem Squashfs, ainsi que les commandes associées

Create the squashed file system dir.sqsh for the regular directory /data/config/ :
bash# mksquashfs /data/config/ /data/dir.sqsh

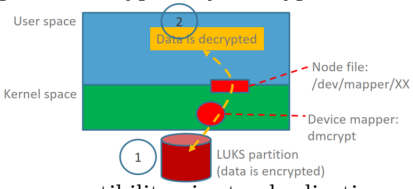
The mount command is used with a loopback device in order to read the squashed file system dir.sqsh
bash# mkdir /mnt/dir
bash# mount –o loop –t squashfs /data/dir.sqsh /mnt/dir
bash# ls /mnt/dir
It is possible to copy the dir.sqsh to an unmounted partition (e.g. /dev/sdb2) with the dd command and next to mount the partition as squashfs file system
bash# umount /dev/sdb2
bash# dd if=dir.sqsh of=/dev/sdb2
bash# mount /dev/sdb2 /mnt/dir -t squashfs
bash# ls /mnt/dir
Squashfs - NanoPi - Buildroot :
cd workspace/nanopi/buildroot
make menuconfig
- Go to: Target Packages → Filesystem and Flash utilities : choose squashfs

44. De connaître les caractéristiques du filesystem tmpfs, ainsi que les commandes associées

Tmpfs is a file system which keeps all files in virtual memory (temporarily transferring data from random access memory (RAM) to disk storage).
Everything in tmpfs is temporary in the sense that no files will be created on your hard drive. If you unmount a tmpfs instance, everything stored therein is lost
devtmpfs (device temporary) is a file system with automatically populates nodes files (/dev/. . .) known by the kernel.
mount -n -t devtmpfs devtmpfs /dev
/dev is automatically populated by the kernel with its known devices
fstab : table des différents systèmes de fichiers sur un ordinateur sous Unix/Linux

45. De connaître les caractéristiques du filesystem LUKS, ainsi que les commandes associées

Linux Unified Key Setup is the standard for Linux hard disk encryption. Data in the LUKS partition is encrypted. Data used in the user space is decrypted by dmcrypt.



Luks features : compatibility via standardization, secure against attacks, support for multiple keys, effective passphrase revocation, free
cryptsetup is a utility used to configure dmcrypt
cryptsetup uses the /dev/random and /dev/urandom node file
dmcrypt (Device-mapper) crypts target and provides transparent encryption of block devices using the kernel crypto API

(kernel configuration, Cryptographic API)
To enable dmcrypt :

```
cd /buildroot
make linux-xconfig or make linux-menuconfig
    Go to: device driver → Multiple Devices drivers support
    (RAID and LVM) → Device mapper support → Crypt target
    support
Important: choose <*> not <M>
```

```
<*>   Device mapper support
[ ]     Device mapper debugging support
< >     Unstriped target
<*>     Crypt target support
```

To use cryptsetup, it is required to add a new package in buildroot :

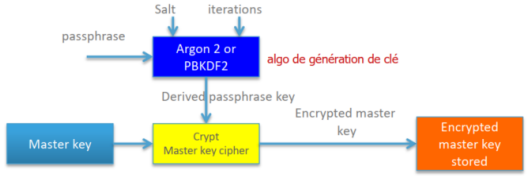
```
cd /buildroot
make menuconfig
    Go to: Target Packages → Hardware handling : choose
    cryptsetup
```

```
[ ] brltty
[ ] cc-tool
[ ] cdrkit
[ ] cpuburn-arm
[*] cryptsetup
[ ] cwiid
[ ] dhadi-linux
```

LUKS with NanoPi : On the SD Card, create a third partition (with fdisk or parted). A passphrase generates

plein de commands.....

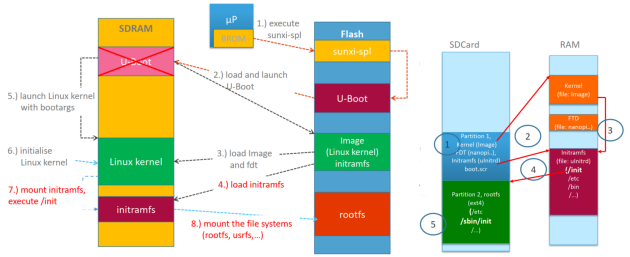
46. Savoir expliquer la gestion des clés de LUKS



TKS1 uses Argon2 or PBKDF2 (Password-Based Key Derivation Function 2) method in order to provide a better resistance against brute force attacks based on entropy weak user passphrase

47. De connaître les caractéristiques du filesystem InitramFS, ainsi que les commandes associées

initramfs is a root filesystem that is loaded at an early stage of the boot process
It provides early userspace commands which lets the system do things that the kernel cannot easily do by itself during the boot process.
Boot sequence
Les points : 1) execute sunxi-spl,2) load and launch U-Boot et 3) load Image and fdt sont les mêmes avec ou sans initramf (voir point 10)
En 4) u-boot copie le initramfs en RAM
En 5) u-boot démarre le kernel Linux
En 6) le kernel Linux s'initialise
En 7) le kernel Linux mount le initramfs et exécute le script /init, qui peut contenir différentes commandes, par exemple : décrypter le rootfs qui est dans une partition LUKS
En 8) le script /init active le rootfs



1) Kernel (Image), initramfs (uInitrd), flattened device tree (Sun50i...) and boot.scr files are located in the partition 1 of the SDCard
2) Kernel, initramfs, Sun50i.. are copied to the RAM
3) Kernel mounts initramfs (uInitrd file)
4) Kernel executes init script stored in initramfs. This init script can execute early different commands
5) Init script executes the command switch_root, which switches to the standard rootfs located in the partition 2 and executes the /sbin/init command

Show boot.cmd file : `cat /buildroot/board/friendlyarm/nanopi-neo-plus2/boot.cmd`

```
setenv bootargs console=ttyS0,115200n8 earlyprintk root=/dev/mmcblk0p2 rootwait
ext4load mmc 0 $kernel_addr_r Image
ext4load mmc 0 $fdt_addr_r nanopi-neo-plus2.dtb
ext4load mmc 0 0x50000000 uInitrd        // Load initramfs: uInitrd is the initramfs
booti $kernel_addr_r 0x50000000 $fdt_addr_r
```

initramfs address

48. De savoir créer un initramFS

Principle to build an initramfs :
1. to copy the right files into a directory (/buildroot/ramfs),
2. to copy these files in a cpio archive file,
3. to compress this file
4. To add the uboot header
Initramfs : kernel configuration :
make linux-menuconfig
CONFIG_BLK_DEV_INITRD=y
General setup —> [*] Initial RAM filesystem and RAM disk (initramfs/initrd) support
Automount a devtmpfs and initiate the /dev :
Device Drivers → Generic Drivers options → Maintain a devtmpfs filesystem to mount at /dev → Automount a devtmpfs

Filesystem security

49. De connaître les « files permissions » sous Linux

voir point 33. Write (create-delete-rename) file/folder.

50. De contrôler et sécuriser les comptes utilisateurs sous Linux

??????????????

51. De connaître les real-effective userID and groupID

User : Each user is assigned a unique number, called a user ID, or UID
Group : Each group is assigned a unique number, called a group ID, or GID. Every group contains one or more user IDs. A single user ID can be a member of some of groups, but groups can't contain other groups ; they can contain only users. Like users, groups have names.
Sticky bit : A directory that has the sticky bit set allows you to delete a file only if you are the file owner

52. De connaître les ACL

Access Control List. Filesystems that allow the ACL : Ext2, ext3, ext4, ReiserFS, JFS, XFS, Btrfs, Tmpfs, JFFS2, CIFS.

53. De connaître les attributs particuliers des filesystems ext2-3-4

-i :file/directory can not be modified, deleted, renamed or linked symbolically, not even by root. Only root or a binary with the necessary rights can set this attribute.
-A :Date of last access is not updated (only useful for reducing disk access)
-S :File is synchronous, the records in the file are made immediately on the disc. (equivalent to the sync option of mount)
-a :File can only be open in append mode for writing (log files, etc.) Only redirection » can be used, the file can not be deleted.
-c :File is automatically compressed before writing to disk, and unpacked before playback
-d :File will not be saved by the dump
-j :Ext3-ext4 :A file with the 'j' attribute has all of its data written to the ext3 or ext4 journal before being written to the file itself
-s : When the file is destroyed, all data blocks are being released to zero.

54. De rechercher des permissions de fichier faibles

#find . -perm 200 // file permissions = 200
#find . -perm -220 // write bit for user and group = 1
#find . -perm /220 // write bit for user or group = 1
#find . -perm +220 // write bit for user or group = 1 (like /220)
"other" write bit = 1
find / -type d -perm -2 -ls (find the directories with the others can write)
chmod -R o-w /dir1/dir2 (Be careful : remove other bit = write for all files under /dir1/dir2)
SUID bit = 1
find / -type f -perm -4000 -ls
chmod u-s /usr/bin/file
chmod -R u-s /var/directory/
GUID bit = 1
find / -type f -perm -2000 -ls
chmod g-s /usr/bin/file
chmod -R g-s /var/directory/
Sticky bit = 1

find / -type f -perm -1000 -ls
chmod -t /usr/bin/file
chmod -R -t /var/directory/

55. Comment sécuriser les répertoires temporaires

voir point 44

/tmp, /var/tmp are directories or partitions which hold temporary files

These directories can store temporary bots, malware, rootkits, ...

A more secure solution would be to set /tmp in its own partition, so that it can be mounted independently of the / partition and have more restrictive options set. Exemple :

/dev/sda7 /tmp ext4 nosuid,noexec,nodev,rw 0 0

This would set the nosuid, noexec, and nodev options, meaning : no suid programs are permitted, nothing can be executed, and no device files (node file) may exist (see man mount)

In order to improve the security for the tmpfs, the /etc/fstab file must by modified

tmpfs /dev/shm tmpfs mode=0777 0 0

tmpfs /tmp tmpfs \defaults,nosuid,noexec,nodev,rw 0 0

If you don't have the ability to create a fresh /tmp partition on existing drives, you can use the loopback capabilities of the Linux kernel by creating a loopback filesystem that will be mounted as /tmp and can use the same restrictive mount options

56. De savoir comment les mots de passe sont mémorisés sous Linux

On a host, the encryption method (DES, MD5, SHA512) used is in the file /etc/login.defs

On NanoPi with busybox, the encryption method used is given by the command :

passwd –help

passwd -a des, or passwd -a md5 or passwd -a sha256 or passwd -a sha512 username

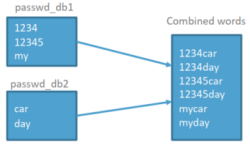
57. De connaitre les différentes possibilités pour casser un mot de passe

- Dictionary attacks : uses a file with words called dictionnary.
- Combinator attack : uses two dictionaries and tries all combinations with these 2 dictionaries : dict1= 1234, abcd, dict2=toto, ruru → 1234toto, 1234rutu, abcdtoto,abcdruru
- Hybrid attacks : uses a dictionary and add some letters/numbers at the end of each word : asdf12, asdf123, asdf1234, . . .
- Brute force attacks : Attempt to try all possible combinations, which means it will take the longest amount of time out of the three types.
- Mask attacks : Similar to Brute Force, this requires additional user inputs. If you know the length of the password, a few of the characters used, or even a prefix or suffix, it takes less time to recover.

58. De savoir utiliser hashcat pour casser un mot de passe

Hashcat attack modes :

- Straight mode : dictionary attack with one password database
  hashcat -a 0 -m 0 hashfile dictionaryFile (MD5)
  Hashcat reads each line of dictionaryFile, computes the MD5 hash and compares with hashes stored in hashfile
- Combinator mode : hybrid attack with two password databases
  hashcat -a 1 -m 0 hashfile dictionaryFile1 dictionaryFile2 (MD5)
  combines dictionaryFile1 words with dictionaryFile1 words



- Brute/mask mode : advanced brute force-mask attack
  hashcat -a 3 -m 0 hashfile ?d ?d ?d
  Mask attack : Try all combinations from a given keyspace just like in Brute-force attack, but more specific.
- Hybrid mode : hybrid attack (password database + mask)
  hashcat -a 6 -m 0 hashfile ?u ?u dictionaryFile ?d ?d ?d
  One side is simply a dictionary, the other is the result of a Brute-Force/mask attack

Hash mode : hash structure (0 : MD5, 1800 : sha512 unix)

Firewall iptables

59. De connaître les principes de Netfilter, iptables

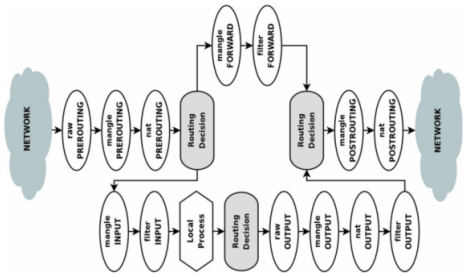
iptables is a user-space application program that allows to configure the tables provided by the Linux kernel firewall

60. Savoir expliquer les notions de chain-tables

When a packet arrives, it traverses different chains. Theses chains are contained in different tables. The combination chain-table are the hooks : Chains (INPUT, OUTPUT, FORWARD, PREROUTING, POSTROUTING) and tables (raw, mangle, nat, filter)

Chain explanation : INPUT(the packet is for the local host), OUTPUT (the packet is sent from the local host), FORWARD (the packet arrives to an interface (eth0) and it is forwarded to another interface (eth1)), PREROUTING/POSTROUTING (used by NAT) Tables :

- Filter : mainly used for the firewalls. Contains the built-in chains : INPUT (for packets destined to local sockets), FORWARD (for packets being routed through the box), OUTPUT (for locally-generated packets).
- mangle : This table is used for specialized packet alteration-modification. It consists of five built-ins chains : INPUT (for packets coming into the box itself), FORWARD (for altering packets being routed through the box), POSTROUTING (for altering packets as they are about to go out), PREROUTING (for altering incoming packets before routing), OUTPUT (for altering locally-generated packets before routing)
- nat : Network Address Translation. This table is consulted when a packet that creates a new connection is encountered. It consists of three built-ins chains : PREROUTING (for altering packets as soon as they come in), OUTPUT (for altering locally-generated packets before routing), POSTROUTING (for altering packets as they are about to go out)



61. Savoir expliquer les différences entre les firewall Stateless et Stateful packet filtering

- Stateless : Stateless packet filtering (table filter et ACCEPT, DROP, REJECT). Permet de protéger au niveau réseau (bloquage d'une ip en particulier, ou ports). Les paquets sont analysés de maniére individuelle

- Statefull : Permet de protéger au niveau du paquet en fonction du contexte (précédents paquets). Il est possible d'accepter des paquets venant de l'extérieur seulement si ils sont des réponses à des requêtes venant de l'intérieur

• Utilisation de connection tables pour traiter les différentes parties des protocoles.

• NEW : Nouveau paquet qui n'est pas lié à une connexion active

• ESTABLISHED : Une connexion passe de NEW à ESTABLISHED losrque la connexion est validée par la direction opposée

• RELATED : Paquets qui ne font pas partie d'une connexion existante mais qui sont liés à une autre. (Par exemple réponses ICMP pour une communication FTP).

62. Savoir configurer avec iptables un firewall simple de types Stateless (pages 17-19) et Stateful (pages 26-32)

NanoPi allows only incoming SSH connections

```
iptables -A INPUT -i lo -j ACCEPT        // Local traffics are allowed between applications
iptables -P INPUT DROP
iptables -P OUTPUT DROP  kick la connexion
iptables -P FORWARD DROP                 -m permet d'utiliser une extension
iptables -A INPUT -i eth0 -p tcp --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m conntrack --ctstate ESTABLISHED -j ACCEPT
```

NanoPi allows only incoming SSH connections and all outgoing connections

```
iptables -A INPUT -i lo -j ACCEPT        // Local traffics are allowed between applications
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

iptables -A INPUT -i eth0 -p tcp --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -i eth0 -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m conntrack --ctstate ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT
```

63. Connaître le principe des NFQUEUE

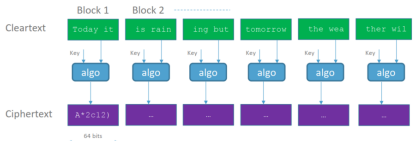
The NFQUEUE target means to pass the packet to userspace.

TPM (trusted platform module)

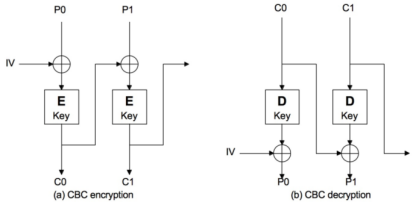
64. Savoir expliquer uniquement le principe des chiffrements symétrique, asymétrique,fonctions de hachage, la signature digitale

- Chiffrement symétrique : Une seul clé pour crypter et décrypter.

Block mode : Cleartext is divided in block. Each block has the same length (64, 128 bits). Each block is encrypted by an algorithm (AES, IDEA, 3DES, ...)



CBC mode : Cipher Block Chaining. Every block is coded with the result of the previous block.



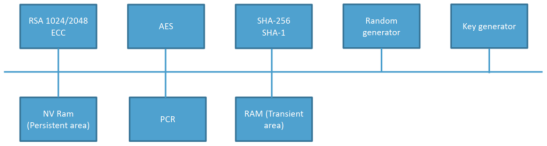
- Chiffrement asymétrique : A key for the encryption (public or private key). Another key for the decryption (private or public key). The public keys must be exchanged (stored in a certificate).

Confidentiality :Encrypt with the public key and decrypt with the private key (confidentiality, integrity). Digital signature : Encrypt with the private key and decrypt with the public key (Authenticity, integrity)

65. Connaitre les différentes implémentations des TPM (discrete, integrated, Hypervisor,Software)

1. Discrete TPMs are dedicated chips that implement TPM functionalities in their own tamper resistant semiconductor package
2. Integrated TPMs are part of another chip. Intel : Platform Trust Technology (PTT), AMD : fTPM, ARM : TrustZone
3. Hypervisor TPMs (vTPMs) are virtual TPMs provided by and rely on hypervisors.
4. Software TPMs are software emulators of TPMs that run with no more protection than a regular program gets within an operating system

66. Connaitre l'architecture interne d'un TPM



RSA 1024/2048, ECC : Asymmetric algorithms, encrypt-decrypt, sign

AES : Symmetric algorithm, encrypt-decrypt, sign

SHA-256, SHA-1 : hash function

Random generator : create random value

Key generator : Create key for asymmetric algo

NV Ram (Persistent area) : Store different objects (keys, data, ...) in NV Ram

PCR (Platform Configuration Registers) stores hash values of different parts : code, files, partitions, ...

RAM (Transient area) : Store keys, data, this area is limited (free this area with : tpm2_flushcontext –t)

67. Connaitre les différentes hiérarchies des TPM (endorsement, platform, owner, null)

- Endorsement hierarchy : The endorsement hierarchy is reserved for objects created and certified by the TPM manufacturer. The endorsement seed (eseed) is randomly generated at manufacturing time and never changes during the lifetime of the device. The primary endorsement key is certified by the TPM manufacturer, and because its seed never changes, it can be used to identify the device
- Platform hierarchy : The platform hierarchy is reserved for objects created and certified by the OEM (Original Equipment Manufacturer) that builds the host platform. The platform seed (pseed) is randomly generated at manufacturing time, but can be changed by the OEM by calling tpm2_changepps
- Owner hierarchy, also known as storage hierarchy : The owner hierarchy is reserved for us - the primary users of the TPM. When a user takes a TPM, the owner hierarchy can be erased by the tpm2_clear command. tpm2_clear generates a new owner seed (oseed)
- Null hierarchy : The null hierarchy is reserved for ephemeral keys. The null seed is re-generated each time the host reboots

68. Savoir créer, utiliser des clés avec un TPM

```
Create RSA endorsement key: tpm2_createprimary -C e -G rsa2048 -c e_primary.ctx
Create RSA platform key:    tpm2_createprimary -C p -G rsa2048 -c p_primary.ctx
Create RSA owner key:       tpm2_createprimary -C o -G rsa2048 -c o_primary.ctx
Create RSA null key:        tpm2_createprimary -C n -G rsa2048 -c n_primary.ctx
```

After tpm2_createprimary command, public-private keys are stored in RAM in the transient area

- tpm2_getcap handles-transient command shows the index of the public-private keys
- tpm2_flushcontext command deletes keys from RAM (tpm2_flushcontext 0x80000000 ou tpm2_flushcontext -t)
- tpm2_readpublic command get the public key from the TPM (tpm2_readpublic -c 0x80000000(ou o_primary.ctx) –format PEM -o o_primary.public)

openssl command shows the public key (openssl rsa -pubin -in o_primary.public -text)

69. Connaitre les commandes principales d'un TPM (pas tous les paramètres, mais savoir expliquer ce que font ces commandes, être capable de dessiner ce que font les commandes)

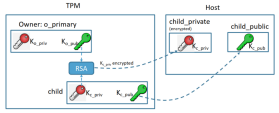
voir point 68

- child_private file contains the private asymmetric key or symmetric keys. These keys are encrypted by the parent keys. - child_public file contains public asymmetric keys and different references

Commands :

tpm2_create -C o_primary -G rsa2048 -u child_public –r child_private

- Child asymmetric keys :



First load the key :

tpm2_load -C primary.ctx -u child_pub -r child_pr -c child

Then ncrypt with child key :

tpm2_rsaencrypt -c child -s rsaes clearfile –o encryptedfile

Or decrypt with Child keys :

tpm2_rsadecrypt -c child -s rsaes encryptedfile -o clearfile

Or Verify signature with Child keys :

tpm2_verifysignature -c child -g sha256 -s file.sign -m file

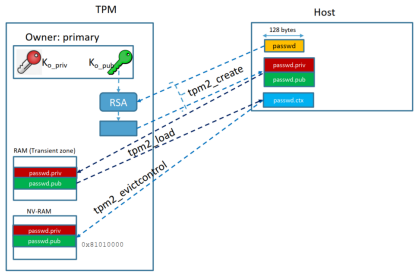
70. Savoir encrypter-décrypter, signer-vérifier avec un TPM

voir point 69

71. Savoir utiliser les registres PCR

Platform Configuration Registers. The prime use case is to provide a method to cryptographically record (measure) software state or configuration data used by a device.

72. Savoir sauver des données sur le TPM



73. Savoir sauver des données et les protéger avec une PCR policy