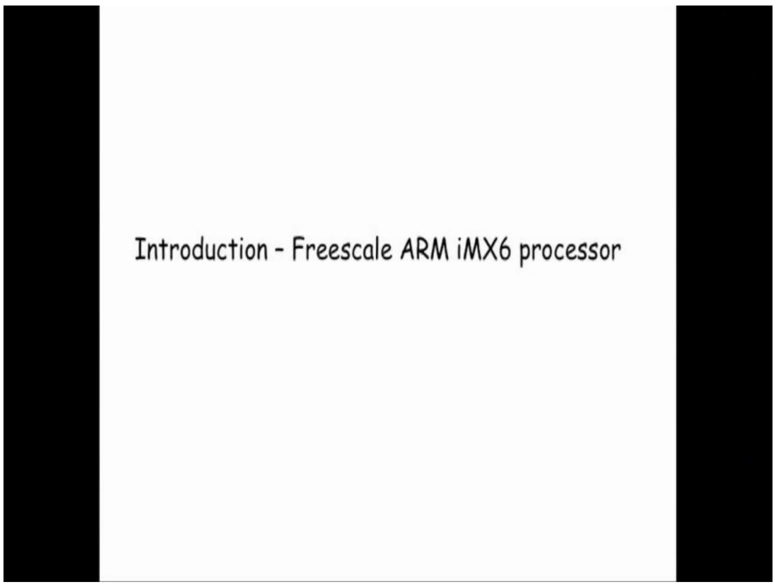
Information Security - II
Prof. V. Kamakoti
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture - 37
Introduction - Freescale ARM IMX6 Processor

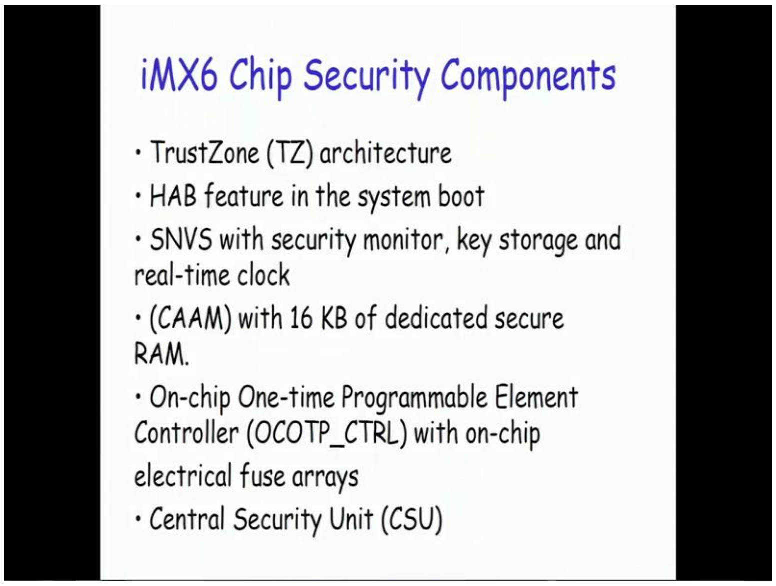
(Refer Slide Time: 00:09)



Introduction - Freescale ARM iMX6 processor

So, what we were discussing till now was about generic security principles and technologies protocols it is actually available.

(Refer Slide Time: 00:24)



iMX6 Chip Security Components

- TrustZone (TZ) architecture
- HAB feature in the system boot
- SNVS with security monitor, key storage and real-time clock
- (CAAM) with 16 KB of dedicated secure RAM.
- On-chip One-time Programmable Element Controller (OCOTP_CTRL) with on-chip electrical fuse arrays
- Central Security Unit (CSU)

So, we will look at with respect to these, what are the components that are actually available inside the ARM processor and what kind of features are actually made use of among the things that we actually discussed in the morning before lunch.

So, the different security components please understand all these are actually part of the processor inside. So, there is nothing that, we are talking external to the processor and that is basically, the beauty of whole thing. So, we have something called as a Trust Zone architecture. The Trust Zone architecture is basically helps to split up running of any application in, what is called as a secure world and what is called as the normal world. So, when I say that I am running an application and the context is right now, inside the secure world that application is going to have only very, very restricted access in terms of what memory areas that it can access. Whereas, when it is running in the normal world anything that has been marked for the secure world access alone, will not be accessible to it.

So, what is called as have feature, high assurance boot feature. Which is basically part of my system boot. So, we are going to take a look at it in detail of what this HAB is all about it helps me to ensure that the image that it is getting booted now, is not something which has been compromised on right. Then there is something called as an SNVS. SNVS stands for secure non-volatile storage which has an internal security monitor key storage and a real-time clock. So, this is something which is actually going to be a non-volatile storage space that is actually available to the any kind of security application that is required to be running here because of the fact there it is non-volatile. It is going be actually, available across power cycles, number one.

It has typically a coin cell operated battery with which, it is able to provide the non-volatility. And then it is a secure because of the fact that any kind of an attack that actually happens on my device the processor is programmable, to go ahead and delete the contents what is actually available as part of this area. Which is essentially what security is all about right, suppose I find that my device that I have built is physically tampered. Somebody had actually tried to open up the box, I actually have stored lot of confidential information like keys and all that as part of the SNVS, I do not want this area to be available for the person who has actually successfully opened it up. So, there are configurable registers which I can program inside my processor to go ahead and cleanup this entire SNVS area on any of these events that has actually recognized.

Then there is something called as a cam. So, cryptographic acceleration and assurance module with 16 KB of dedicated secure ram. Now the first a in cam actually stands for acceleration. So, with whatever we have talked in the morning and when we use the word acceleration right now what comes to your mind.

So, what is that I am going to accelerate?

Student: (Refer Time: 03:58)

For doing what

Student: (Refer Time: 04:04)

Yes, so not only encryption and decryption, but my entire gamut of security functionality. So, I want to generate random number, I want to generate my keys, I want to encrypt, I want to decrypt all the functionality this particular cam module basically helps me to accelerate. So, what is do by helping me to accelerate? He has got the complete functionality implemented inside the processor, because of which any application that I am running at my device right need not really run all these computationally intensive stuff as part of my software. It may need very intuitive for us to understand the moment I run it inside my processor it is going to be definitely much faster as compared to running it outside right. So, that is basically what this guy helps me to do.

And then in on chip one time programmable element controller with an on chip electrical fuse arrays. So, have you heard of term fuse arrays before. So, what do we mean by a fuse array. Any idea? Or were have we come across the term fuse array? So, when you say something is fused, literal English meaning what do you mean by that?

Student: (Refer Time: 05:35)

Joined, right? So, the very fact of security I would need certain memory locations in which I actually have written some values and those locations I do not wanted to be first read by outside. Secondly, I do not want them to be really being modified once I have written them, right?

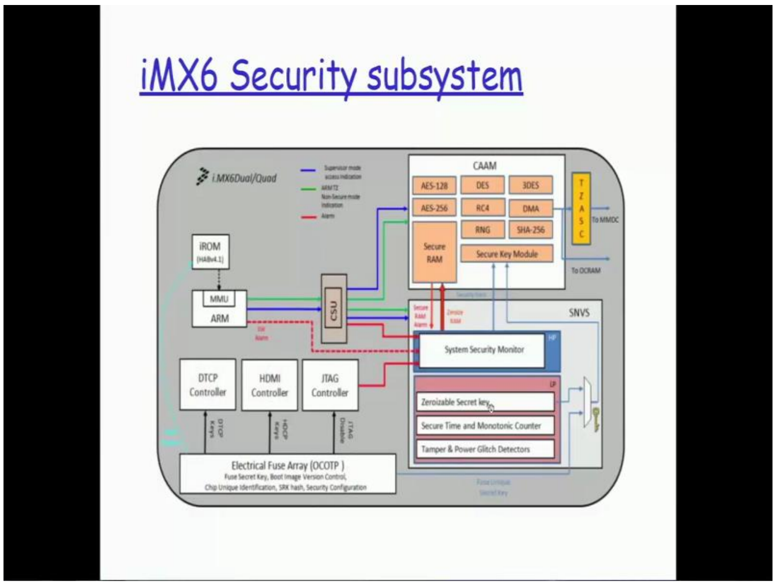
So, that is basically what we refer to as OTP, one time programmable. So, one time programmable essentially means that once I write on to that and then safe was I am fusing it right now; that means, subsequent modifications to that is completely denied right. So, inside the processor some of the functionalities can read the value, but there can never be any more modifications in those particular registers, right. So, that is basically what this part of my processor is helping me to do. And then there is something called as CSU, the central security unit, that is the core unit that is very helpful whenever I basically want certain peripherals right to be allowed accessed only in a secure mode. So, we talked about it trust zone architecture, where I was telling you an application can either run in a secure world or in a normal world and when I am running in a secure world, I would need as running as the application access to some very specific secure area secure memory areas.

So, who is going to control access to those specific areas, that guy is basically the CSU. So, all these components that we are talking of here are different blocks inside my processor. So, that is basically the key that we will have to understand. So, these are not somethings which are external to the processor, but internal to the processor and every block every functionality that we have talked about because these components are inside my processor are all available as part of my processor functionality itself right. So, very broadly speaking that this is how the whole thing is looking like. So, this is basically the accelerator and the assurance module that I talked about. So, all kinds of algorithms run here. So, des, 3 des, AES, 128 AES, 156 RNG. What is RNG? can you guess. So, I said something is required for generation of the keys.

Student: Random number

Random number generator, right. So, that is what I told. So, when you talk about acceleration. This acceleration is not only just plain encryption and decryption right. So, I have a RNG portion which is actually used very rarely. Only, when I need to record. I mean generate a new keys right the RNG is going to be actually made use of now even this RNG functionalists actually available to me as part of the cam module inside my processor right and then the SNVS the non volatile storage that I mentioned about right.

(Refer Slide Time: 08:53).

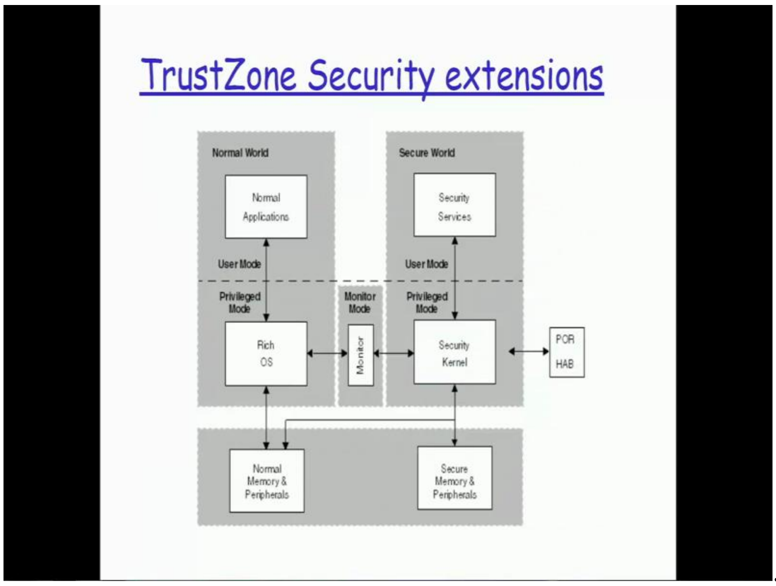


iMX6 Security subsystem

So, what is this zeroizable secret key. So, this basically what I told that the moment the processor detects that there is event that is an audit able event, what do you mean by an audit able event? Some scary event that has actually happened which I do not want it to happen right. So, if that event has happened what is the action to be taken all was configurable suppose if I had basically configured it saying that on an event that is a serious event that is actually happened, I want this portion to be completely cleared, then the portion which is pertaining to the zeroizable secret key will completely be wiped off.

So, other things like the independent controllers and my MMU then the CSU that is actually going to be providing restricted access to peripherals depending on whether this application is running, right now in a secure world or in the normal world right. So, essentially these are the different components that is there inside my processor as far as a security related aspect is concerned. And which part of it is being used for what purpose will be dependent on what is the state of my application currently and what is it actually trying to do right.

(Refer Slide Time: 10:19)



TrustZone Security extensions

So, yes. So, the TrustZone security extension. So, as I was telling you I am going to have a normal world and a secure world when my application is actually running in a secure world at that point in time the application will have access to all the memory areas and the peripherals that are basically marked for being accessible under the secure memory. So, how is it going to be determined whether it is accessible or not? And that is where the CSU comes into play right. So, the CSU is going to be basically either allowing or denying access to an application to certain memory areas and certain peripherals depending on whether this application is right now running in a secure world or in the normal world right.

So, when it is running in the secure world it is going to have access to secure memory area and all the secured peripherals. When it is actually running in the normal world it will have access only to the normal memory area and the peripherals. So, out of my entire memory area I would have actually configured a starting address to an ending address range which I would have told is accessible only when the application is running inside the secure world.

So, on a POR. What is POR? Power on Reset. So, when you basically give the power what internally happens is, the power on reset circuit actually gets kicked off right power starts, coming in to the respective components and then depending on what my configuration is, my high assurance boot might be running right. So, we will see what is high assurance boot little later. So, after that is successfully run the authentication has been successfully done it is actually go into the go into the secure world or my security kernel is actually going to be running. And the way by which this will be getting toggled between the secure world and the normal world is by executing a monitor execution that is actually available inside my processor right.

So, this is a very unique instruction that is actually available as part of the arm core because the entire TrustZone functionality is something which is actually available inside the ARM core. And IMX 6 processor being a ARM processor as derived the entire core and thereby is able to provide me this functionality right. So, I will basically be executing the monitor instruction to swap myself in the secure world to the normal world and back again whenever, I want to get it to the secure world. So, now, what do you see as an advantage here? Why do you think the processor through the TrustZone is giving you something like a secure world and something like a normal world?

So, remember that it has the capability to access some very specific memory areas alone and very specific peripherals also. So, when you say peripheral, we all understand what we mean by that right. So, what example peripherals can we think of?

Student: USB

USB.

Student: (Refer Time: 13:38)

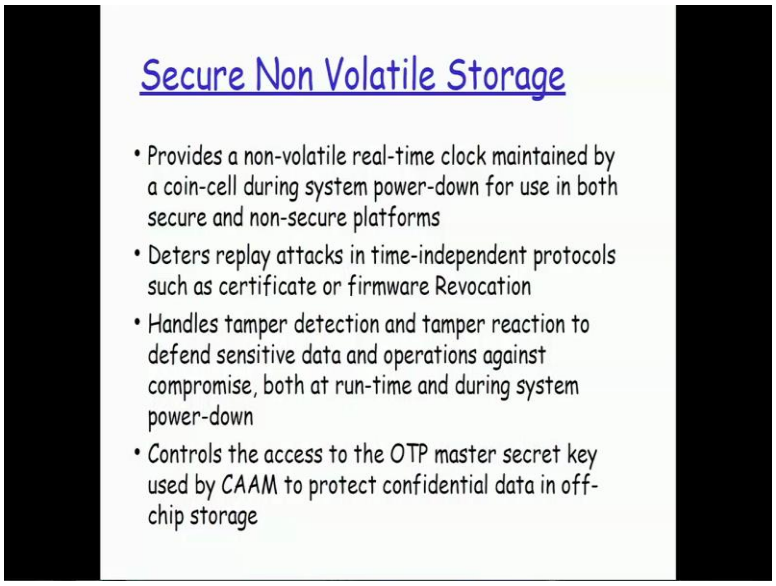
Anything else? If I have a serial port, right? My display, right? All these are peripherals. What is a need for me to protect certain peripherals? Have you ever come across a situation were certain peripherals have to be protected? Any example scenarios? If you go to organizations, they will say that they will give you a desktop PC. PC will have all the peripherals because it is a standard p c that I will possibly get even in (Refer Slide

Time: 14:22) right. But if I try to put up my pen-drive into my u s b port, it wont work, why? Because the organization might have lot of confidential data which is pertaining to their customers which they do not want to risk it out by letting the employees take it out on a pen-drive and then take it outside the facility.

In that particular example scenario, my USB port becomes a secure peripheral. But still I will want certain times when I want to access that peripheral when very dependable application is running. That application will be doing a monitor instruction to go from the normal world which is the default mode in which it will be running, to go into the secure world do whatever job it wants to done and it will be always have a watchdog associated with it. So, I cannot have an application perennial in a secure world. So, the moment the watchdog timer expires, that guy is going to be being forcibly brought out back into the normal world depending on the entire thing is configurable. So, the whole thing is configurable and once the job is done with a secure peripheral, I will actually in my application come back into the normal world and my execution will continue running from wherever I have left right.

So, that is basically the essence of my TrustZone functionality. So, the whole TrustZone please remember as I told is available as the part of the ARM core the IMX6 processor that we are talking right now is a derived from the arm core it is license to completely out of arm core because of which I have this functionality available as part of the IMX6 processor based device also.

(Refer Slide Time: 16:14)



Secure Non Volatile Storage

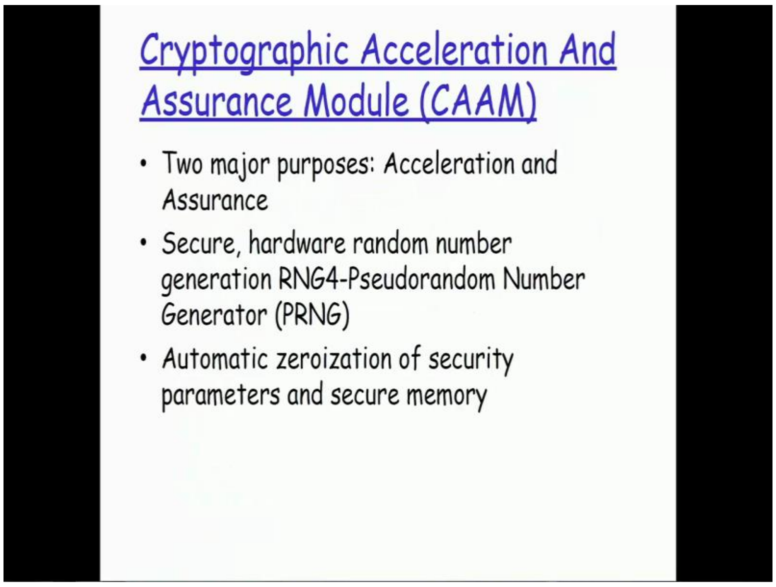
- Provides a non-volatile real-time clock maintained by a coin-cell during system power-down for use in both secure and non-secure platforms
- Deters replay attacks in time-independent protocols such as certificate or firmware Revocation
- Handles tamper detection and tamper reaction to defend sensitive data and operations against compromise, both at run-time and during system power-down
- Controls the access to the OTP master secret key used by CAAM to protect confidential data in off-chip storage

So, secure non-volatile storage as I mentioned provides a non-volatile real time clock maintained by coin cell during system power down for use in both secure and non-secure platforms. So, because of the fact that it is driven by the coin cell battery this storage area is actually available to me even when it is power down in in in in in an offline manner right. So, deters the replay attacks in time independent protocols such as certificate or firmware revocation. So, we already understand what is the replay attack correct. So, handles tamper detection and reaction to defend sensitive data and operations against compromise both at run time and during system power down.

Now what do you mean by tamper detection? I gave an example. I have a device. So, the device is right now functioning essentially the device is actually powered up, I am using the device either in a stand-alone manner or on the network remotely. Somebody is actually trying to come in and do some amount of tamper, maybe they try to forcibly open maybe they try to access some peripheral that they are not supposed to access by putting in some kind of hardware part whatever. It is any of these kind of tamper events that I have detected, or rather I have program to detect right the moment it is actually done. So, I have a very brief description how this whole things works later down there will be a reaction that will be given out by the processor what is the reaction that will be given out whatever areas that I actually have the data I have actually stored in the SNVS portion right. So, that is basically what we saw here in this slide right all these will actually be wiped off clean my pins my pin numbers my keys.

All these are actually expected to be stored as part of the SNVS area right. So, the moment the tamper happens this entire portion will be getting completely cleaned because of which the device will possibly be in a non-usable scenario after that which is essentially what I want. That is one form of security somebody has physically tried to penetrate my device even at the cause of the device being unusable, I do not want that guy to get access to the confidential key data that I have right may be my ATM pins may be the encryption decryption keys that I am using right any of that stuff I do not want to him have access to. So, the whole thing will actually be getting cleaned up and also controls the access to the OTP master secret key used by the cam module to protect confidential data and off chip storage. So, that again is actually stored in in in inside as part of my SNVS, right

(Refer Slide Time: 18:59).

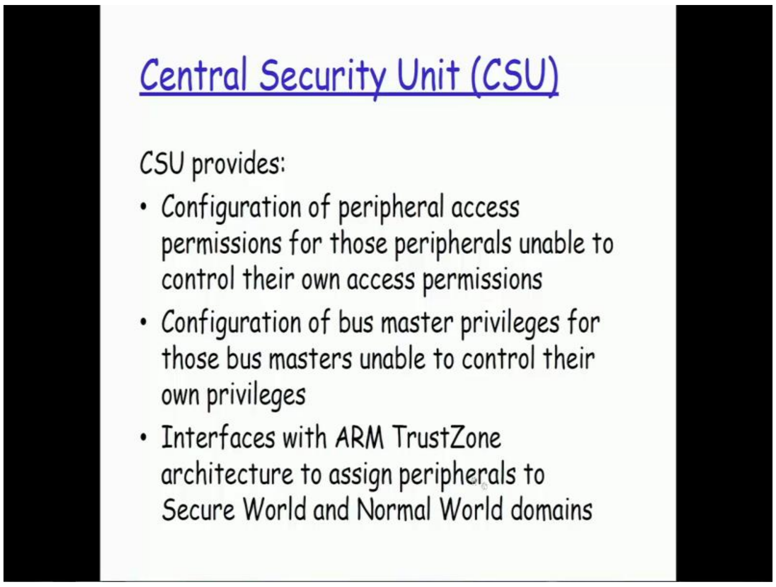


Cryptographic Acceleration And Assurance Module (CAAM)

- Two major purposes: Acceleration and Assurance
- Secure, hardware random number generation RNG4-Pseudorandom Number Generator (PRNG)
- Automatic zeroization of security parameters and secure memory

So, cryptographic acceleration assurance module as I already told two important things acceleration and assurance is actually provided by this module of the processor it has a secure hardware random number generator or pseudorandom number generator right. So, these is actually available ah internally inside this particular module whenever I have to generate the key this particular RNG or PRNGP's of my processor is what is actually going to get into play to generate me the keys because the source of that is I mentioned requires a large random number right and that is basically were my cam actually comes into play and then automatic zeroization of security parameters in secure memory whenever I have these kind of unwanted events getting triggered this guy initiates the clearance of the entire portion.

(Refer Slide Time: 19:59).



Central Security Unit (CSU)

CSU provides:

- Configuration of peripheral access permissions for those peripherals unable to control their own access permissions
- Configuration of bus master privileges for those bus masters unable to control their own privileges
- Interfaces with ARM TrustZone architecture to assign peripherals to Secure World and Normal World domains

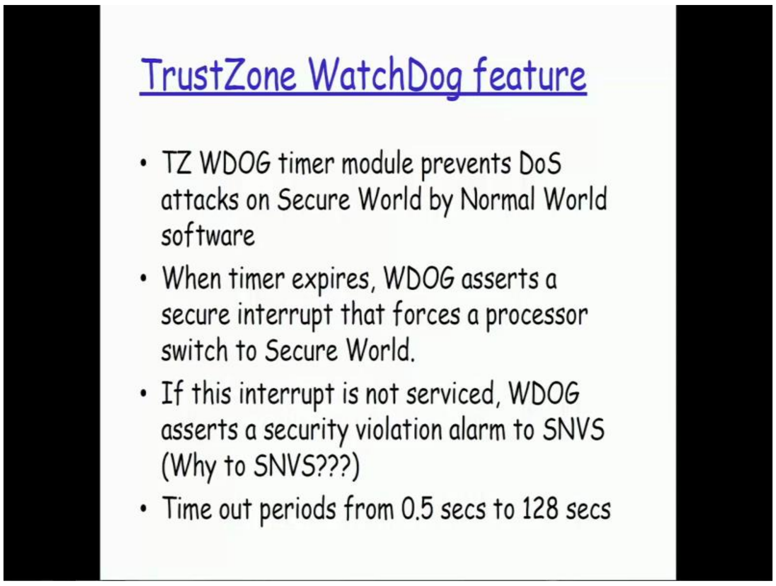
So, central security unit, I also talked about it configuration of peripheral access permissions for those peripherals unable to control their own access permissions. Configuration of bus master privileges for those bus masters unable to control their own privileges and interfaces with ARM TrustZone architecture to assign peripherals to secure world in normal world usage, right. So, as I was telling you the CSU unit is the one that basically governs which out of the peripherals that I actually have. So, I might have a VGA. I might have a touch screen I might have a USB I might have a parallel port whatever it is it gets allowed when a particular mode of operation is currently being executed by the application. So, whether this guy is in the normal mode or whether he is in the he is in the secure world depending on that what peripherals are allowed access is governed by this particular component inside. So, I basically program the CSU to say depending on the peripherals and depending on the memory address ranges.

If there is a memory address memory access to a particular address right. So, that address if it is basically part of my secure memory the application can be can successfully, access that particular location, only when it has done monitor switch into the secure world successfully. If he tries to access that particular memory address location as part of the normal world it is something like a segmentation fault and the application will be terminated, right. So, I would basically be very paranoid thinking that an application is trying to access a memory location which has been configured to the part of the secure world and I will only suspect the application. Now because, I am a secure device right. So, I will not generally give the benefit of the doubt of the application saying that was may be just try to do an access to this particular secure memory area by mistake or whatever it is straight away the application will be short dead only.

Then I will do the postmortem and like our normal legal system right. So, a person doing a murder in daylight will be arrested will be put in prison for 15 years, he will be fed daily. After 15 years, if he is taken to gallows, there will be lot of hue and cry. How can you hang a person? It is against human rights and all that, but in security world no questions asked I will shoot you first. So, the process will be killed, then subsequently I might do a postmortem whether I was right in killing him or not. If I was right, what was the actual problem that he did if I was not right? why did I make a mistake? All those postmortem exercises will happen. But first things first, right?

So, when you have to do somethings do something do not talk. So, the process will be straight away terminated and then subsequently the postmortem if at all possible will be attempted to be done right. So, that is actually taken care of by this guy right

(Refer Slide Time: 23:14).



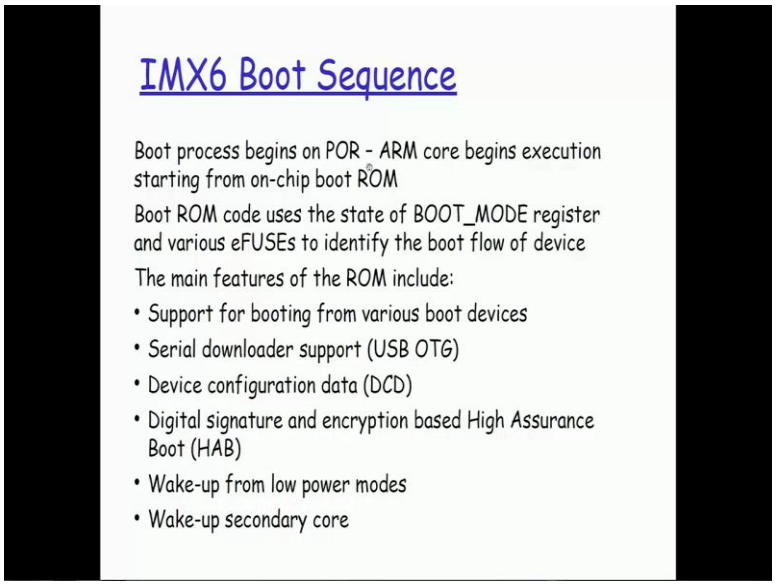
TrustZone WatchDog feature

- TZ WDOG timer module prevents DoS attacks on Secure World by Normal World software
- When timer expires, WDOG asserts a secure interrupt that forces a processor switch to Secure World.
- If this interrupt is not serviced, WDOG asserts a security violation alarm to SNVS (Why to SNVS???)
- Time out periods from 0.5 secs to 128 secs

So, TrustZone watchdog features as I was telling you there is a watchdog timer module. So, we all understand watchdog right processor it is internally available, but this trust zone watchdog is a special intensation of the watchdog, where it is as I told you some guy cannot be continuing to be in a secure world perennially. If he tries to act smart says, that boss I will execute the monitor instruction, go to the secure world and continue to be ever in the secure world having thereby access to all the peripherals and all the memory area which is the part of my secure world secure TrustZone. Watchdog will come into play. So, whatever duration I have configured he will be forced back into the normal world even if that guy has not initiated the transition back into the normal world, right. So, this interrupt is not serviced watchdog asserts a security violation alarm to SNVS now you see this security alarm violation that you get here right. So, this will basically be getting asserted to SNVS, but why is it getting to SNVS.

So, what is so special about SNVS, that this trigger alarm has to go to that component inside my processor because I want in this kind of a scenario also for all my keys and data private data to be cleaned off. So, the moment is alarm is actually generated and it and it is received by SNVS that guy will automatically take care to wipe off the data completely right. So, see when we talk of the security features that too inside the processor every feature that you will realize is paranoid feature what do I mean by paranoid feature here. So, I will always be very suspicious. So, I will never be in the mood of being optimistic or giving the benefit of doubt or whatever it is right anything that I come immediate reaction most the times you will find me cleaning up the private data I am perfectly with the device getting to be unusable state, but I am completely, I have the biggest problem in my life if I basically try to have the data compromised by and accessible by somebody who yes who is not supposed to access it. So, for all events you will find SNVS areas getting cleaned off completely right. So, with all these components one of the first things that we will have to look at is what is the boot sequence that generally happens in the processor.

(Refer Slide Time: 26:04).



IMX6 Boot Sequence

Boot process begins on POR - ARM core begins execution starting from on-chip boot ROM

Boot ROM code uses the state of BOOT_MODE register and various eFUSEs to identify the boot flow of device

The main features of the ROM include:

- Support for booting from various boot devices
- Serial downloader support (USB OTG)
- Device configuration data (DCD)
- Digital signature and encryption based High Assurance Boot (HAB)
- Wake-up from low power modes
- Wake-up secondary core

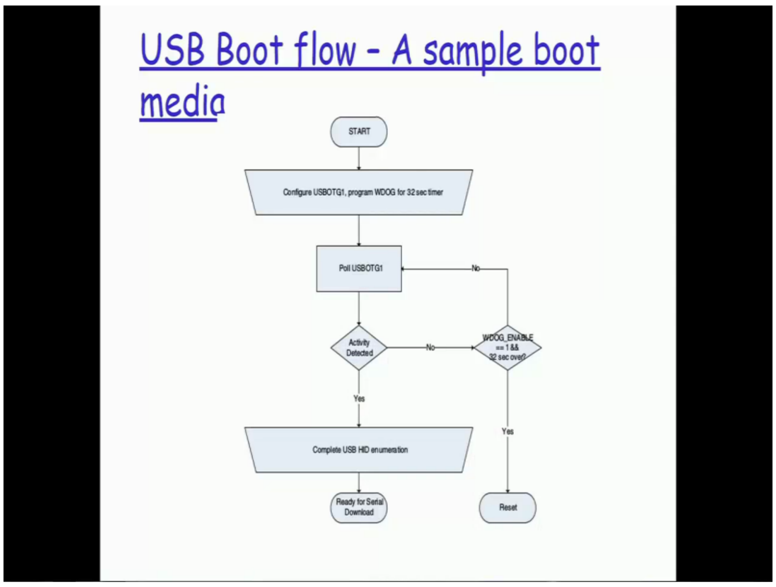
So, the boot process begins with the pure power on reset as we saw. So, the arm core begins execution starting from my on-chip Boot ROM. So, I have a on chip Boot Rom that is there and that code is actually going to start getting executed. Now what does this Boot Rom usually do is, it basically checks the status of one register called the boot mode register. So, this boot mode register is going to tell me a few things saying what are the different boot devices that are configured and supported right now and which is which out of those supported boot devices I am going to be right now booting from. So, where am I going to get my boot loader right. So, we all understand the boot loader right. So, Boot Rom is going to give control to the boot loader boot loader will give control to my OS. OS will mount my route file system and then my system is already correct. So, we are right in the process where Boot Rom is going to be looking at different boot devices which has been configured and out of those boot devices it is going to be expecting the boot loader to be coming in.

So, serial down loader is there support is there. So, I can either download it over USB or I could either typically download it over the network I could typically download it over the serial port right. So, there are different kinds of ah boot devices that are possible in which my boot loader can come in right. So, once the boot loader comes in I expect the OS image also to come in over the same media right. So, if it is for example, a USB. USB, OTG will be ah scanned to see if I have the image coming in from there right then there is a device configuration data.

So, device configuration data will tell me what are all the different peripherals that are actually available and then digital signature and encryption based high assurance boot. So, that is something which we will be looking at it in detail now and wake up from low power modes and wake up secondary core right. So, what is this wake up from low power modes have you ever seen this in other processors why do you have this. So, what is a low power mode. So, what do I do in a low power mode. So, when it is less burden what is the advantage. So, less power consumed right. So, I have a mobile phone. So, every mobile phone is running android only now. So, there is no fundamental difference between the features because any application I can download it from google play. So, in a mobile phone what is a primary differentiator now power so.

What is a battery capability? whether is it working for one day, two day, three days, one week, two weeks, whatever it is. Have you seen a mobile phone with two weeks battery non smart phone is talking about a smart phone. So, the when it is going from a ideal activity or a least activity to a very high performing activity. I have to move from a low power mode to a high power mode right where in I will also possibly be waking up my other cores. So, I might be running only one core when I do not have really many applications, but moment I am fedded with more applications I might have to actually start of the other cores also to make use of them right. So, all these things are actually done as part of the Boot Rom that I there.

(Refer Slide Time: 30:01)



USB Boot flow - A sample boot media

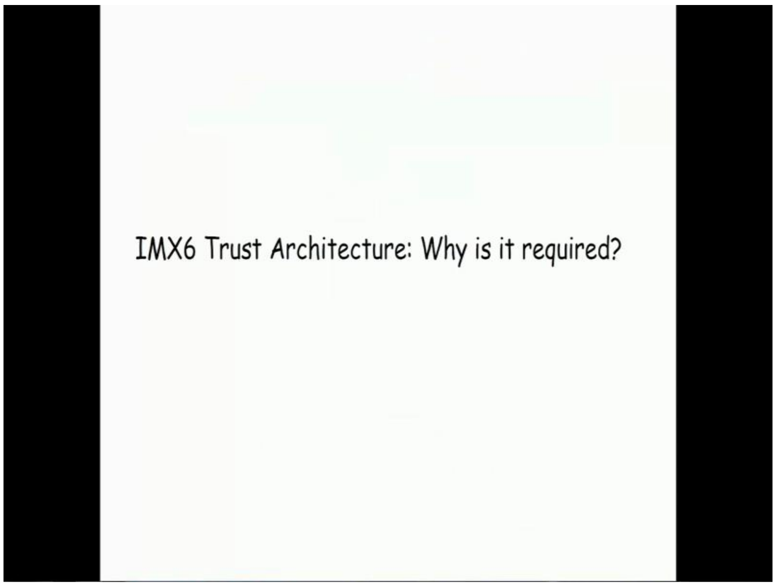
START
Configure USBOTG1, program WDOG for 32 sec timer
Poll USBOTG1
No
Activity Detected
No
WDOG_ENABLE == 1 && 32 sec over?
Yes
Yes
Complete USB HID enumeration
Ready for Serial Download
Reset

So, as I was telling you I could really have the boot loader and the image coming in over the USB. So, when it is actually happened like that the flow that will typically happen is I will check if the USB OTG is configured. Essentially USB OTG is my register which will basically tell that I will need to expect my boot image to be coming in over the USB media. And then I have a programmable watchdog which I will set it for 32 seconds. And then for this period of time I will keep poling that particular media. So, after I have started polling for 32 seconds, if the image has actually started coming in entire image need not have come in, but I should have started scanning the image inside my Boot Rom.

So, if it happens then if I basically detect the activity the enumeration of the USB device. The HID here is a hardware id right USB has this hardware id field with the for which the enumeration has to be completed without which it will not be recognized as a valid device in a system right. So, the enumeration is completed and then it is basically ready for the serial download. Suppose if the activity is not detected I will keep waiting for up to whatever is my watchdog duration configured right. So, for this configure duration of time if I do not really get the image over the USB port, I basically go in for a complete reset meaning this entire [FL] will restart again and it will be in a sort of a perennially hung state unless and until I get the image over this media.

So, likewise for any other possible medium that has being configured for the boot image to come in the same sequence will typically happen every similar sequence will typically happen there.

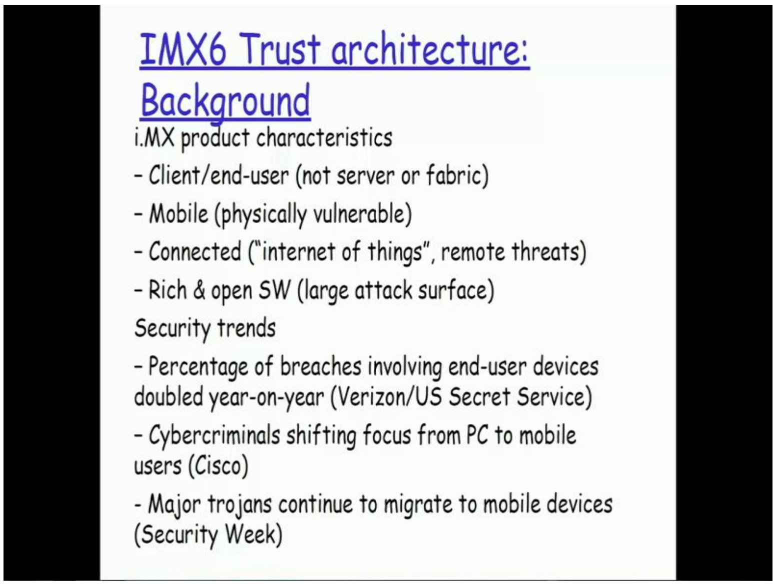
(Refer Slide Time: 32:00)



IMX6 Trust Architecture: Why is it required?

Now, with these kind of different components that is there inside the IMX6 processor, we will take a quick look at the trust architecture and then see how something like a high assurance boot is really implemented.

(Refer Slide Time: 32:10)



IMX6 Trust architecture: Background

i.MX product characteristics

- Client/end-user (not server or fabric)
- Mobile (physically vulnerable)
- Connected ("internet of things", remote threats)
- Rich & open SW (large attack surface)

Security trends

- Percentage of breaches involving end-user devices doubled year-on-year (Verizon/US Secret Service)
- Cybercriminals shifting focus from PC to mobile users (Cisco)
- Major trojans continue to migrate to mobile devices (Security Week)

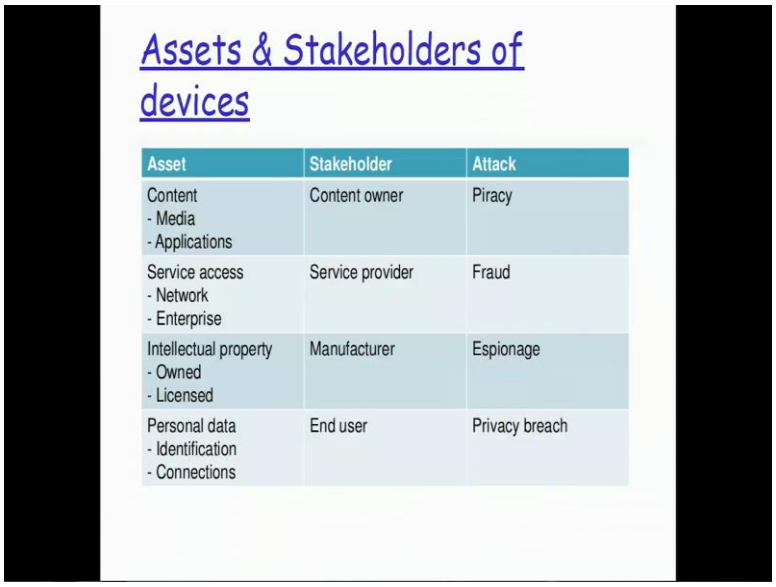
So, with respect to the trust architecture, we will have to understand, where are the devices that are built out of this processor going to be typically made use of right. So, what are the characteristics is going to be a typically a client device or an end user device right. So, it could be something like my mobile phone right. So, it could run a ARM processor, IMX6 ARM processor based mobile phone and that is basically what we have also actually developed right now in the networks lab of IIT. It could be typically a mobile when I say mobile it is physically vulnerable, what do you mean by physical vulnerability somebody can steal it somebody can easily tamper it.

Not very big deal, because we are not going to be putting it in an environment which is going to very secure. So, I just keep the mobile phone there go outside and come back within that time anything could have happen to the phone. It is it is connected with the IOT there. I have complete amount of remote thread that is possible. Moment it is connected. My phone is a continuously having 3G enabled. Or I am connected to my Wi-Fi. Moment it is on the internet a hacker need not be physically close to me he can be totally away anywhere. And I have a rich and open software. The more open software that I have, more software features that I have, more possible sources of attacks correct? So, this is the kind of a device that you are going to be building on a processor like this kind of a ARM processor.

Now for all these kind of threats, you have to have a solution with which, all these threads will be able to be prevented or at least you will be able to know that something like this has happened.

So, what are the different security things that are happening? Percentage of breaches involving end user device is doubling on year on year. Cyber criminals shifting focus from PC to mobile users, because the PC market any way is down. Instead of having one mobile people are starting to have two mobiles. So, more attacks can actually happen on more mobile devices right now, as compared to on stand alone desktop PCs. Major trojans continue to migrate to mobile devices. So, there are possibilities of trojans and getting infected on a mobile device also.

(Refer Slide Time: 37:49)



Assets & Stakeholders of devices

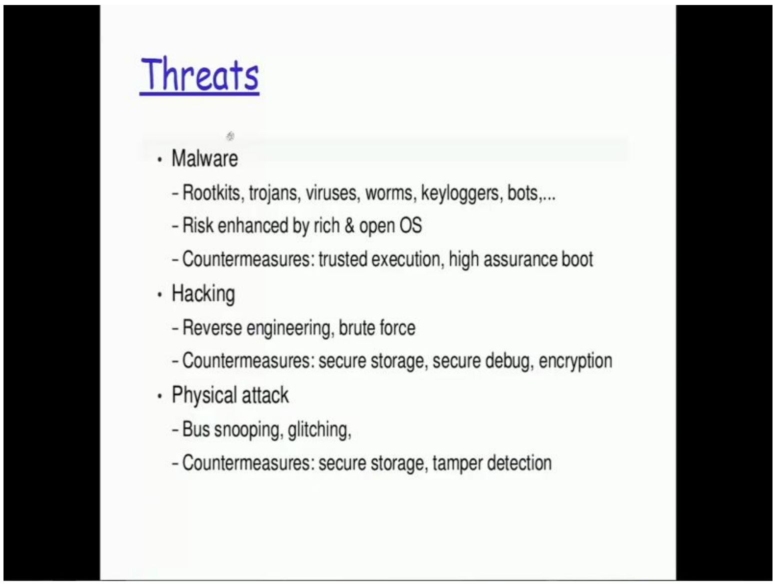
Asset	Stakeholder	Attack
Content - Media - Applications	Content owner	Piracy
Service access - Network - Enterprise	Service provider	Fraud
Intellectual property - Owned - Licensed	Manufacturer	Espionage
Personal data - Identification - Connections	End user	Privacy breach

So, with respect to the different types of a data or the assets, what kind of things could actually happen. So, you if you look at the content. So, whether it is a media or applications or whatever it is. The stake holder of that it basically the content owner and I can be subjected to a piracy attack.

So, Dilwale is running in the movie theaters, I can get the entire Dilwale movie, downloadable from you tube or on a CD or whatever it is very easily. That is the piracy attack. So, somebody has been very smart enough to even record the digitized the HD version of it and then provide it to me. This in terms of the server's access network or enterprise, the service provider is basically the person concerned about this and I can be subjected to a fraud attack. So, fraud attack here could be impersonation. All the possibilities that we saw earlier.

So, intellectual property owned or licensed. So, I might be owning intellectual property data or my customer could have given access to me temporarily for doing his work. Either way the stake holder is basically the original manufacturer and espionage. So, espionage is basically spying right. So, personal data. So, all my identification details. So, you take a photo of your Aadhar card on your mobile phone because everywhere they are asking your Aadhar card and you will use the mobile phone, gone. Correct? So, any body was stolen your mobile phone, just by looking at your photos album of your mobile phone can get access to your Aadhar card number. And you can code that Aadhar card wherever he is convenient for him he can code this own Aadhar card number wherever, it is better convenient for him, right? So, privacy breach. So, these are the different things that could really happen.

(Refer Slide Time: 36:47)



Threats

- Malware
  - Rootkits, trojans, viruses, worms, keyloggers, bots,...
  - Risk enhanced by rich & open OS
  - Countermeasures: trusted execution, high assurance boot
- Hacking
  - Reverse engineering, brute force
  - Countermeasures: secure storage, secure debug, encryption
- Physical attack
  - Bus snooping, glitching,
  - Countermeasures: secure storage, tamper detection

So, in terms of the threats, Malware. So, Rootkits, trojans, viruses all these things are possible especially, when it is a rich and open OS. So, android open OS anybody can download any application from google play. I do not know, who has written the application. What that application is doing internally, no idea. With a very fancy name. I just go by the name and download internally might be doing lot of things.

So, for example, how many of us are familiar with Linux? We have used and you have something like a root user, correct? He is an administrator user of your window system on Linux. So, there is a security thing that is given saying that, the current working directory should never be set in the path of the root user. What is the path? What is the path variable denote? So, directory locations in which I want to find my executable which I want to run.

Suppose let us say there is a program called l s and it has been… we know what l s does, l s basically is the files. So, I had basically logged in as a root user on my system. Just gone out for a cup of coffee, I have a very large screen log time period. I come back before I come back somebody, has used my session window. What has he done? He has copied a program called LS in to my current directory. Cleaned up the history and went away. Now I come back, I do LS. I am in a same current directory right, but what will I be running? I will be running the program that this guy has copied. What is LS suppose to do? LS is supposed to list the files. It will list me the files, but because it is his program, it also would have done something else in the back end. He could have mailed the password file. He could have mailed all files to some id which is otherwise non-readable by him on the system. But I would not come to know of it, because when I run L S I expect the see the list of files, I see the list of the files. So, I am completely oblivious to it of that that l s program has been compromised, correct?

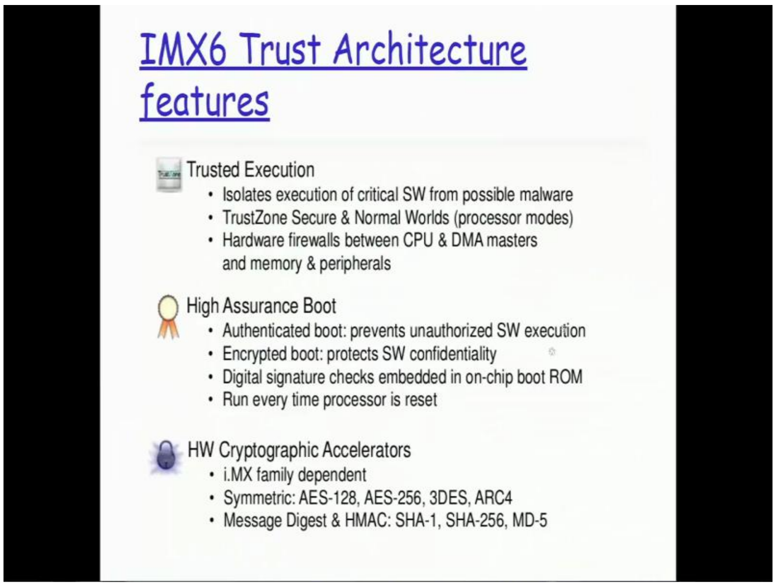
Now that is basically what we call as a root kit compromised. So, there are parts of my file system image which is actually being compromised and I do not really get to know of it and I keep using it which basically keeps sending confidential data out to other people who are not supposed to be accessing the data. So, then hacking. So, reverse engineering or brute force. So, what kind of countermeasures are there you do a secure storage, secure debug and encryption. So, when we say secure storage. What do you exactly mean? How can you do a secure storage?

When I store the data itself I will encrypted in store, right? So, unless and until the person knows the key to decrypt it when it is mounting, he will not be able to access the data right. So, I have an SD card. So, I store some data in it. When I stored the data I will have the entire file system itself created as an encrypted file system and then only store the data on top of that. Somebody really forcibly takes out the SD card and tries to put it in to another SD card reader and plugs it into his machine, unless and until that person has the password, that machine will report him saying that this SD card is corrupted. They will not be able to even recognize that there is a file system on that SD card and the file system is also containing some data inside it.

So, physical attack like bus snooping, glitching and all that is also possible. For that I basically have the tamper detection along with the secure storage. So, the moment I detect a tamper. So, the tamper I gave an example, of a physical tamper. Trying to somebody forcibly opening up the device or whatever it is. But I could also sense a voltage fluctuation, you understand what I am saying? When I am expecting, let us say a

3.3 volt on a particular circuit line. If I find that I am getting a 5 volt before even it hits my components, I will basically arrest it of right. So, there my device getting faulty or get the device getting conned off will be completely avoided. So, the key aspect here, is the moment we say security our minds are all generally tuned saying that, I will have a password access. Password access a first standard school going student will crack it because there are dictionary cracker program which are available which students of that age knows today to run. So, you try to say that I have actually protected it to the password 5 year, 6 year old kid today little bits smart kid will be able to run the dictionary cracker program find out your password unless it is. So, complicated password and have access to the data. We will have to get out of that mode and processor like this IMX6 with the ARM core inside is giving us. So, many benefits that you try to do a complete end to end security. So, that your device which you are building with this kind of a processor is completely secure against any kind of an attack. Physical attack, network attack just plane he is dropping attack right all kinds of attacks are preventable but, we will have to take pains to make all these features and then program it.

(Refer Slide Time: 43:00).



IMX6 Trust Architecture features

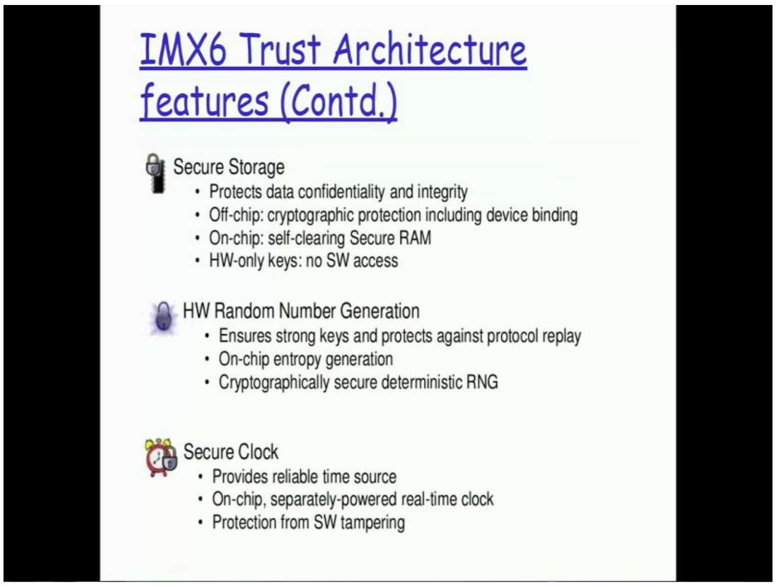
Trusted Execution
- Isolates execution of critical SW from possible malware
- TrustZone Secure & Normal Worlds (processor modes)
- Hardware firewalls between CPU & DMA masters and memory & peripherals

High Assurance Boot
- Authenticated boot: prevents unauthorized SW execution
- Encrypted boot: protects SW confidentiality
- Digital signature checks embedded in on-chip boot ROM
- Run every time processor is reset

HW Cryptographic Accelerators
- i.MX family dependent
- Symmetric: AES-128, AES-256, 3DES, ARC4
- Message Digest & HMAC: SHA-1, SHA-256, MD-5

So, trust architecture features trusted execution, so isolates execution of critical software in possible Malware. So, they will be talked about it hardware firewall between CPU and the DMA masters and memory and peripherals. Who does it memory and peripherals firewalls, who is doing it we just talked about it right now which component is doing it CSU. So, high assurance boot we will talk about it right now. Cryptographic accelerators I have already told about.

(Refer Slide Time: 43:43).



IMX6 Trust Architecture features (Contd.)

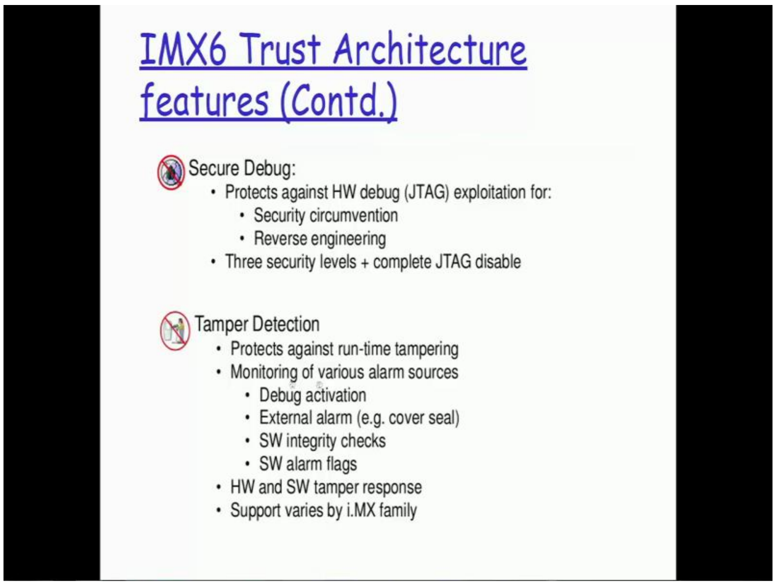
Secure Storage
- Protects data confidentiality and integrity
- Off-chip: cryptographic protection including device binding
- On-chip: self-clearing Secure RAM
- HW-only keys: no SW access

HW Random Number Generation
- Ensures strong keys and protects against protocol replay
- On-chip entropy generation
- Cryptographically secure deterministic RNG

Secure Clock
- Provides reliable time source
- On-chip, separately-powered real-time clock
- Protection from SW tampering

It secure storage hardware random number generation secure clock, all these are the features that are actually available here so.

(Refer Slide Time: 43:50)



IMX6 Trust Architecture features (Contd.)

Secure Debug:
- Protects against HW debug (JTAG) exploitation for:
  - Security circumvention
  - Reverse engineering
- Three security levels + complete JTAG disable

Tamper Detection
- Protects against run-time tampering
- Monitoring of various alarm sources
  - Debug activation
  - External alarm (e.g. cover seal)
  - SW integrity checks
  - SW alarm flags
- HW and SW tamper response
- Support varies by i.MX family

Secure debug, so you have a hardware debug the JTAG platform, that is actually available. So, protects against hardware debug exploitation for security circumvention reverse engineering. So, we all understand JTAG. JTAG is basically the debug port that I have, which I use typically when I am building up a device. During the times of my development.

But the moment the development is done and I still have a JTAG port available on my production device. What can be done? Because the JTAG port actually got access internally to the entire peripheral to the entire memory area. Somebody can use the JTAG port which I have put on my production device to sort of completely over write my CSU and all those checks that I have actually put in. So, in order to prevent that there is something called as secured JTAG that is actually available, that is also part of this. And then the tamper detection. It protects against run time tampering, monitoring of various, alarm sources like debug activation, external alarm like, the cover seal getting off, software integrity check, software alarm flags. All those things are possible to be detecting under the tamper.