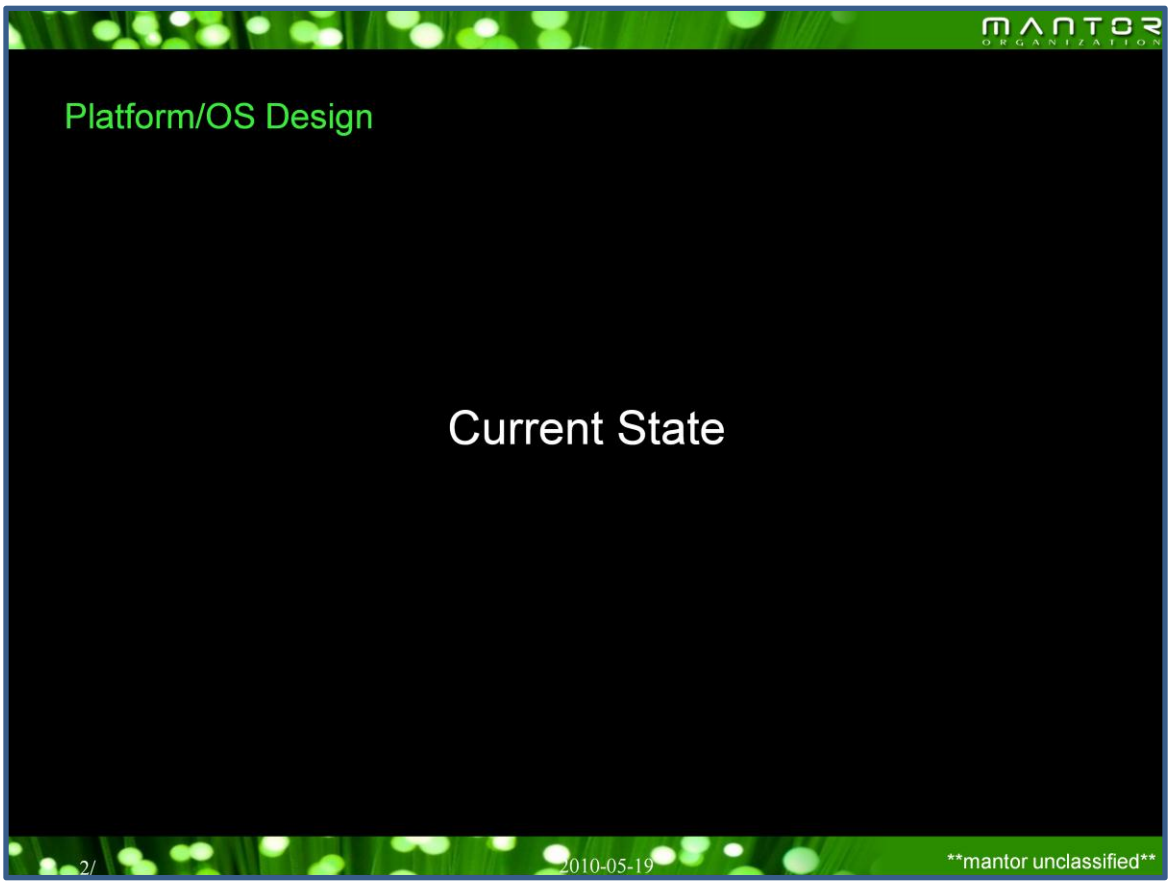
Something's wrong. So much computers compromised, so much in botnets, so much stolen credit cards, everyday sees more zero-days and weakens our posture.

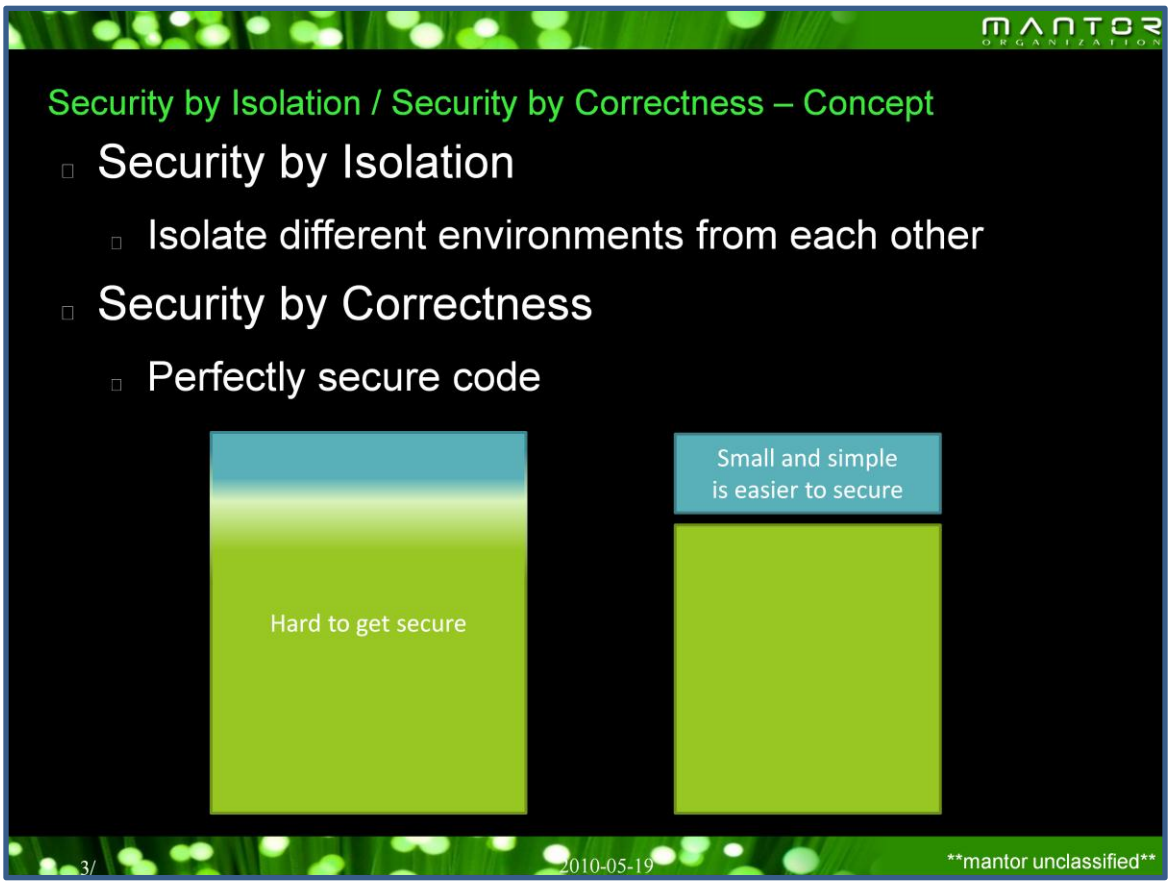
We’re still relying on reactive technology such as anti-virus which can’t keep up with the threat. Do you feel safe after installing the latest security patch? I don’t, not anymore. It’s a chase between us trying to keep up with vulnerabilities. We still heavily rely on too much reaction and too little prevention. We are playing mouse and cat.

I believe we are doing something wrong. I believe we have to change the way we do security. At this point I feel like we are loosing the battle.



The fact is: it's really hard to write secure software. It's hard to justify in the first place and it's really hard to get right. It's been like that since the beginning and nothing has changed.

The question is « what do we do about it? »

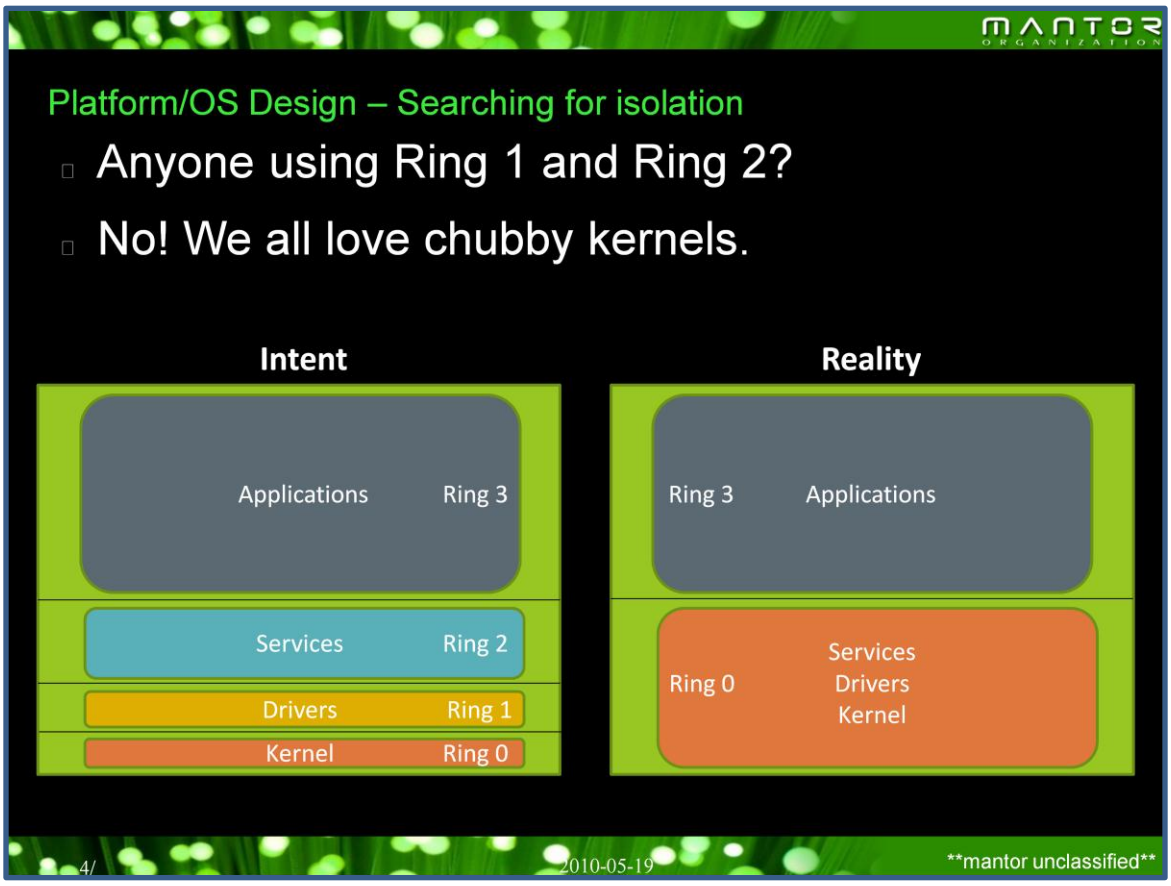


… we use isolation.

Those rectangles represent the code of an application. The blue part is sensitive code which, as an example, might be used to administer the application.

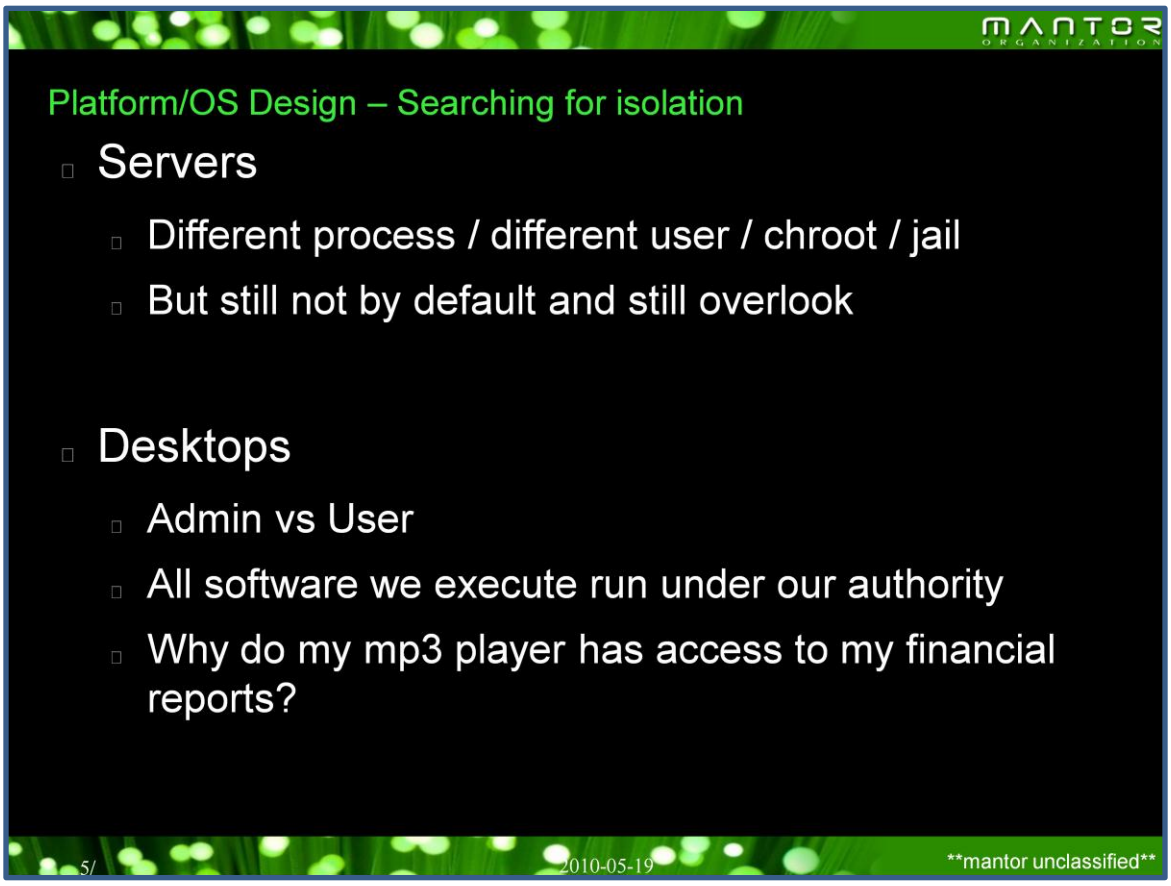
At the left, any bug would compromise the entire application while at the right, the sensitive code is isolated. Thereof, the application on the right is much easier to secure. You can put your « limited » effort/time in securing this part instead of the entire picture.

« Security by Correctness » is highly dependent on size and simplicity. In that sense « Security by Isolation » is a great complement to it.



Architecture (ia32, amd64):

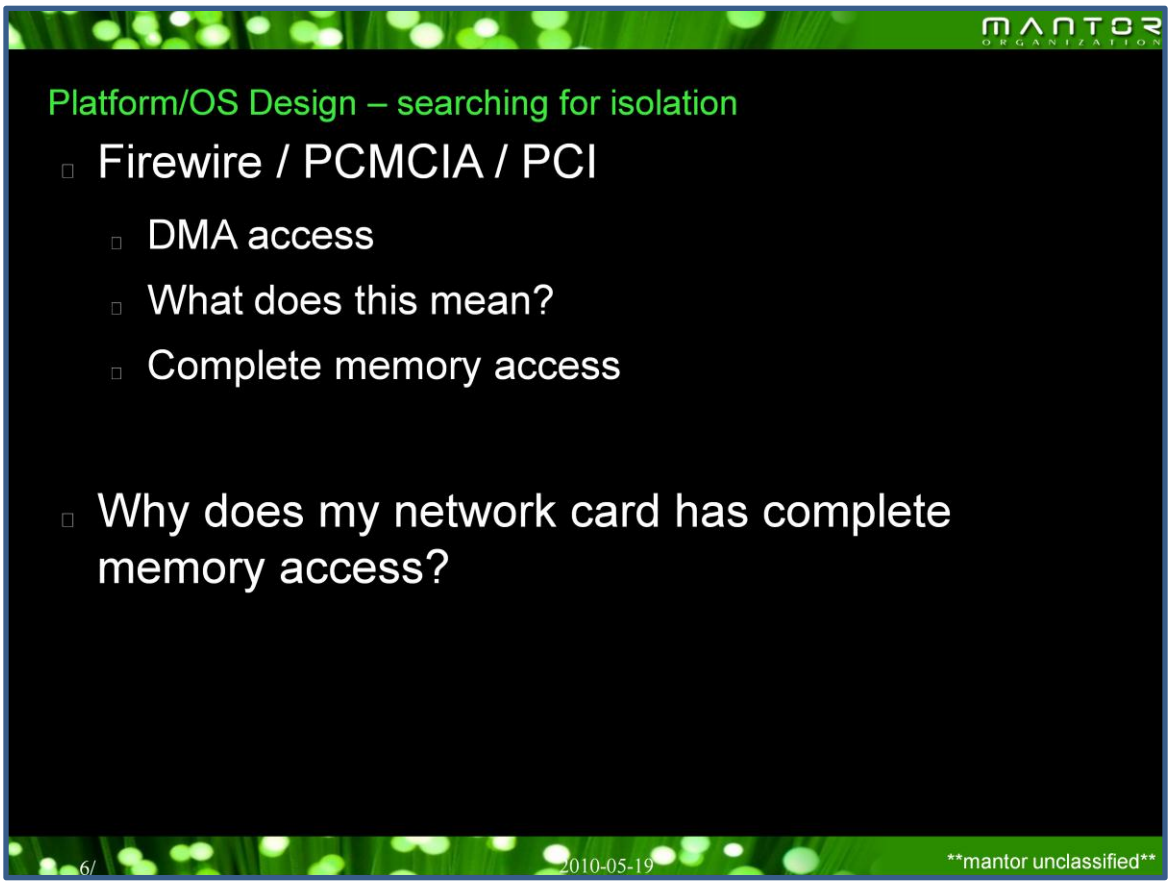
- At first, we tried isolation. The goal was to build a deep Isolation (3 layer) between untrusted code and the kernel.
- Obviouly it dind't work as intended and there is a good reason for this: performance issues, complexity issues.
- The end result? Current OS security model (unix/linux/windows/macosx) are highly dependent on « Security by Correctness ». Any bug in drivers, services or the kernel is a major issue.
- Don't get me wrong, « Security by Correctness » is an important aspect of security but it should not be used at this magnitude.



On the server, we do have some tools but the problem? It's still not defacto. Most company does not use this. It's not available by default, requires custom setup (AIX, HPUX,Solaris, Redhat, most Linux). Still loads of commercial « enterprise » application require administrator/root privilege.

On the workstation side it's a lot worst. Every software you download and run get access to everything you have. I remember seeing some notice such as « don't run Xchat under root privilege »… yeah right, like root had more to protect then I do: everything I need to protect, all my documents are owned by me, not root.

On android each app runs in a different context and have different rights. This is a great start, but still not perfect.

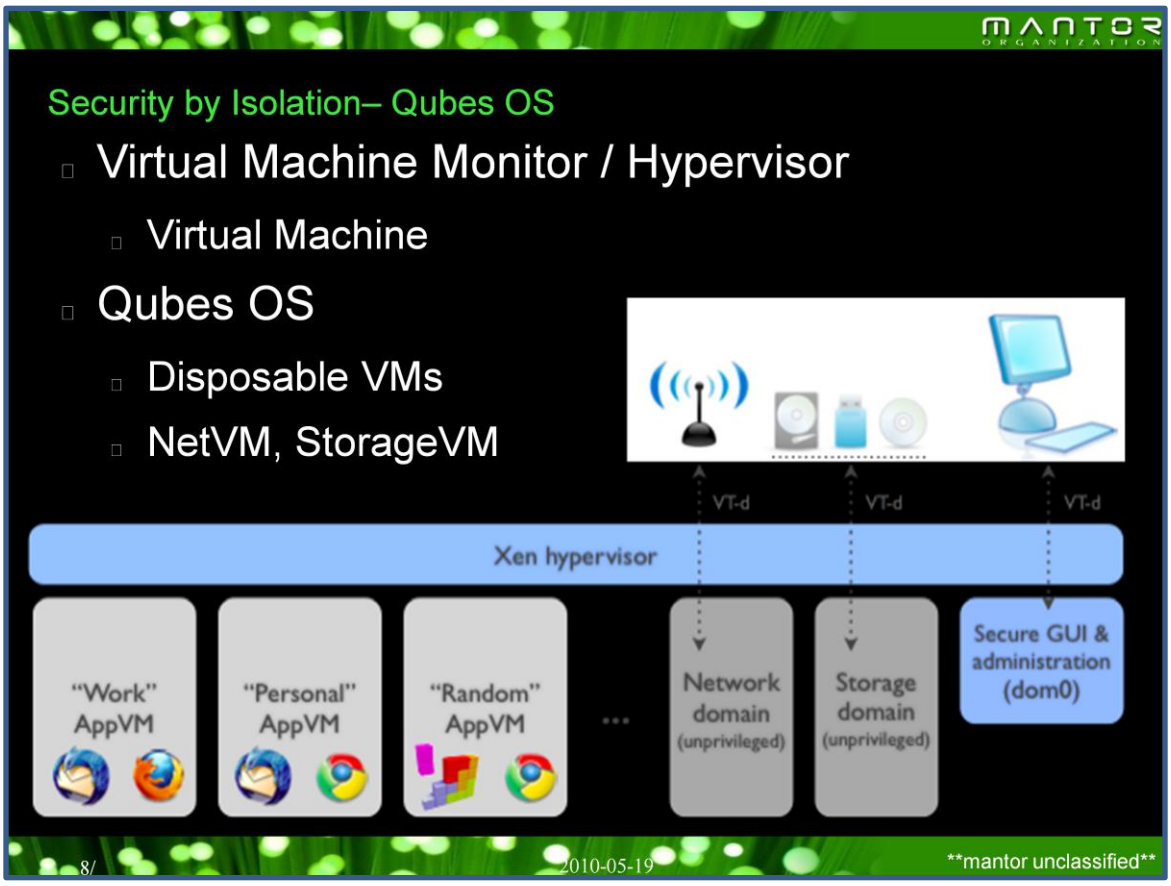


Nothing new for any security guy but I think this one really show how deeply our system architecture is broken.

If I hook a Firewire cable from my laptop to yours, I get direct access to your entire system memory which mean I can unlock your screen, inject a rootkit or whatever. It’s game over. BTW, a PoC is available.

Shouldn't the network card be limited to the network driver memory space?

Really, something's wrong.



I believe Qubes OS should have a lot more attention from the community. This is one model which really has the ability to bring our workstation to a new level.

In order to understand QubesOS you have to free your mind about what you used to think when we say Virtual Machine. We are not talking about VMware server/ESX here. We're talking about running multiple environments isolated from each other.

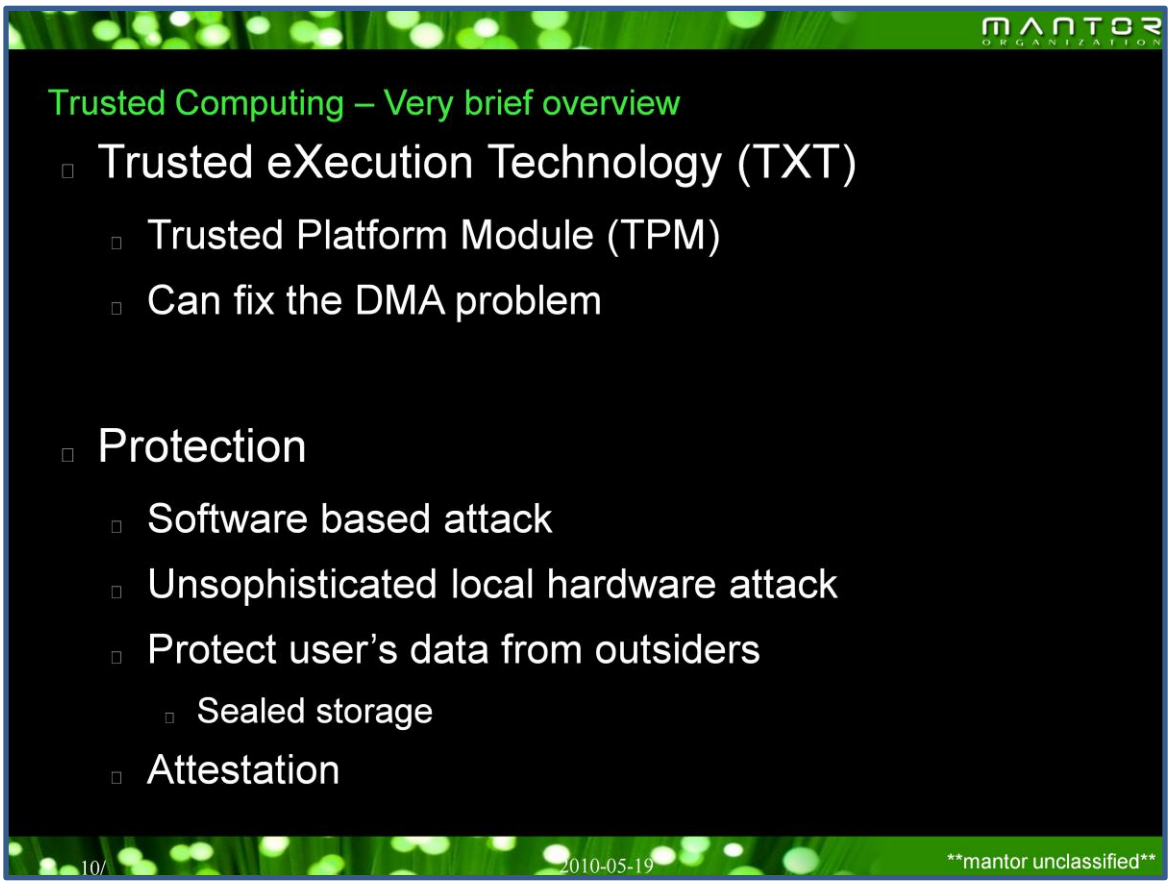
Disposable VM: An environment where you don't care being vulnerable and compromised. You have no personal info in this VM and each time you close the window, everything is restored.

Net VM: Your network stack get compromised? No big deal, it's isolated!

Storage VM…

The different window border colors identifies the different VMs. In this model the VM running my MP3 player wouldn't have access to my financial reports.

The PDF I just downloaded from an untrusted website could be open in a Disposable VM. If the PDF was malicious it wouldn't have access to anything interesting and the malware would vanish as soon as I close it.



This is a really big project which will radically change the way we do security. [explain (Intel, IBM, HP, AMD, etc), VT-d/IOMMU]

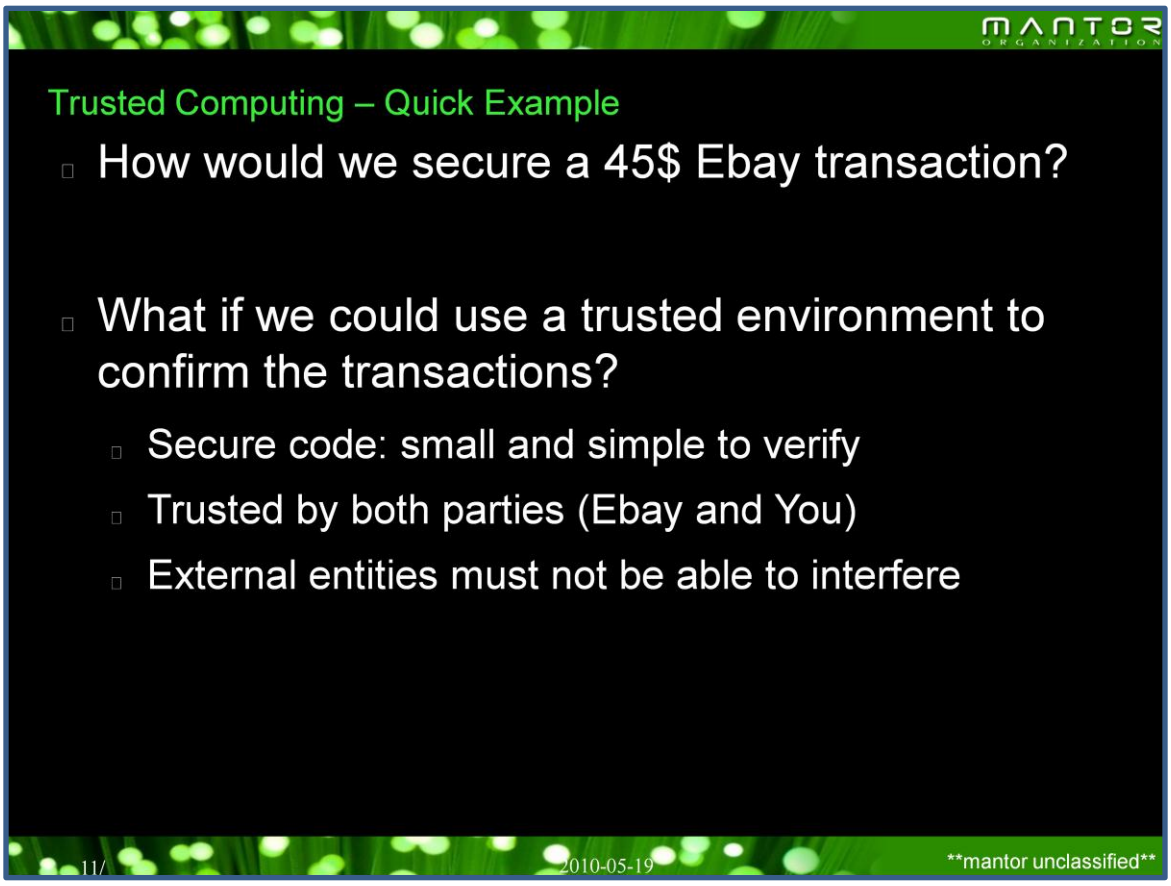
It's very different from other security measures: it's hardware supported.

TXT aka « lagrande » is the Intel implementation of Trusted Computing.

Measurements: ability to determine which environment is running (bios, bootloader, os, etc) in a secure manner.

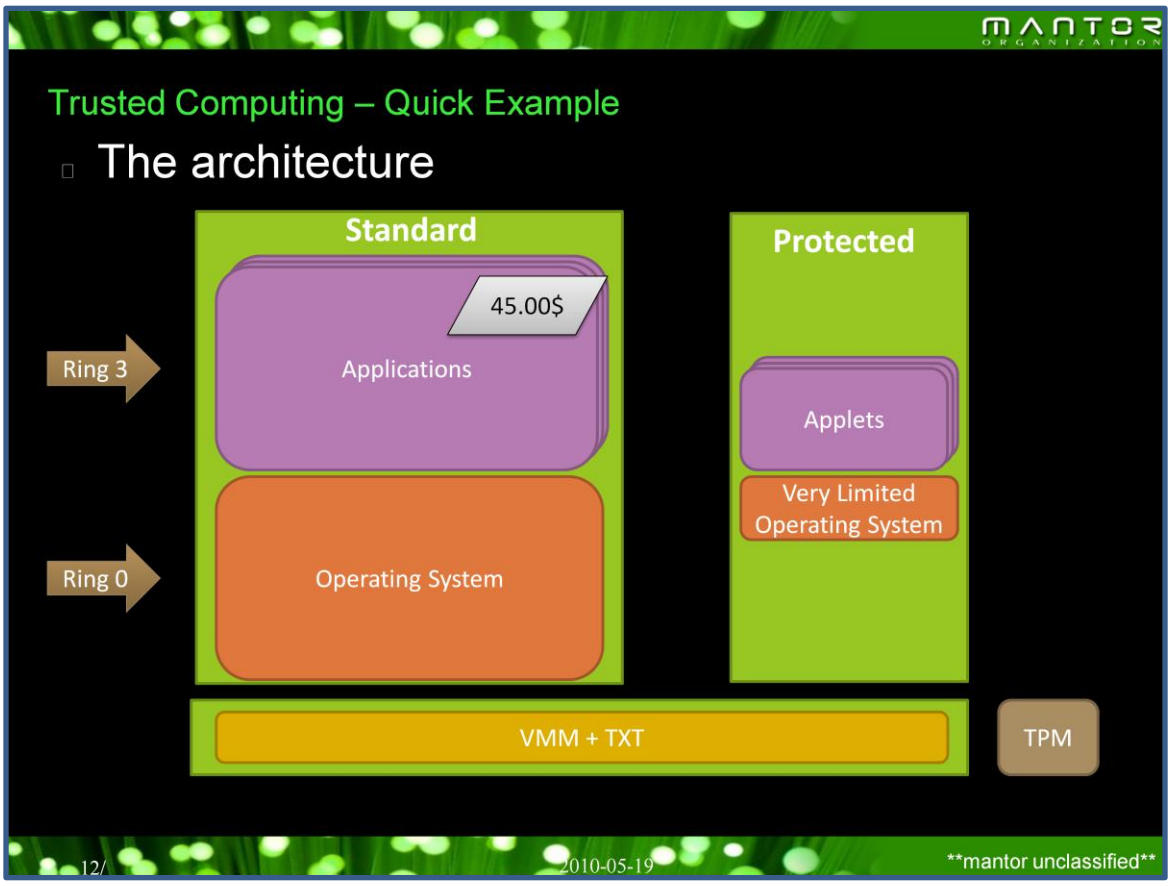
Sealed storage: Unlock data only when a specific environment is running.

For more info a highly recommend reading: Dynamics of Trusted Computing- A Building block approach from David Grawrock Intel Senior Principal Engineer and Security Architect.



Here's a really simple use case which uses some trusted computing mechanism. Trusted computing is a tool and can be used in different ways and this is only a simple example of what can be done.

In this example, the principle is to send a validation request to a trusted environment which is isolated (and can be small and simple) and can be verified (audited).

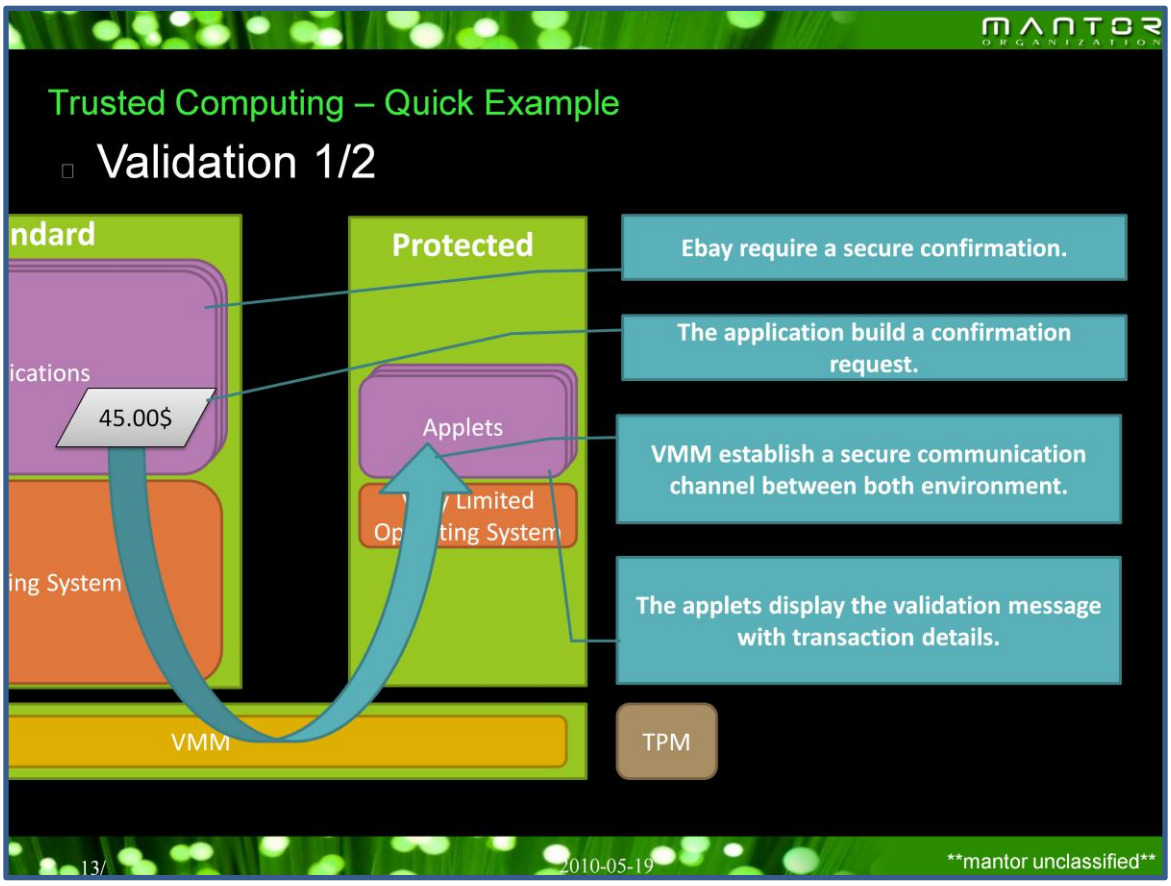


On the left we have a standard operating system, let's say Windows or Linux with Firefox.

On the right side, we have a very limited protected operating system. Protected means protected by TXT. Therefor, it would not be affected by the Firewire (DMA) attack I talked about or whatever else.

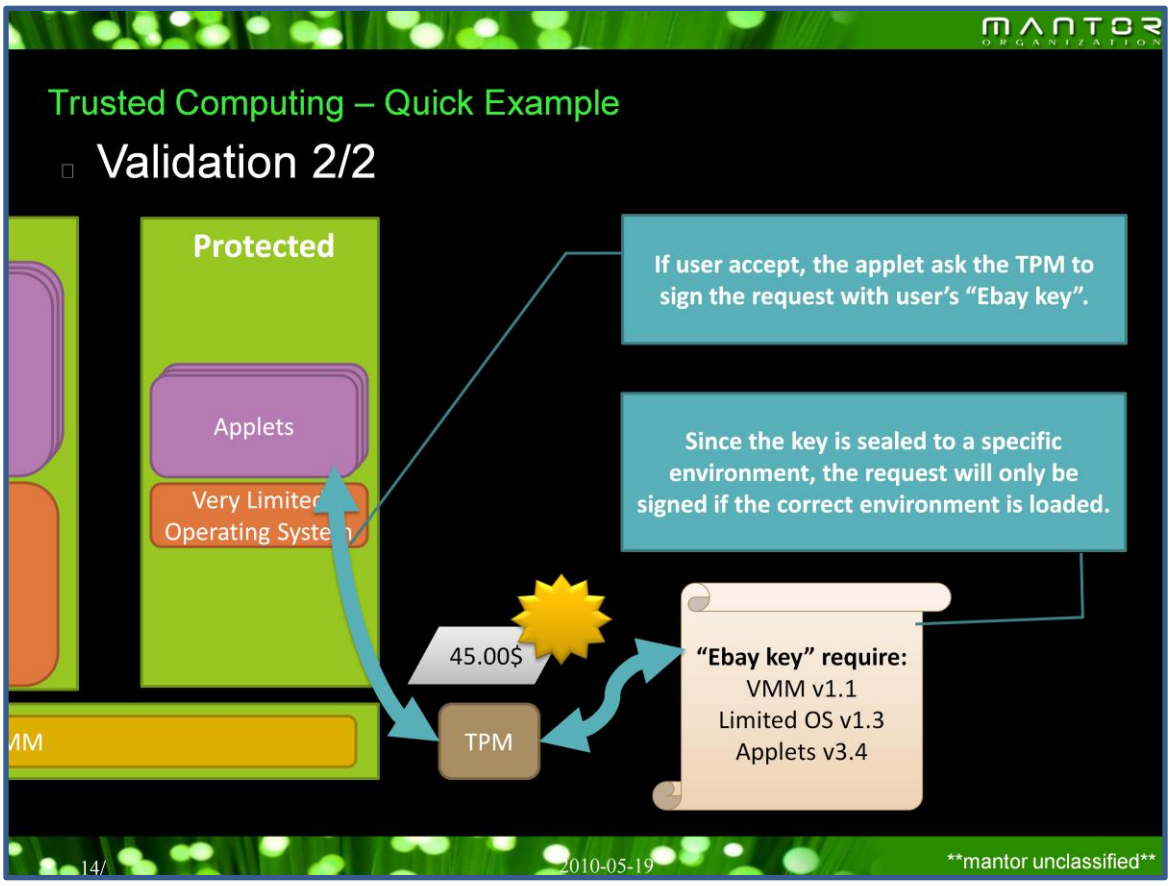
At the bottom we have a VMM/Hypervisor using TXT too.

The VMM and the limited OS are very small and simple software in which we can have a high level of trust (tested/audited).



An XML confirmation request is created and sent to the protected system which displays a confirmation message to user using a “trusted output”: « Ready to buy this item for 45$? ». The user accepts or denies the confirmation request using “trusted input”.

Both “trusted input and trusted output” are paths which cannot be manipulated by the Standard partition.

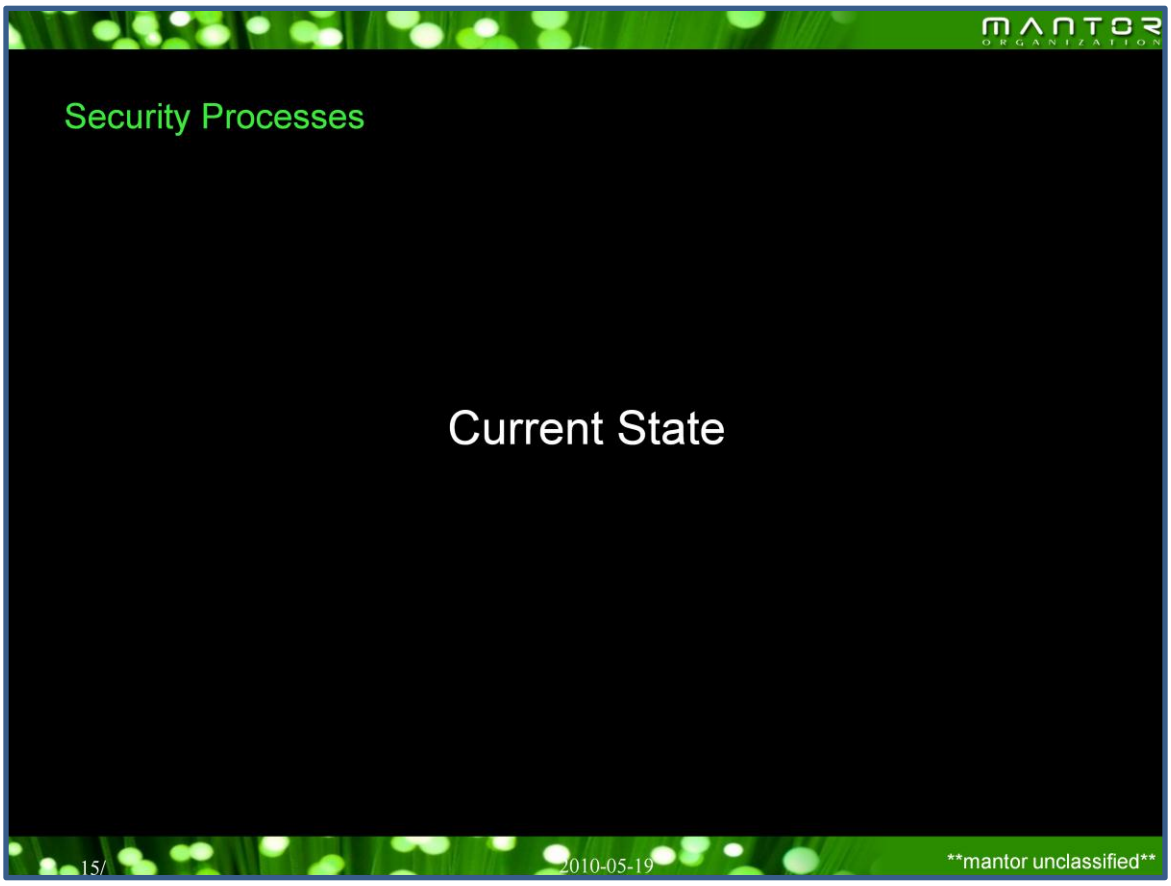


If the user accepts, the applet asks the TPM to sign the request with sealed « user's ebay key ». Since the key is sealed, the TPM will take measurements of the current system and evaluate if the required environment is loaded before signing the request. This measurement process is done by the hardware and it validates every piece of software running (VMM, Limited OS, Applet).

This process ensures the confirmation has been done in a trusted state. No malware, no standard partition, no rootkit can interfere.

This is an excellent demonstration of « security by isolation » and « security by correctness » in the right place (small and simple).

I believe this is a better way of getting real security.



Ok, let's get on something else. The technical stuff is always the easy part in Enterprise. The biggest problem is never on the technical side. It's always, `the Enterprise` itself: paperwork, political issues and so on.

We, the security community, have been putting all our focus on technical stuff and I believe we've been overlooking an important aspect for too long: Security is a Process (Bruce Schneier) nothing else; but what kind of efforts are we putting into getting this process done correctly??!

ISO and NIST are fine but they are audit oriented and they do not tell you how to build this 'security process'. They tell you what you should be doing. Nothing else. The « how » part is somehow missing. At this date, I consider we apply security in a ad-hoc fashion. It is still badly understood and badly implemented in Enterprise.

We have no way to measure our posture. Any new security effort, let's say a new antivirus, could actually weaken our security posture and we couldn't know. At the end of the project we would congratulate ourselves for the good work we *think* we just did. After all this new antivirus sale pitch was promoting a 60% higher rate of detection! What they weren't telling us is that they detected a lot more, but their quarantine functionality sucked. Would you see the increase in workstation reinstallation due to malware infection?

Now we're asked to use external services/infrastructures (clouds) while we have absolutely no clue how they manage security. How can we trust a 3rd party? Does it has something to do with reputation? What processes do they have in place? I want some proof!!!

The community has been putting not much effort on getting those processes right.

In enterprise it's always a question of getting a global picture and putting our energy/money on what requires it the most.

I believe Enterprise security should be just like a game. We should be able to see our score and find out if we did better then the previous months.

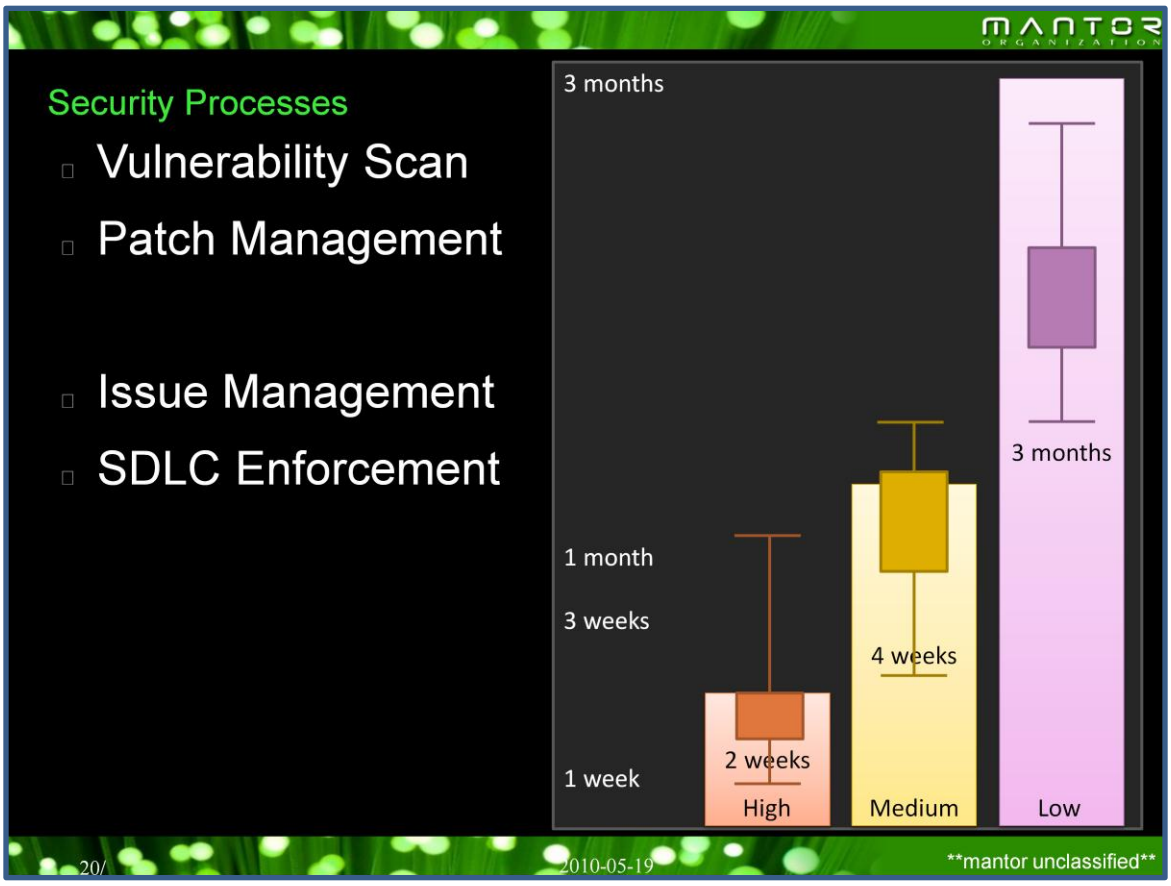
But the problem is that we don't have any *metrics*. We're blind.

The idea we had is to build a security standard as we do « Test-Driven Development ».

A standard which would take form depending on what you *really* put in place (measured). A standard which is brought alive instead of being an ideal we fantasize about.

Let’s say: all interactive access must be strong (2-factor, encrypted, strong password (8 char). The standard would test and ensure telnet/ftp are disabled, pam require 8 char password and 2-factor authentication.

The software is no magic, he won’t do security for you but will help you getting a structure and better visibility. It is presently develop in Ruby on Rails 3 while probe/agent are written in mostly any language.



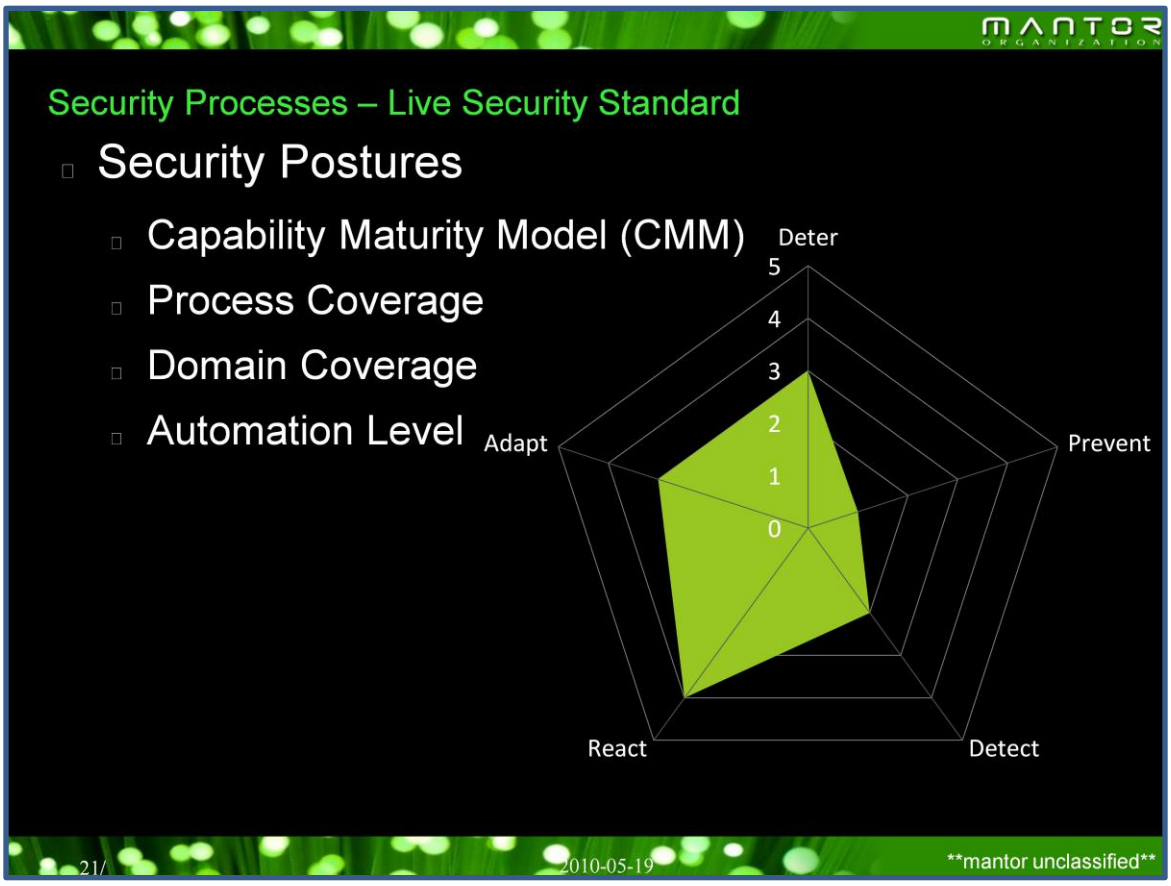
Some are manual controls, some are automated.

What if we could see exactly how we score. Example:

Instead of saying we patch high vulnerability within 2 weeks, speak the truth and get some metrics!

SDLC: Each time you deploy a new software.

Depending on the classification it would require your developer to comply to the standard: peer review, threat modeling, vulnerability testing, penetration testing with proof and approval.



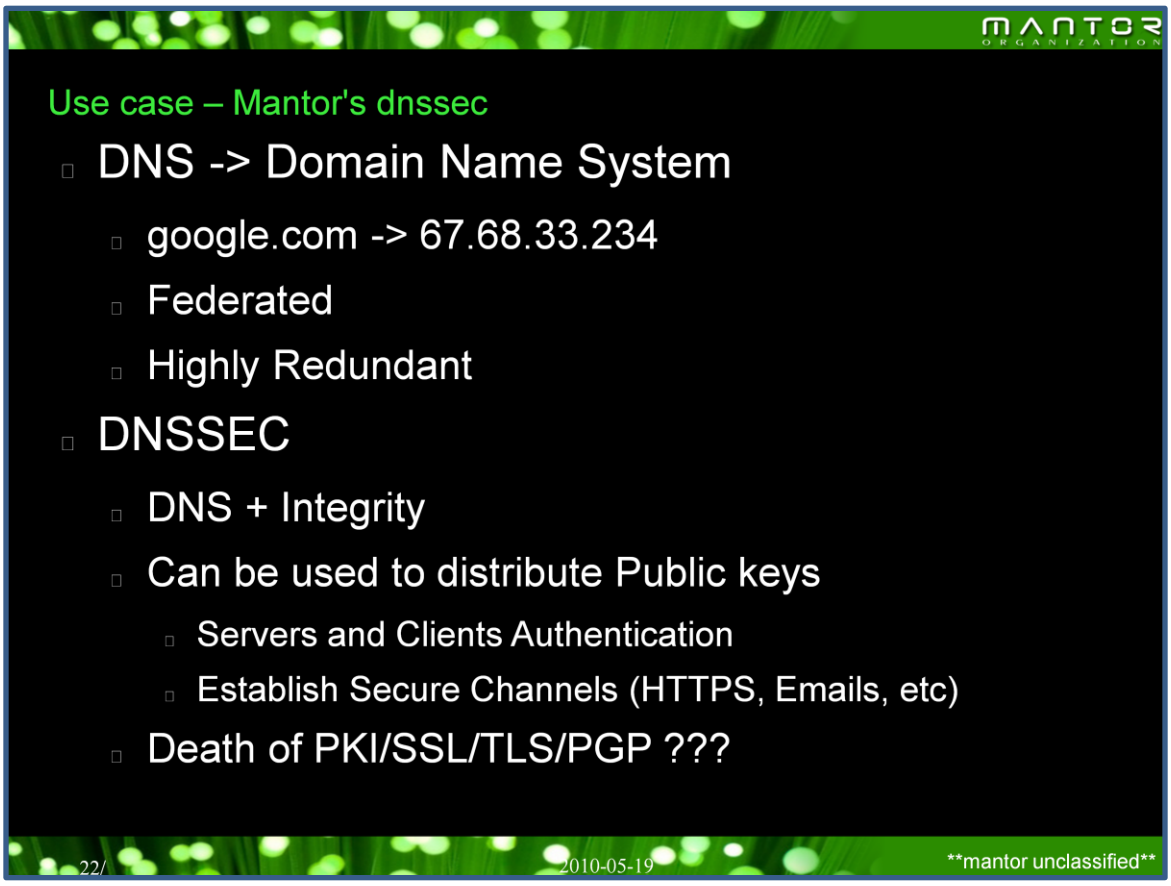
What type of controls do we most have in place? (Preventive? Detective? Reactive? Etc.)

How mature are we within each process? (patch management CMM 2)

Process coverage: what part of my policy do I cover with tests? (Oh I have very few test on “physical access management”!)

What security domain do I neglect? (Hum, we have no controls on the “human resource background check”)

All of this is be address by this “live security standard”.



In the next few months, Mantor will be providing a DNSSEC management service. Since this a security product, I don't get to see how people would let us manage their DNS record without trusting us. They need to known what we do from a security standpoint… Well, that's what we'll do. We'll use the « live security standard » application to provide insight into how we manage and maintain the security of our infrastructure (2011).

At the end, if we leverage « security by isolation » to use « security by correctness » at the right place and we work on getting metrics out of our « security processes », I believe we will bring information system security to a all new level.

Links

- Mantor: www.mantor.org
- Blog: blog.mantor.org
- OpenSec: dev.mantor.org/opensec
- Qubes OS: www.qubes-os.org
- Trusted Computing: www.trustedcomputinggroup.org
- TrouSerS: trousers.sourceforge.net
- Bsssd: bsssd.sourceforge.net
- Firewire attack: www.hermann-uwe.de/blog/physical-memory-attacks-via-firewire-dma-part-1-overview-and-mitigation