Symbian OS

Symbian OS 9,
Platform Security

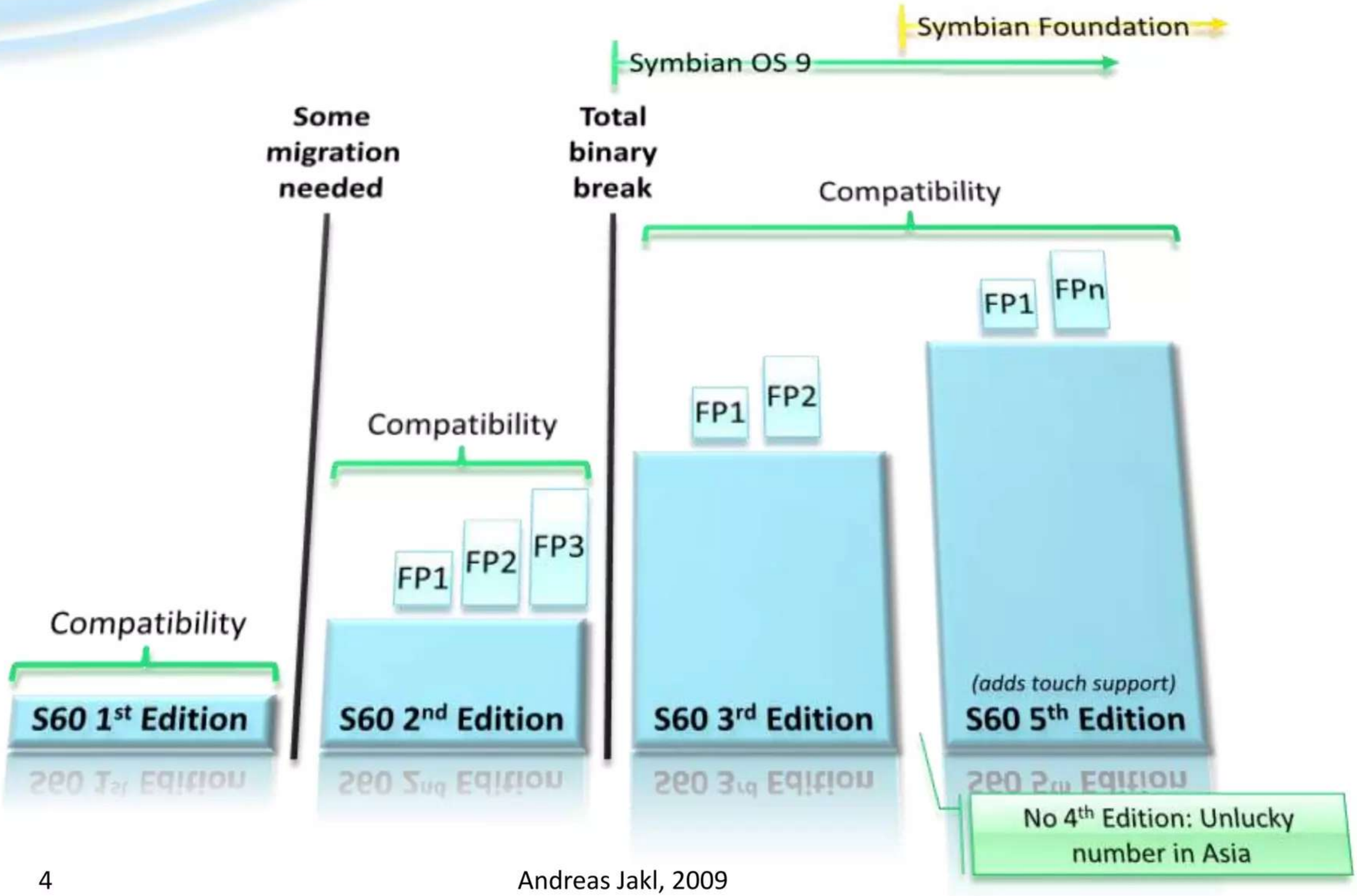
Disclaimer

- **These slides are provided free of charge at http://www.symbianresources.com and are used during Symbian OS courses at the University of Applied Sciences in Hagenberg, Austria (http://www.fh-hagenberg.at/)**
- **Respecting the copyright laws, you are allowed to use them:**
  - **for your own, personal, non-commercial use**
  - **in the academic environment**
- **In all other cases (e.g. for commercial training), please contact andreas.jakl@fh-hagenberg.at**
- The correctness of the contents of these materials cannot be guaranteed. Andreas Jakl is not liable for incorrect information or damage that may arise from using the materials.
- Parts of these materials are based on information from Symbian Press-books published by John Wiley & Sons, Ltd. This document contains copyright materials which are proprietary to Symbian, UIQ, Nokia and SonyEricsson. “S60™” is a trademark of Nokia. “UIQ™” is a trademark of UIQ Technology. Pictures of mobile phones or applications are copyright their respective manufacturers / developers. “Symbian ™”, “Symbian OS ™” and all other Symbian-based marks and logos are trademarks of Symbian Software Limited and are used under license. © Symbian Software Limited 2006.

Contents

- Platform Security
- Trust Model
- Capabilities
- Data Caging
- Identifiers (UID, SID, VID)
- Developer Certificates
- Symbian Signed
  (Open Signed, Express Signed, Certified Signing)

S60 Compatibility



Motivation for OS 9

- **Expand market** from High-End to Mid-Tier
  - Cheaper manufacturing
- **Demands of shareholders**
  - Licensees and network operators
- **Market**
  - Secure, robust and efficient phones
  - DRM and m-Commerce
  - Platform for mass market services

Improvements in OS 9

- **Hard Real-time Kernel** (EKA2)
- New **tool chain** (ARM Application Binary Interface, ABI)
- **Platform Security**
- Additional concepts for **Inter Process Communication** (IPC, e.g. Publish & Subscribe)
- Closer to **ISO C++**
- **However:**
  - No binary compatibility to < v8
  - Several source compatibility breaks

Compiler

- Pre-v9: GCC 2.98
- **ARM RVCT Compiler**
  - Makes optimal use of ARM v5, v6, …
  - 5-10% smaller application size, better performance
  - Used to compile Symbian OS and ROM-applications
- **GCC-E** (2.9)
  - Free compiler (comes with Carbide.c++)
  - Used for most 3rd party applications
- Both compilers compatible because of EABI

Closer to ISO C++

- **Supports try/catch/throw**
  - For porting from other platforms
  - Can not be combined with Symbian OS system APIs
  - → continue using TRAP/Leave for your own projects!
- **Writable static data for DLLs**
  - Simplified porting process
  - Downside: large overhead

P.I.P.S. / OpenC

- Available since 2007
- **P.I.P.S:**
  - Allows using C standard libraries (POSIX) on Symbian OS
  - OpenC (Nokia) adds additional libraries
- Useful for porting applications and game development
- UI development still requires Symbian-dialect of C++

Symbian OS v9.5

- **Some highlights:**
  - **Performance improvements**
    - e.g. **Demand Paging**: Phone loads only required parts of an DLL instead of the whole file → performance boost
    - Application start-up time improved by up to 75%
  - **SQL** integrated
  - New API for all **location based services**
  - **“FreeWay”: Better networking**
    - Seamless switching from Wi-Fi to 3G networks for VoIP, email, ...
  - **“ScreenPlay” and new multimedia features**
    - e.g. different UI layers, UI acceleration, panorama stitching, red eye correction
  - **Digital TV** (DVB-H, ISDB-T)
  - **ActiveSync** support

The concepts behind …

Platform Security

Pre-v9: Perimeter Security

- **Security checks**
  - User is asked during installation
  - Checks origin of application
  - After installation: complete access to device
- **Symbian Signed**
  - No trust-warning during installation
  - Same access rights as unsinged apps.

Platform Security – Reasons

- **Trust**
  - Access to private data
- **Security**
  - Cost control
  - Compromise important telephone functionality
- **Protection**
  - Software ownership
  - DRM

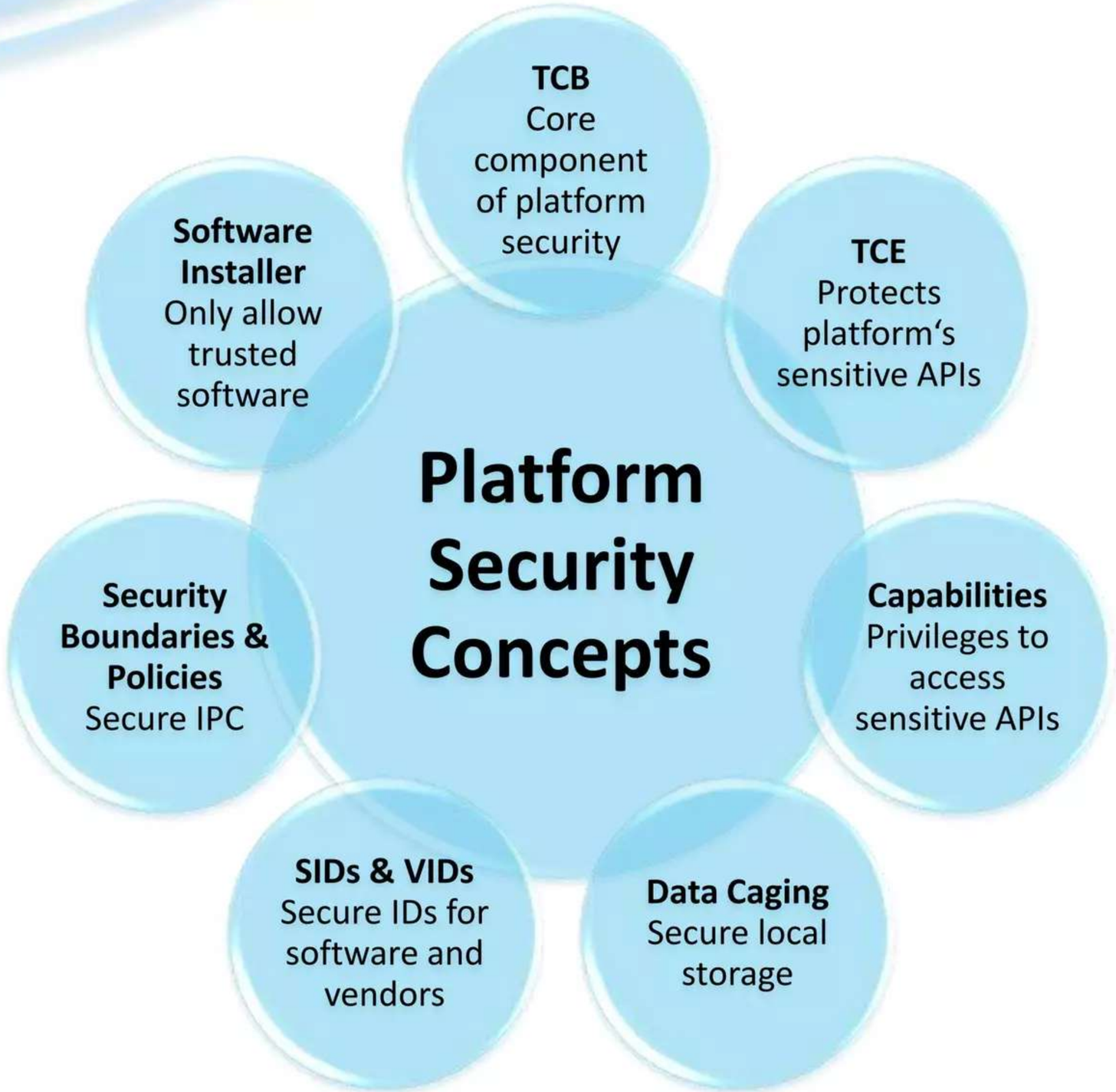
Platform Security – User Side

- **With Platform Security normal users have …**
  - No surprise on their telephone bill
  - No virus
  - Phone simply works when needed
  - Private data stays private
- **… don’t have …**
  - Cryptic security warnings
  - A lot of security decisions

What is it all about?

- **Platform Security IS:**
  - Protection of telephone integrity
  - Protection of sensitive data
  - Controlled access to sensitive functionality
- **is NOT:**
  - Encryption of data
  - Virus-Scanning
  - Key management (Public Key Infrastructure)

Platform Security – Concepts



Trust Model

What's "Trust"?

- **Process**

  = **unit of memory protection** (virtual address space)

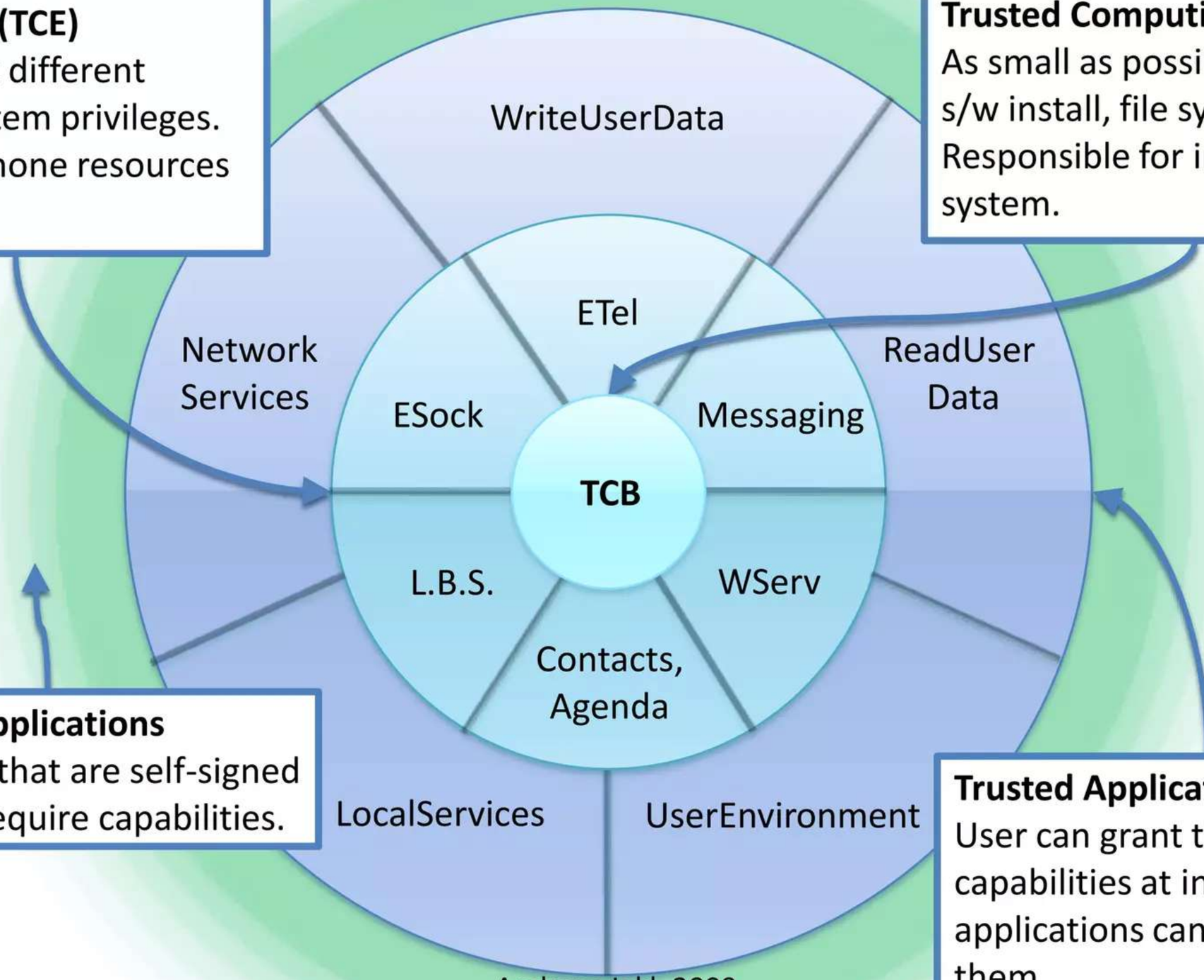
  = **smallest unit of trust**

- **Platform security** controls what a process can do
  - OS prevents access service-request if process does not possess required privilege
    (= it's not trustworthy enough)

Trust Model

Trusted Computing Environment (TCE)
Servers run at different restricted system privileges. Protect the phone resources from misuse.

Trusted Computing Base (TCB)
As small as possible (kernel, s/w install, file system access). Responsible for integrity of the system.



Untrusted Applications
Applications that are self-signed and do not require capabilities.

Trusted Applications
User can grant these capabilities at install time OR applications can be signed for them

Additional Details

- **Trusted Computing Base** (TCB)
  - Highest level of privilege, careful code checks
- **Trusted Computing Environment** (TCE)
  - Each component only has privileges to carry out defined services
  - e.g. Window Server (WServ): Privileged access to screen, no phone network required
  - Provides **APIs for software** outside TCE
    - Only TCE needs to communicate with hardware
    - **TCE-Servers** responsible for moderating and **protecting resources**
    - Communication using **client-server framework**

Additional Details

- **Trusted (Signed) Applications**
  - **Need privileges** to access services provided by TCE (= app. needs to be signed)
  - Various signing methods available (Symbian Signed)
- **Untrusted software**
  - … is “self-signed” (own, temporary certificate)
  - Enough for e.g. Solitaire game, etc.

Defining privileges:

Capability Model

Capabilities – Model

- *„How trustworthy is the application?“*
  - The closer to the kernel, the more trustworthy it has to be
- **Capabilities check level of trust**
  - Have to be defined for an application when compiling
  - Managed by the kernel
  - Can not be modified after installation
  - Defined for each process
  - Choose capabilities depending on required functionality

Capabilities

- **User Capabilities**
  - Can be allowed by the user
  - Easy to understand
  - App. has to be at least self-signed
- **System Capabilities**
  - App. has to be Symbian Signed
  - Apps can go deeper into the system
- **Self-signed Applications**
  - Access to functions that don't require capabilities
    + those that can be allowed by the user (User Capabilities)

API Access

Basic Capabilities

Extended Capabilities

Self-Signed (~ 60%)

Not classified, no capability associated

User-Grantable Capabilities – warning upon installation when self-signed.

Symbian Signed (~ 40%)

APIs can be accessed only through signing the application

Phone manufacturer approval

Capabilities – Overview

User Capabilities (user grantable)

- **Read/WriteUserData** (confidental user data)
- **NetworkServices** (Networking, Dialing, Messaging)
- **LocalServices** (Local connectivity: Ir, Bt, USB)
- **UserEnvironment** (Information about user and environment, eg. Camera)
- **Location** (Phone location)

System Capabilities

- **Read/WriteDeviceData** (Sensitive data)
- **PowerMgmt** (Kill process, turn phone off)
- **SwEvent** (Capture & Generate Sw Key/Pen Events)
- **TrustedUI** (Can not be influenced by other apps)
- **SurroundingsDD** (access according device drvs.)
- **ProtServ** (Start reg. server with prot. name)

Certified Signed / Publisher ID

- **DiskAdmin** (eg. formatting a drive)
- **CommDD/MultimediaDD** (Access to device drivers, deeper control)
- **Network Control** (Modify or access network protocol controls)

Device Manufacturer Capabilities

- **DRM** (Access to protected content)
- **AllFiles** (System files-visibility, extra/private write access)
- **TCB** (Access to /sys and /resource directories)

Define Capabilities

- Capabilities defined in MMP file

```
// program123.mmp
TARGET program123.exe
TARGETTYPE exe
UID 0x00000000 0x00000123
SOURCEPATH ..\mysource
SOURCE myfile.cpp
USERINCLUDE ..\include
SYSTEMINCLUDE \epoc32\include
...
CAPABILITY ReadUserData WriteUserData
```

Carbide.c++ MMP File Editor:

More about Capabilities

- **Capabilities are not hierarchical**
  - Having TCB-capability != having everything else
  - Each capability allows access to **specific protected resource**
- **Kernel** holds **list of capabilities** of each process
  - When offering services: Kernel can be asked to check capabilities of calling process
- **Software Installer**
  - **Gatekeeper**, validates if program is authorized
  - Refuses to install if it’s not

Verification of Capabilities

- Required capabilities can depend on parameters
- **Example: CFileMan::Copy()**
  - **Checks for AllFiles-Capability** when accessing secured directories (e.g. \private\<other SID\>)
  - **Not checked** when accessing public or own directory (\private\<own SID>\)
- **→ AllFiles normally not needed**

Copy ()

Capability: AllFiles

Capability: Dependent — If the path for anOld begins with /Sys then Tcb capability is required. If the path for anOld begins with /Resource then Tcb capability is required

```
IMPORT_C TInt Copy(const TDesC &anOld, const TDesC
&aNew, TUint aSwitch=EOverWrite);
```

Description

Copies one or more files.

Verification of Capabilities

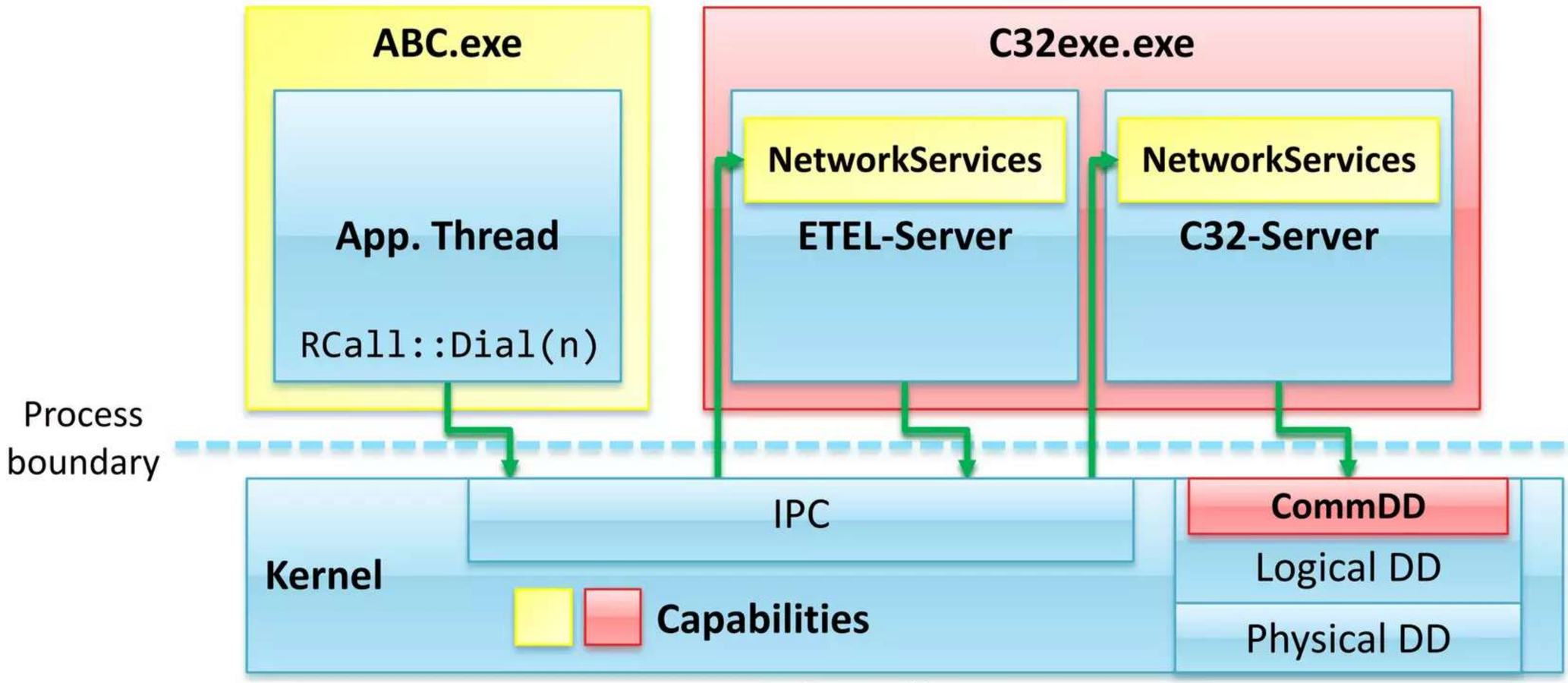
- **Rule 1: Capabilities of a process never change**
  - No method to extend or to limit capabilities
  - Not even through loading DLLs
- **Rule 2: Process can not load DLL with less capabilities than it has itself**
  - DLL-Code runs with the capabilities of the process
  - DLL can have more capabilities

Implications

- **DLLs with interfaces have to have enough rights**
  - e.g. simple signal processing DLL might need capabilities for accessing telephony functionality as well
- **Plug-in DLLs run with same rights as the host process**
  - e.g. MTMs have the same trust-level as the messaging server

Capabilities – Client/Server

- **Capabilities checked at process boundaries**
- **Calling process:** Must have enough capabilities for calling Server-API



Capability Scanner

- Available in Carbide.c++ 1.3+
- Gives an **estimate of which capabilities might be required** (static code analysis)
- Project → Run Capability Scanner on Project MMP...

```
    // Send the request to the asynchronous service provider
    iTelephony->GetCurrentNetworkInfo(iStatus, iNetworkInfoV1Pckg);
    // We're waiting for the callback...
    SetActive();
```

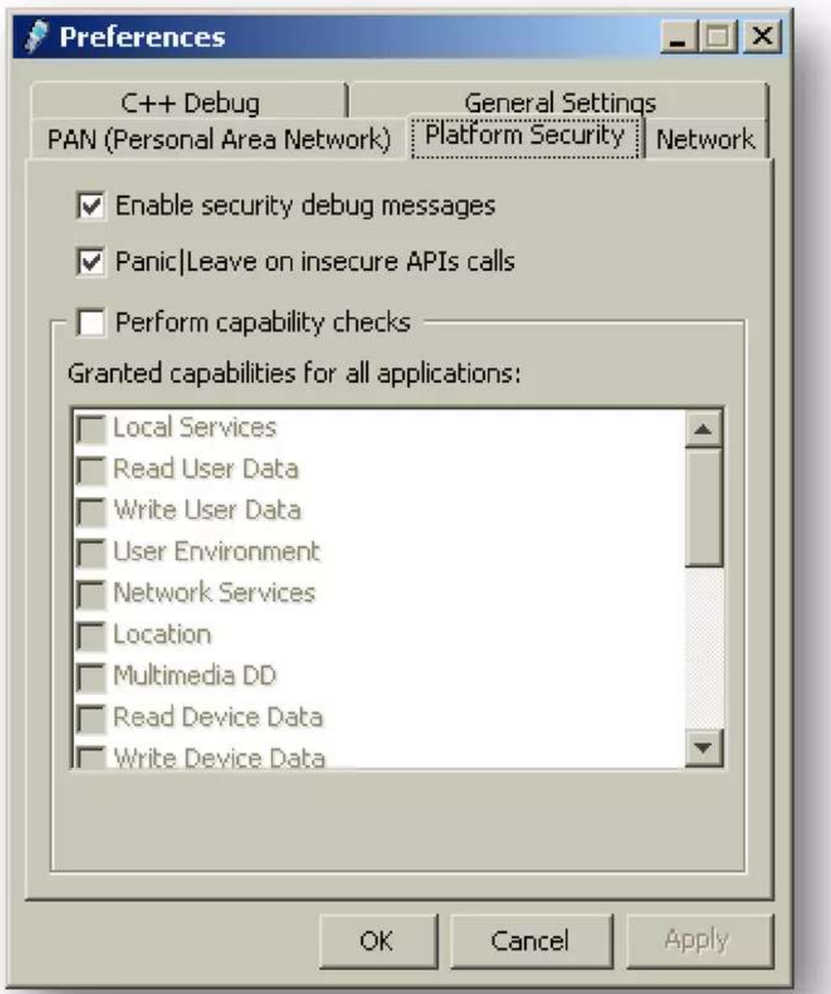
Console | Progress | Dependency Explorer | Outline | Platform Security

The platform security results are only an estimate of what capabilities your application requires. Always make sure to perfor

CapabilityName	Type	Function	File	Location
READDEVICEDATA	Unconditional	CTelephony.GetCurrentNetworkInfo	Journey/src/SysInfo.cpp	Line: 74

Capabilities in the Emulator

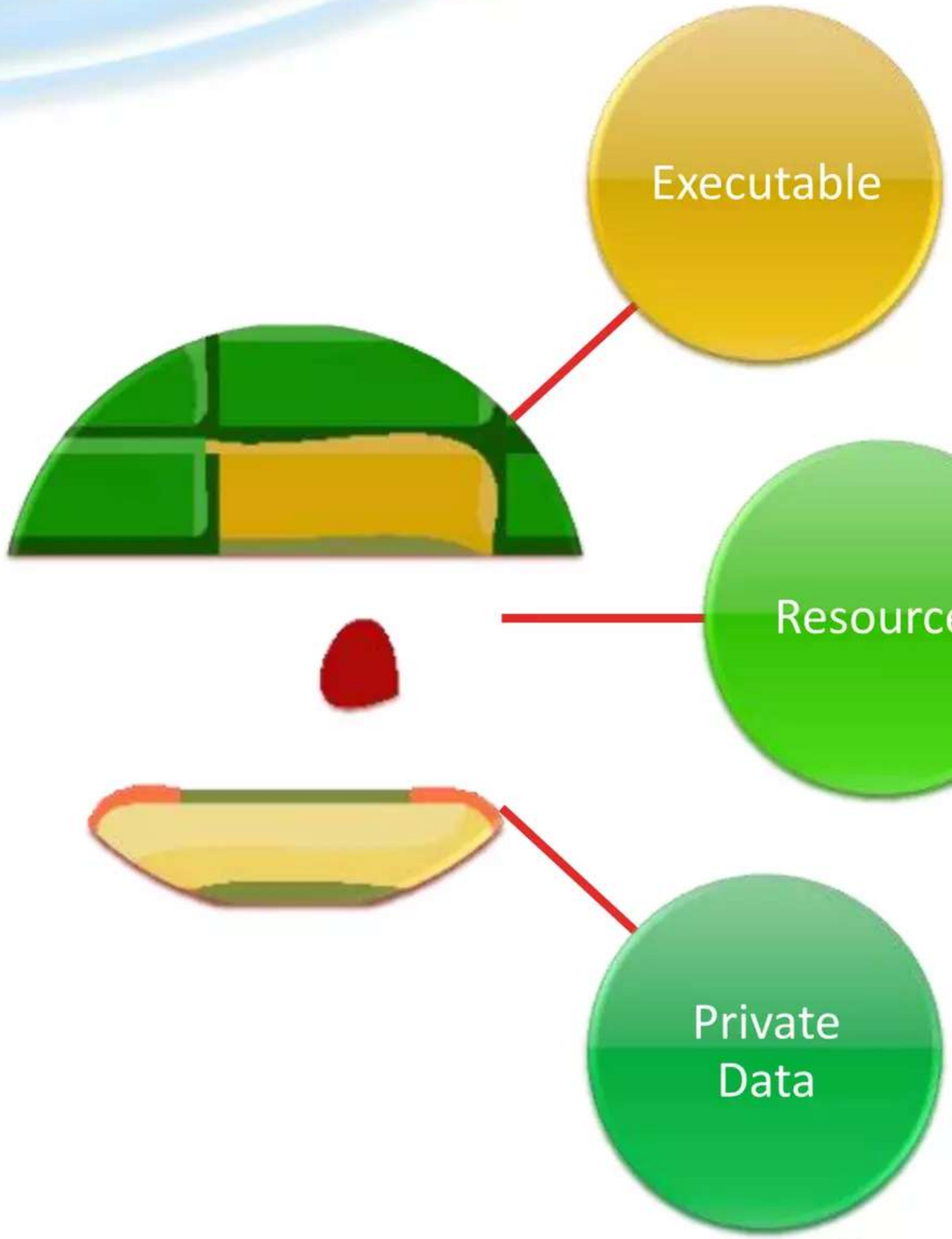
- Possible to develop **without capability restrictions** in the emulator
- **Capability checking** can be **(de)activated** in the emulator options
  - Issues a warning if a required capability is not defined



Preserve security of important files

Data Caging

Data Caging – Overview



- All binaries of all apps in one dir
- Can only be executed from there
- Usually: no read / write access
- \sys\bin\

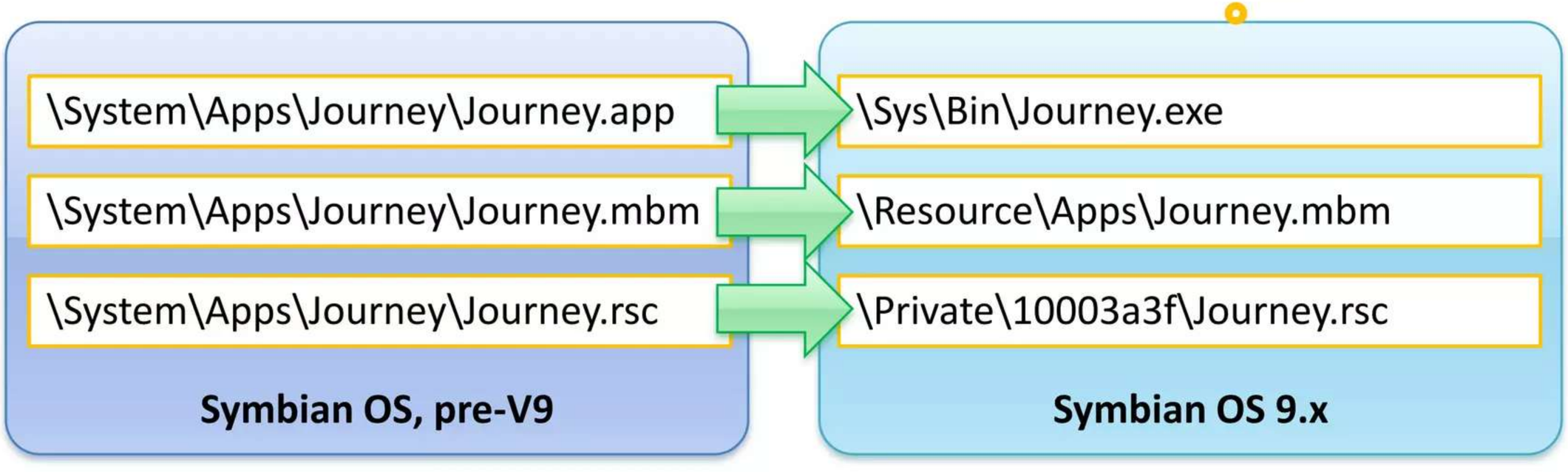
- Bitmaps, fonts, help files
- Usually: read-only
- \resource\

- Each application its own dir
- Used for settings, ...
- Usually: read / write only for own private dir
- \private\<UID3>\

Data Caging

- **App. only has access to:**
  - Own directories
  - "Open" directories
- Access based on capabilities and identity

Separating code and data!



Data Caging – Directories

Directory	Capability (Read)	Capability (Write)	Used for
\sys	AllFiles	TCB	Applications can only be executed from here.
\resource	-	TCB	Fonts, bitmaps, help-files, … Only read-access for apps, to prevent corruption.
\private\<own SID>	-	-	For application data (settings, …). DLLs do not have own private dir, but instead use that of their loading process.
\private\<other SID>	AllFiles	AllFiles	AllFiles-capability required to access private data of other apps.
All others	-	-	Unrestricted data – images, …

More about directories

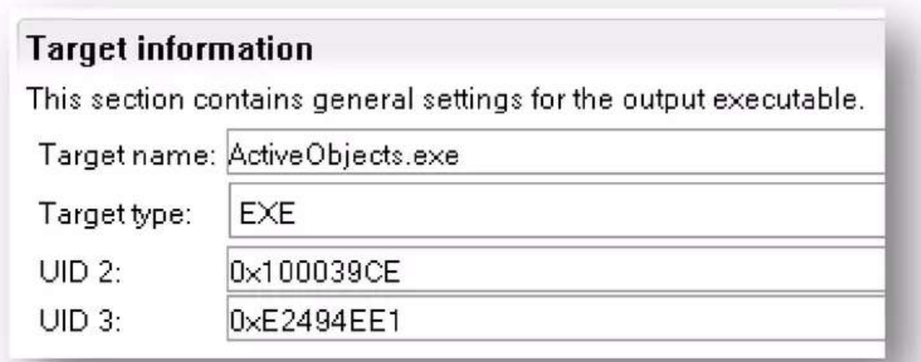
- Data caging provides **secure area for application's data**
- **All executables installed to \sys\bin**
  - Risk of **filename clashes** – use your unique UID as part of the executable filename
  - **Removable drives:**
    Hash stored to c:\sys\hash
    Prevents execution of modified executables

32 bit-numbers that identify your executable:

Identifiers

Unique Identifier (UID)

- Uniquely identify binary file
- Built into first 12 bytes of any Symbian OS file
- **UID 1 (Target type)**
  - Application type (exe for OS v9+, dll <=OS v8)
- **UID 2**
  - Subdivides certain target types (static / polymorphic DLLs)
- **UID 3**
  - Unique identification for binary.
  - UID requested and assigned through Symbian for commercial applications (get one at www.symbiansigned.com)



MMP File Editor in Carbide.c++

UID3 Ranges

- Error message when installing on the device
  - **Use correct UID3** (changing UID: Search/Replace in *.* + subdirectories with a text editor)

	UID3	Range	Purpose
Protected (Symbian Signed)		0x00000000 – 0x0FFFFFFF	Development use (pre-v9)
		0x10000000 – 0x1FFFFFFF	Legacy UID (pre-v9)
		0x20000000 – 0x2FFFFFFF	v9 protected UIDs
Unprotected (unsigned)		0xA0000000 – 0xAFFFFFFF	v9 unprotected UIDs
		0xE0000000 – 0xEFFFFFFF	Development use (v9)
		0xF0000000 – 0xFFFFFFFF	Legacy UID (pre-v9)

Secure Identifier (SID)

- **Secure ID (SID)** = Unique identifier for each executable
- Locally unique (on the device)
- **Used for:**
  - Access to which private directory (\private\<SID>\)
  - Identification for IPC (Inter Process Communication)
- **Default: Same as UID3 (recommended!)**

Vendor Identifier (VID)

- **Vendor Identifier (VID)** = Unique identifier for software vendor
- Globally unique through Symbian Signed
- Unsigned apps: no VID
- **Used for:**
  - Limit access, e.g. for internal APIs, only accessible for Nokia
  - IPC
- **Note:** SID / VID not relevant for DLLs, execute within process of .exe and use their SID / VID

Testing applications:

Symbian Signed

Symbian Signed

- **Self-Signed application:**
  - Security warning during installation
  - Only access to user capabilities
- **Reasons for signing:**
  - **Prevent sabotage** of installation files (.sis)
  - **Identification** of the software developer
  - Extended access to APIs (**Capabilities**)
  - Get rid of warning during installation
- **Signing through:**
  - Independent Test Houses (traditionally)
  - Includes test of the application

Overview

Capabilities	User Grantable	Open Signed w/o Publisher ID	Open Signed with Publisher ID	Express Signed	Certified Signed	Symbian Signed for Nokia / …
LocalServices	For testing & sales version	For testing	For testing	Sales version	Sales version	Sales version
ReadUserData						
WriteUserData						
NetworkServices						
UserEnvironment						
Location						
SwEvent						
ProtServ						
TrustedUI						
PowerMgmt						
SurroundingsDD						
ReadDeviceData						
WriteDeviceData						
CommDD						
DiskAdmin						
MultimediaDD						
NetworkControl						
AllFiles			Device Manufacturer approval			
DRM						
TCB						
Lead-time	Immediate	Immediate	Immediate	Immediate	1 week	1 week
Note	Developer tested	Upload SIS	Certify on PC	Developer tested	Test house tested	Test house tested
Price	100% free	100% free	Certificate, no testing costs	Certificate, small testing costs	Certificate + testing costs	Certificate + testing costs

Signing Programs

- New signing process starting with Q4 / 2007:

	Publisher ID required	Independent Testing required	IMEI restrictions
Open Signed	not required if signing online	no	yes
Express Signed	yes	no	no
Certified Signed	yes	yes	no

Self Signed

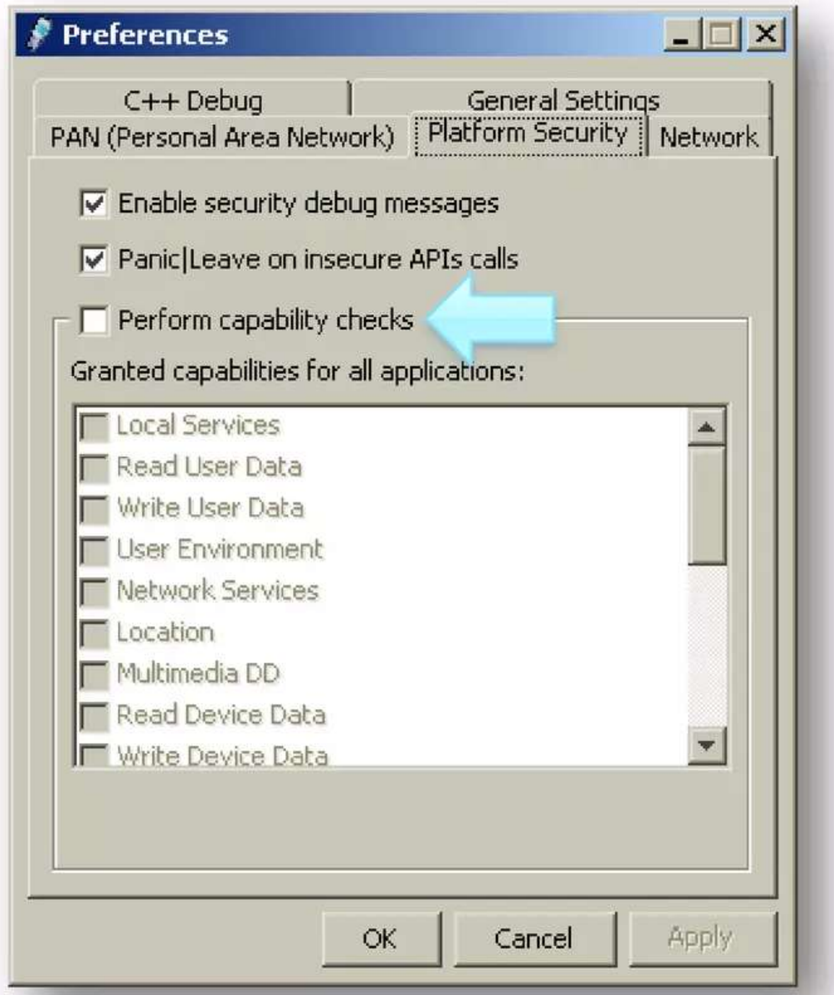
- Can be used for testing and distribution
  - … if no or only user-grantable capabilities are required

Open Signed

- **Certification for testing (not for sale!)**
- **Without Publisher ID:**
  - Upload application to website
  - URL to signed app sent per email
  - Restricted to one IMEI
  - Completely free, valid for 36 months
- **With Publisher ID:**
  - Signing works offline through Developer Certificate
  - Valid for 36 months
  - < 1000 IMEIs

Open Signed – Developer Certificates

- **Requires Publisher ID**
- Allows offline signing (open signed) for accessing (nearly) all capabilities
  - Certificate limited to < 1000 devices
- Valid for 36 months (to prevent distribution)
- Only for devices – development using the emulator possible without certificates!
- **Request:**
  Through DevCertRequest-Tool from
  www.symbiansigned.com



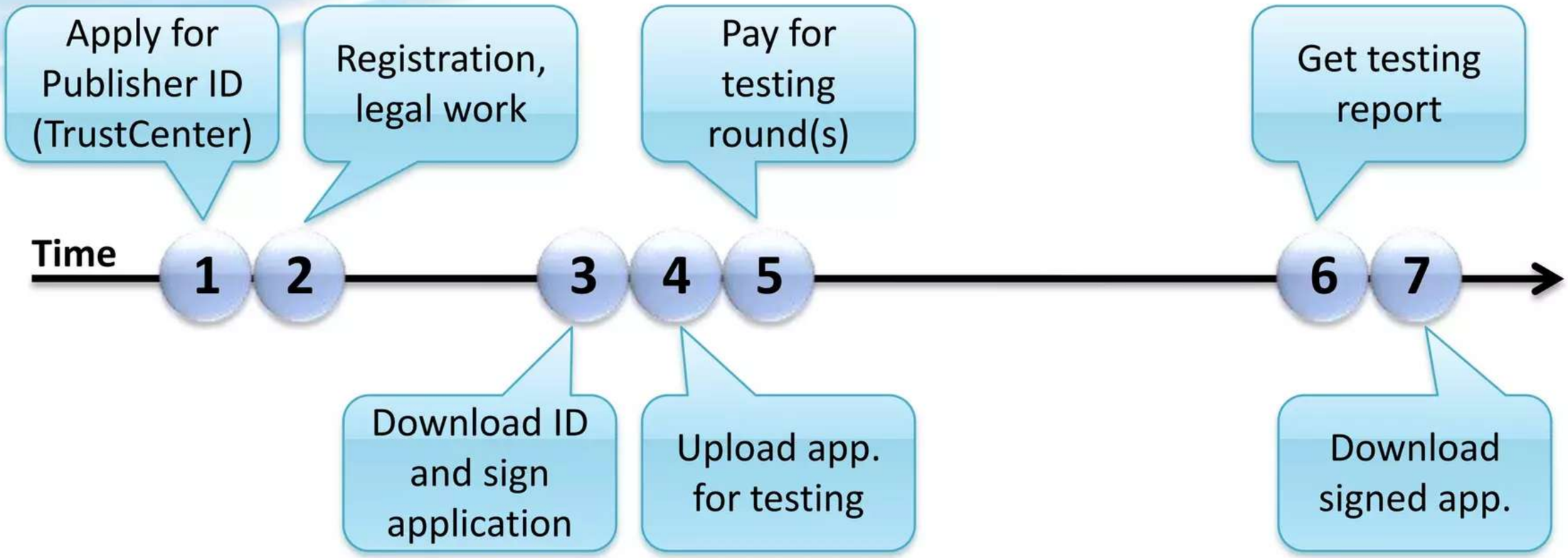
Express Signed

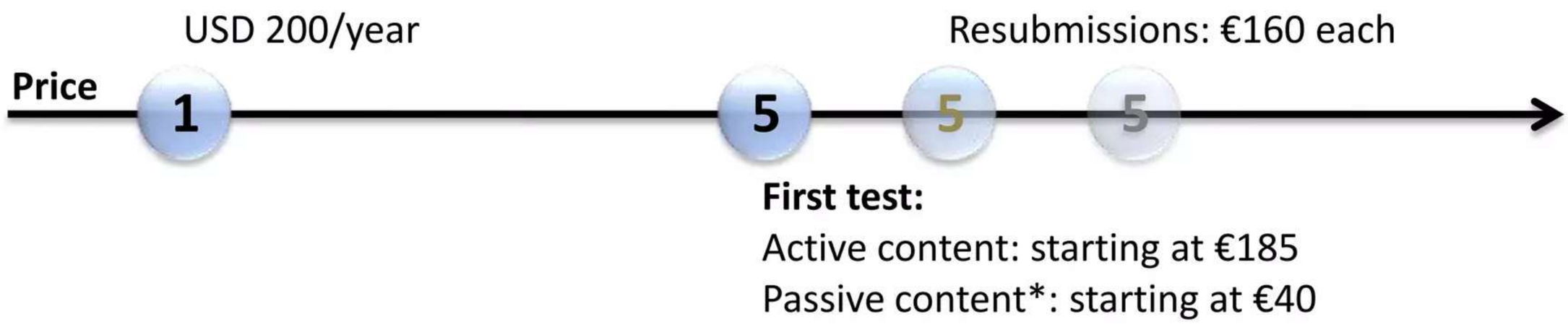
- **Certification for sale**
- **No security warnings**
- **Developer tests** the application
  - … but some apps may be tested and **results audited**
- Valid for 10 years
- **Costs**
  - Publisher ID (USD 200 / year), but also possible through publishing partners
  - 20 USD / submission

Certified Signed

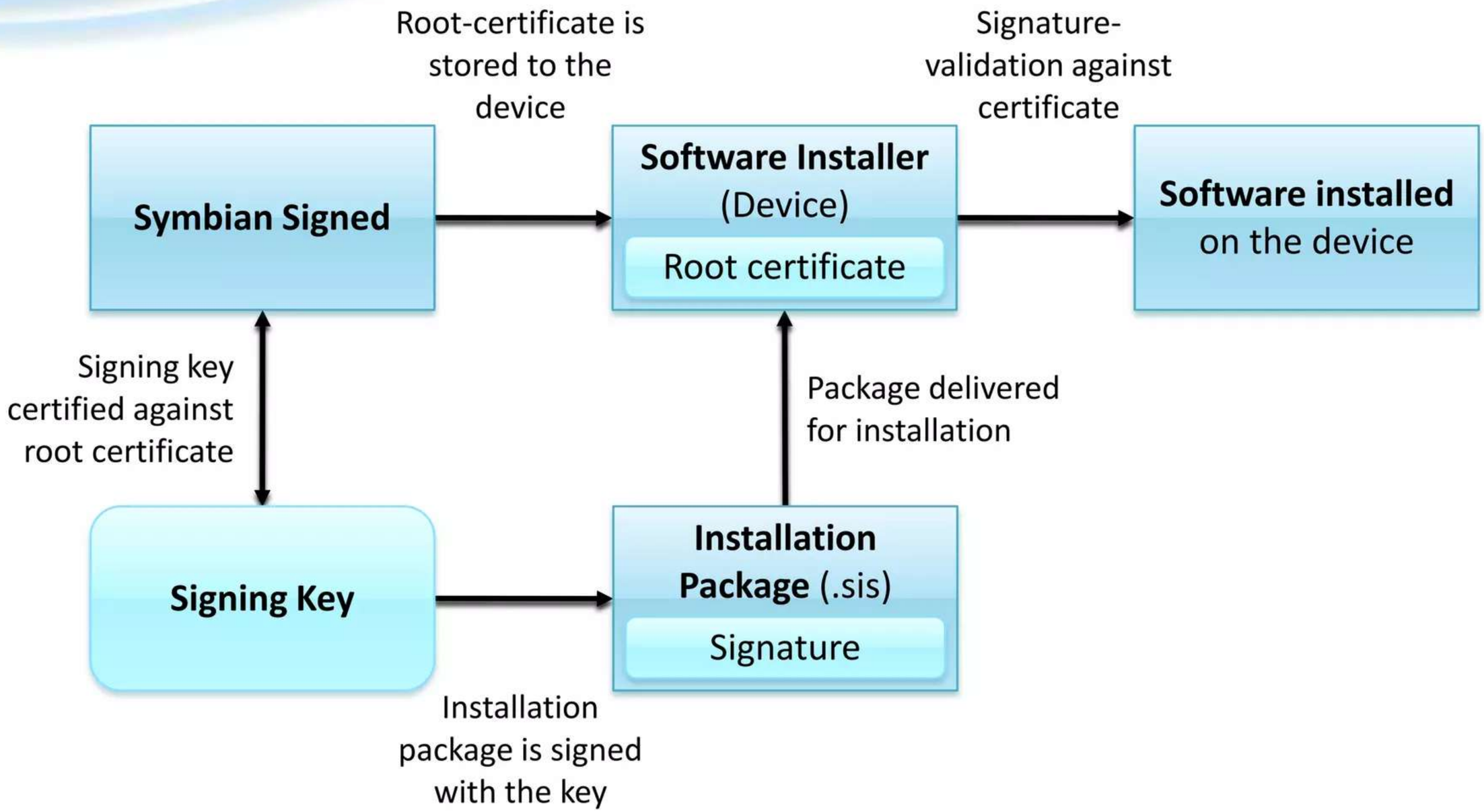
- **Most trusted option**
- Tests done by **independent test house**
  - Quite expensive
  - Takes about one week
  - Faster signing available, but even more expensive
- Certified after passing tests

Certified Signed





Software Installation



Application Tests

- Some examples:
  - **Friendly to the system?** (Using system features like calendar or making a call while app. is running)
  - **Stress tests** (Start camera, fast key presses, remove MMC or battery, …)
  - **Low memory** during start-up or while application is running
  - **De-Installation** removes all files?
  - …
- https://www.symbiansigned.com/app/page/overview/testcriteria

Certificate Error Messages

- Overview about errors and possible solutions:
  - http://blogs.forum.nokia.com/view_entry.html?id=93

Did you understand everything?

Test your Knowledge

Quiz

- **Situation:**
  - A Bluetooth-game (game.exe, Capability: LocalServices) loads and uses an engine-DLL (gameEngine.dll, without Capabilities), which only handles game statistics.

- **Question:**
  - Can game.exe load gameEngine.dll?

Quiz

- **Answer:**
  - No, the engine can not be loaded
- **Reason:**
  - App. can’t load DLL with less capabilities than itself
  - DLL runs with same Capabilities as the app.!
  - Capabilities set for the whole process

ASD-like Question – Easy

- **Which of the following statements about Symbian OS capabilities are incorrect?**
  - **A.** The capabilities of executable code are specified using the *CAPABILITY* keyword in the MMP file.
  - **B.** The following specification in an MMP file grants privilege to access the user's files stored anywhere on the phone or removable media:
    CAPABILITY ReadUserData
  - **C.** The following specification in an MMP file allows the code to power down the phone:
    CAPABILITY PowerMgmt
  - **D.** The capabilities of an application can be boosted by calling User::SetCapability().
  - **E.** The following statement in an MMP file will grant the binary *SwEvent* capability in emulator and hardware builds:
    PlatSecDisabledCaps SwEvent

Solution

- **A.** Correct.
- **B.** Incorrect. The “ReadUserData”-capability only allows accessing private data of the user (like calendar, contacts), but not to all his files.
- **C.** Correct.
- **D.** Incorrect. The capabilites of a process never change during its lifetime.
- **E.** Incorrect. Parts of the platform security can only be disabled in the emulator, not on the device.

ASD-like Question – Medium

- **Which of the following statements about data-caging on Symbian OS are correct?**
  - **A.** Executable code can be installed into and executed from any subdirectory of \system
  - **B.** The \resource directory can be used to store writable configuration files.
  - **C.** A DLL with a Secure Identifier of 0x20005268 owns a private data-caged directory called \private\20005268.
  - **D.** A private data-caged directory can only be accessed by the owning process and other processes with *AllFiles* capability
  - **E.** Symbian OS provides a special directory on removable media which is used to detect whether executable code, installed to the card, has been tampered with.

Solution

- **A.** Incorrect. The directory which allows executing binaries is called \sys instead of \system starting with Symbian OS 9.
- **B.** Incorrect. The private directory can be used to store configuration files. For normal apps without the TCB-capability, the resource-directory is read-only.
- **C.** Incorrect. DLLs do not have their own private directory. They use the same as their owning process.
- **D.** Correct.
- **E.** Incorrect. Storing this information on the removable media would not make sense. The correct directory is: c:\sys\hash

ASD-like Question

- **Which of the following statements about the groups of Symbian OS capabilities are incorrect?**
  - **A.** User capabilities are those capabilities which the user may grant at installation time, if a SIS file is not itself already signed for those capabilities.
  - **B.** The user can decide whether to install code that reveals their location.
  - **C.** The user can decide whether to install code that accesses and modifies the system settings of their phone.
  - **D.** Installable software that needs system capabilities must be certified by a trusted body, such as a test house operating on behalf of Symbian Signed, before it can be installed and tested by the developer.
  - **E.** Installable software that only needs user capabilities (or no capabilities at all) does not need to be certified by a trusted body such as Symbian Signed before it can be installed and tested.

Solution

- **A.** Correct.
- **B.** Correct.
- **C.** Incorrect. The user can only take decisions related to himself, but not to the phone integrity. Modifying system settings is not user-grantable and requires Symbian Signed.
- **D.** Incorrect. The developer can get a “developer certificate” to test an application on devices prior to sending it in for Symbian Signed-testing.
- **E.** Correct.

That’s it!

Thanks for your attention