John Pagonis

Symbian OS v9.x
Introduction to Platform Security

What is Symbian OS v9.x Platform Security ?

It is a fine grained way to efficiently restrict or completely prevent unauthorised access to sensitive APIs and data on the mobile phone while keeping the device open to developers

- ✓ It follows a per-process capability-based model
- ✓ It compartmentalises the system according to access capabilities to APIs and files
- ✓ It makes sure that the users can make policy decisions they understand
- ✓ It is Kernel mediated but server enforced

Why do this ?

- Why introduce a finer-grained, Platform Security model?
  - … Phones are open, networked & data communication devices
  - … Users expect their phones to be highly reliable
  - … Users care about their privacy – and their phone bills
  - … Mobile networks are not like the internet – they can restrict access
  - … "Perimeter Security" model enables unrestricted access to all phone capabilities once installed

Platform Security – user centric view

Plat Sec means for users that:

- They have
  - … No unexpected items in their phone bill
  - … Their phone working when needed
  - … No virus
  - … Their private data staying private
- They do not have
  - … To take security decisions they do not understand
  - … To take security decisions too often

Scope

- Includes
  - … Symbian OS & device drivers
  - … User interface
  - … Applications
- Excludes
  - … Hardware
  - … Network infrastructure
  - … Remote servers

When we talk about Platform Security…

- It is about
  - … Protecting phone integrity
  - … Protecting sensitive data
  - … Controlling access to sensitive operations
- It is not about
  - … Encrypting data
  - … Securing network protocols
  - … Scanning for viruses
  - … Managing public key infrastructure

Benefits

- For developers
  - Maintains network operator & user confidence in open phone environment
  - Grows opportunity for mass market applications, content & services
  - enables m-commerce applications & high value DRM content
- For network operators
  - Protects network & handsets from malware
  - Protects customer data & privacy

Impact for Developers

Don’t Panic !

☺

New Symbian OS Concept – Capabilities

- Every executable is tagged at build time with some capabilities, this applies for both EXEs and DLLs
- At run time, every process has a set of capabilities
- Capabilities of a process never change
- Capabilities are assigned based on which APIs a process needs and therefore is authorised to use
- Capabilities and policing of, is transparent to API users

New Symbian OS concept – Data Caging

- Separating code from data (API vs FS)
- File-system structure changes
  - … \sys, \resource, \private\<process specific>, \<other>
  - … Executables will be placed and only run from \sys\bin
- Processes are confined to their own part of the file-system
- Access rules based on directory path
  - … Single user, no access control list required
  - … No extra storage needed
- Support for removable media file systems
  - … tamper evidence for binaries

New Symbian OS Concept - Process Identification

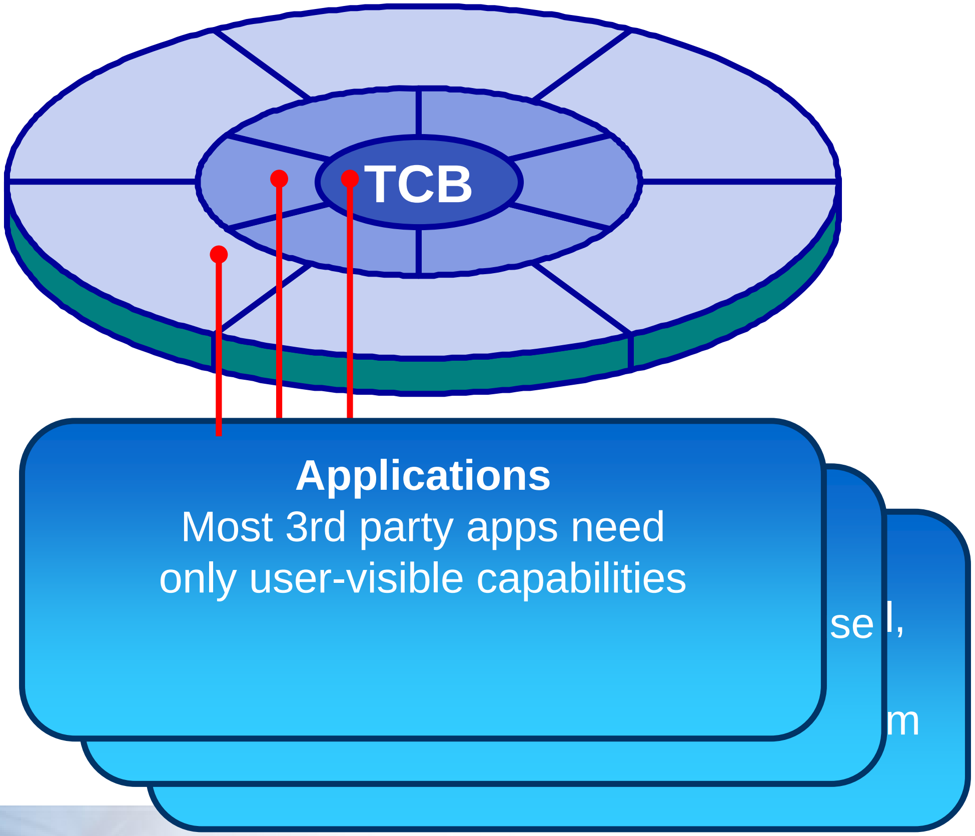
- Each executable now contains a Secure ID (SID)
- Secure IDs are guaranteed to be locally unique
  - … Hence \private\<Secure_ID>\
- SIDs will come from the upper part of the UID range
- SID is specified by the SECUREID keyword in an .mmp file
  - … If not given UID3 is used, otherwise KNullID
- Each executable now can contain a Vendor ID (VID)
- VIDs allow for unique identification of vendors
- VID is specified by the VENDORID keyword in an .mmp file

New Symbian OS concept - Trusted Computing

- Trusted Computing Base (TCB) → access all areas
  - New Kernel, EKA2
    - New Inter-Process communication protocol
    - New kernel memory model
  - New Software Install
    - Better rollback of interrupted or failed installation
    - Verification of application’s access rights at install-time
  - File server & Loader
    - New file access control
    - New loading rules
- Trusted Computing Environment (TCE)
  - All important system servers (e.g, ETel, ESock, WServ etc)

Capabilities Model enables Compartmentalisation

- Based on their assigned capabilities, processes may access API calls over IPC or by DLL loading
- System servers will need to police such calls and grant access to callers
- The kernel passes ,like a token, to servers the capabilities of calling processes on each IPC
- The file server will police access to parts of the file-system based on the capabilities and identity of the caller process.



Capabilities categorisation

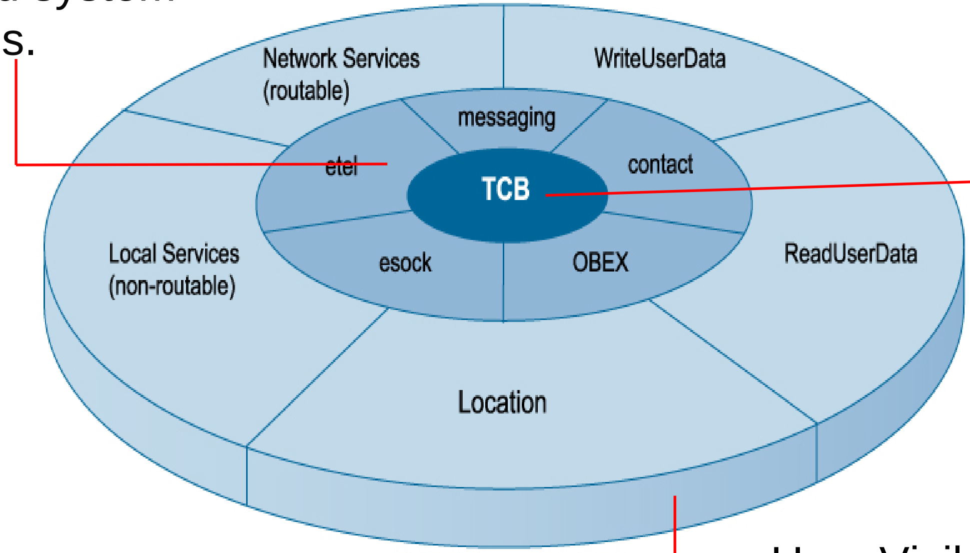
- Full file system privilege
  - … Reserved for Trusted Computing Base
- System privileges
  - … Reserved for the Trusted Computing Environment
  - … Coarse-grained capabilities: CommDD, MultimediaDD, NetworkControl, DRM, DiskAdmin etc
- User privileges
  - … NetworkServices, LocalServices
  - … ReadUserData, WriteUserData
  - … Location, UserEnvironment
- According to capabilities, service access is policed by the next level service providers

TCB→ TCE→ rest

Capabilities & Trusted Computing Platform

Trusted Computing Environment System servers: Run at different restricted system privileges.

Trusted Computing Base: Runs at full file system - permission to modify executables.



User Visible Range: User can grant these capabilities at install time OR applications can be signed for them.

How to assign capabilities to binaries

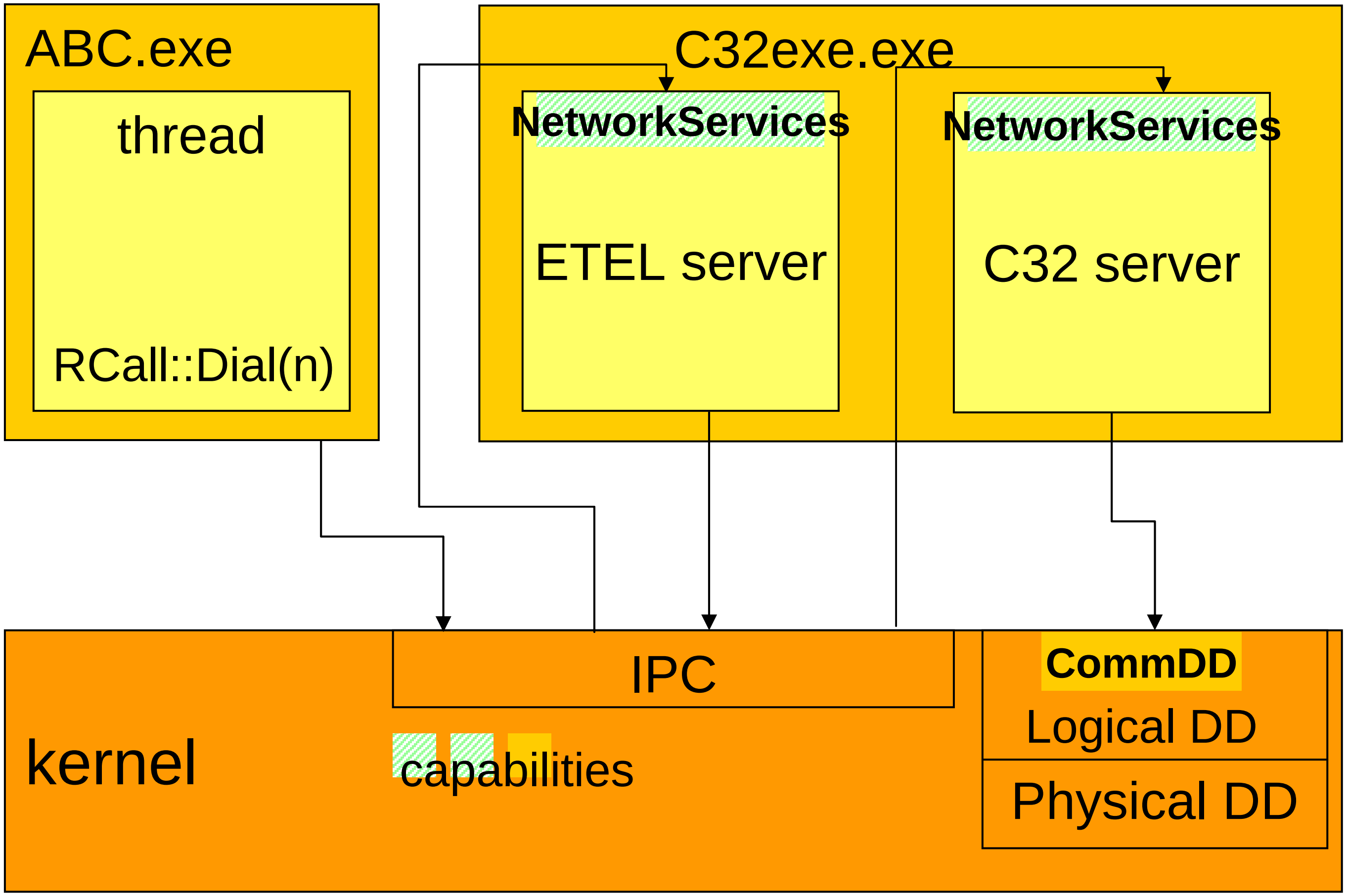
- Capabilities are stored in executables
  - They are part of the EKA2 executable file format
- Capabilities are defined in mmp files

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

Capabilities at load time

- Rule 1:The capabilities of a process never change
  - No way to add or remove capabilities to a process
  - Loading a DLL never change the process capabilities
  - DLL code runs at process capabilities level
- Rule 2: A process cannot load a DLL with less capabilities than itself
  - DLL capabilities do *only* reflect a level of trust
  - DLL capabilities do not authorise anything

How do capabilities work at run-time?



They are worth checking only when a process boundary may be crossed

Data caging directory access rules

- \sys
  - … Read/Write access reserved to TCB
  - … All binaries under \sys\bin
- \resource
  - … Read access for all, Write access for TCB
  - … Used for storing fonts, bitmaps, help files…
- \private\<process_SecureId>\
  - … One private space per process
  - … Process_SecureId == EXE's 3rd UID
  - … Read/Write access reserved to process owner & TCB
- \<others>
  - … Read/Write access for all

So what if you want to share ?

- Publish & Subscribe
  - … New EKA2 IPC allows publisher to specify subscriber capabilities, SIDs or VIDs
- Central repository
  - … Service for sharing persistent settings
- DBMS
  - … Service for sharing relational databases
- Shared file handle between processes
  - … New EKA2 – Symbian OS v9.x feature

What happens to applications then ?

- ABC.app becomes ABC.exe
  - To assign ABC.exe the capabilities it needs
  - To protect ABC’s private data
  - Only a few code lines to change
- Application files need to be relocated

\System\Apps\ABC\ABC.app	\Sys\Bin\ABC.exe
\System\Apps\ABC\ABC.mbm	\Resource\Apps\LocalisableFiles\ABC.mbm
\System\Apps\ABC\ABC.rsc	\Resource\Apps\UIResourceFiles\ABC.rsc

What about polymorphic interface DLLs ?

- Plug-in DLLs limited to what the host process can do
  - … Implementers do not have to implement capability checking
- Plug-in DLLs as trusted as the host process
  - … Recognisers, same trust level as Apparc server, MTMs same trust level as Messaging server

What about static interface DLLs

- Shared libraries that export a static interface will need to have capabilities such that all its users may load them
- This means that even a simple DLL that does for example some signal processing calculations will need to have capabilities such that a telephony application may use it.
- A DLL that is loaded by another DLL will need to have the same or greater capabilities as the calling executable

..and what about servers ?

- Servers will need to police access to their resources accordingly (use of CPolicyServer)
- Policing must occur at IPC boundaries
- Servers which are trusted by the TCE and others, should be careful not to ‘leak’ such trust

A .pkg example

```
;*Languages
&EN
;This section specifies the package name, UID, and version/build numbers. Add the package TYPE here if needed.
#{"voice"},(0x2000521D),1,0,0;
;
;*SDK Compatibility Product UID/Platform Identification should specify the highest SDK version your application will support.
;Series 60 v3.0
[0x101F7961], 0,0,0, {"Series60v30ProductID"}

;*Unique (Non-Localised) Vendor name, used in combination with signing to prevent the unauthroized upgrade of a package by someone other than the rightful vendor.
:“Symbian"
;*Files To Copy...<src> <destination>
;The destination files should be a full path. If you use a '!' character
"O:\Symbian\9.1\S603MR\S60_3rd_MR\Epoc32\release\GCCE\UDEB\voice.exe" -"!:\sys\bin\voice.exe"
"O:\Symbian\9.1\S603MR\S60_3rd_MR\Epoc32\data\Z\private\10003a3f\apps\voice_reg.R01" -"!:\private\10003a3f\import\apps\voice_reg.rsc"
"O:\Symbian\9.1\S603MR\S60_3rd_MR\Epoc32\data\Z\resource\apps\voice.R01" -"!:\resource\apps\voice.rsc"
"O:\Symbian\9.1\S603MR\S60_3rd_MR\Epoc32\data\Z\resource\apps\voice_loc.R01" -"!:\resource\apps\voice_loc.rsc"
```