Maemo 6 Security Framework

Dmitry Kasatkin, Maemo Security team

FRUCT 7, Saint-Petersburg, Russia

30.04.2010

Agenda

- Why security
- Linux security models
- Chipset security & boot process
- Integrity Protection
- Access Control
- Privacy Protection
- Q&A

Why? – Reasons for Security

- To protect the user
  - Data from being stolen or misused
    - File encryption
    - Device lock
    - Device wipe
  - Unexpected costs
    - Against mallware sending sms to pay numbers
- Device protection - liability & product safety
  - To satisfy specification requirement
    - IMEI (3GPP), Serial numbers (BT, MAC address etc.) protection, WIMAX credentials
  - To protect device integrity & identity
    - RF, EM and WiFi tuning values – if malicious code changes EM values, batteries starts to explode
    - Product variant specific settings – e.g. supported RF bands. We can sell 3G variant at higher price than 2G and use the same HW

Why? – Reasons for Security

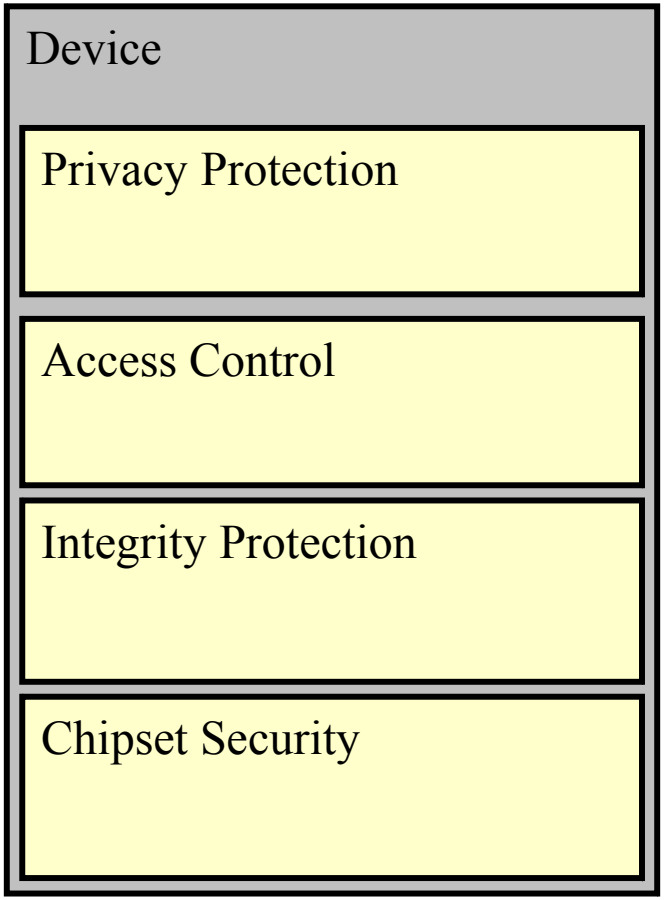
- Nokia business protection and empowerment
  - Compliancy to operator requirements
    - SIM locking / Subsidy locking – Nokia loses business immediately when Simlock is broken
    - Compliancy to existing operator solution. In practice: there is already existing tools & smartcards @ operator side and new solutions needs be compliant against these
  - Product variation and variant identity protection
    - AT&T variant need to stay AT&T variant
  - To reduce fraud against Nokia business
    - False service bills, Device cloning, back-door manufacturing
  - To enable Nokia service business
    - DRM – enable e.g. ComesWithMusic && App store
    - Empowerment of future service business like mobile payments and billing

Linux Security Models

- Classical UNIX DAC
  - Multi user model
  - POSIX capabilities are not really in use – root does everything
  - No process based access control
- SELinux
  - Domain Type Enforcement (DTE)
  - Labels
  - Too complex administration for mobile use
- Tomoyo/AppArmor
  - Path based
  - No APIs
- Smack
  - Simple MAC
  - Administration overhead not suitable for mobile use

Maemo Security Framework – multilevel model

- Set of mechanism to protect entire platform
- Chipset Security
  - secure cryptographic services for OS
- Integrity protection
  - integrity protection of TCB, applications and data
- Access Control
  - limit application access to critical resources if used
- Application privacy protection
  - integrity/confidentiality services for application



Chipset Security

Chipset Security – HW enabler

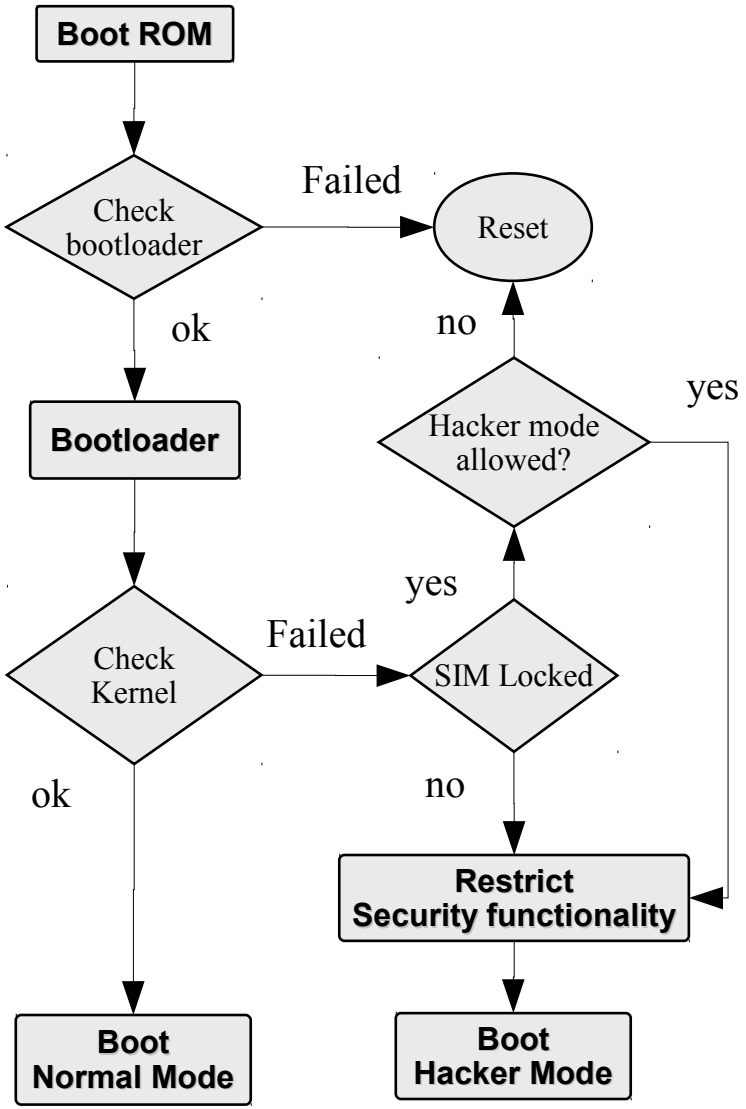
- Chipset security is the key component whole platform security relies on
- Provides key management and secure cryptography services
- Provides secure boot
- Provides device identity
  - Unique PublicID
    - Certain data can be bound to device using PublicID
  - Unique device private and public keys
    - Used to encrypt and decrypt device dependent data
  - Chipset specific Root Public Key
    - Used to verify signature of Public Key Certificate (PKC)
- Provides Secure Execution Environment (SEE)
  - Secure ROM/RAM
  - Allows to execute authenticated “protected” applications PA

Maemo 6 Device operation modes

- Normal Mode
  - Default operation mode
  - Controlled access to critical resources
  - Applications can get authorized access to resources
  - Device Keys are available to decrypt sensitive data
    - Access to OVI services, Games, etc...
    - Copy Protection can be enabled
- Hacker mode
  - Open Source friendly
  - Compile and flash your own kernel
  - Developer can full access to platform resources
    - Low level development
  - However some functionality disabled
    - Chipset Security limits access to device keys
    - E. g. DRM keys are not available. DRM content cannot be decrypted

Boot Process

- Boot ROM verifies Public Key Certificate (PKC) using Root Public Key
- Boot ROM verifies Bootloader image using OEM public key from PKC
- Bootloader verifies kernel image using OEM public key



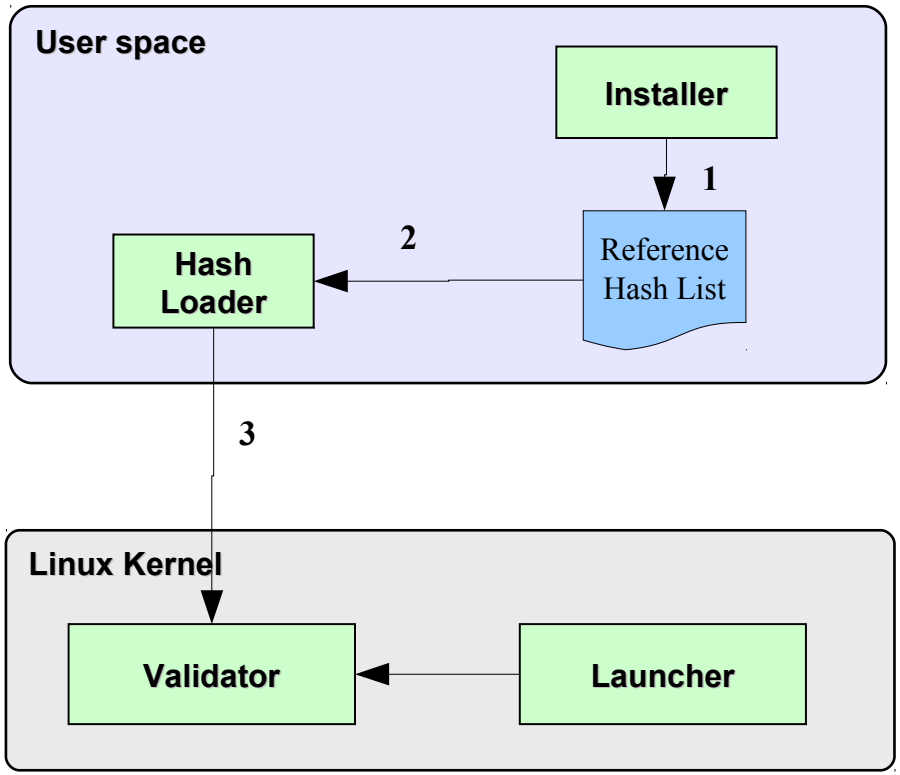
Integrity protection

Integrity protection – Maemo Validator

- Protects integrity of TCB components
  - kernel modules, sensitive user level components (Installer, D-BUS daemon)
- Insures integrity of executables such as kernel modules, binaries and libraries
  - Corresponding SHA1 hash is stored in the reference hash list
  - Hash list is stored in Maemo Protected Storage
- Validator is a LSM kernel module
  - Loads reference hash list on startup
  - Calculates SHA1 hash of executables
  - Compares to the one stored in the hash list
  - Cache is used for optimization
- Application Manager updates hash list when new packages installed or removed
- Integrity protection policy
  - Defines an action in case the integrity check fails. Currently it will block the execution

Maemo Validator components

- Hash database
- Installer
  - Updates hash db
- Hash loader
  - Loads into the kernel
- Validator
  - LSM module
  - Called “by” launcher



Access Control

NOKIA

Access Control – principles & concepts

- Maemo Access Control Principles
  - Principal of least privileges
    - Application should be allowed to access only needed resources
  - Minimal changes to the Linux default model
- Protected Resources
  - Virtual objects which represents some functionality
- Resource Tokens
  - String naming a protected resource
  - Cellular, UserData, etc.
- Application needs to declare required or provided resources
  - Maemo Manifest File

Credentials

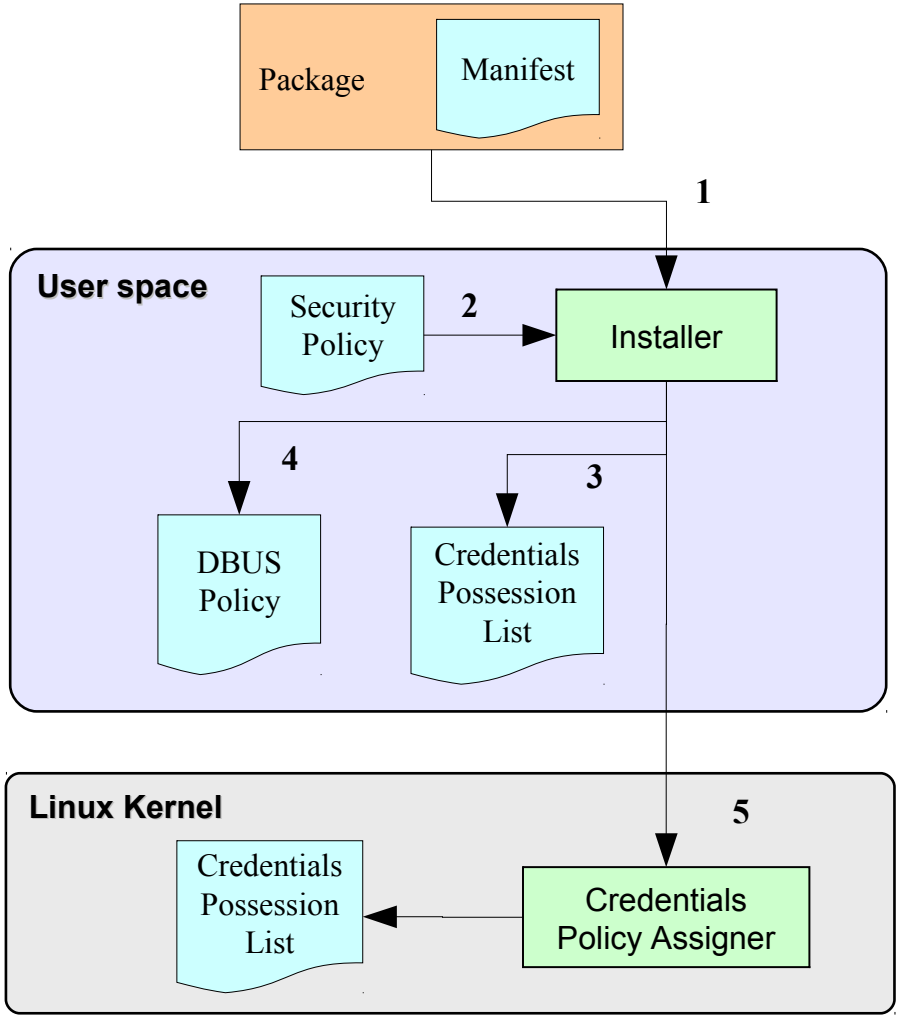
- Credentials
  - Access Control is based on credentials
  - POSIX capabilities
    - CAP::cap_sys_rawio
  - User/Group
    - UID::mysql, GID::admin
  - Resource tokens
    - Global: user-data, telephony
    - Package specific: my-package::access
    - Resource token is a new concept in Maemo security model
- When application starts, it gets credentials assigned
- Service is able to verify possession of the token at runtime

Application identity

- Requirements
  - should be impossible to forge it
  - should be unique (identify an application uniquely)
  - should remain the same for the application updates, between boot-ups, for the different instances of the same application, and after the application restart
- Application identity defined as
  - AppID = {SourceID, Package, Application Name}
    - AppID = {ovi.com, CoolTools, AddressBookPlugIn}
  - Application name is given in Manifest file
    - Optional

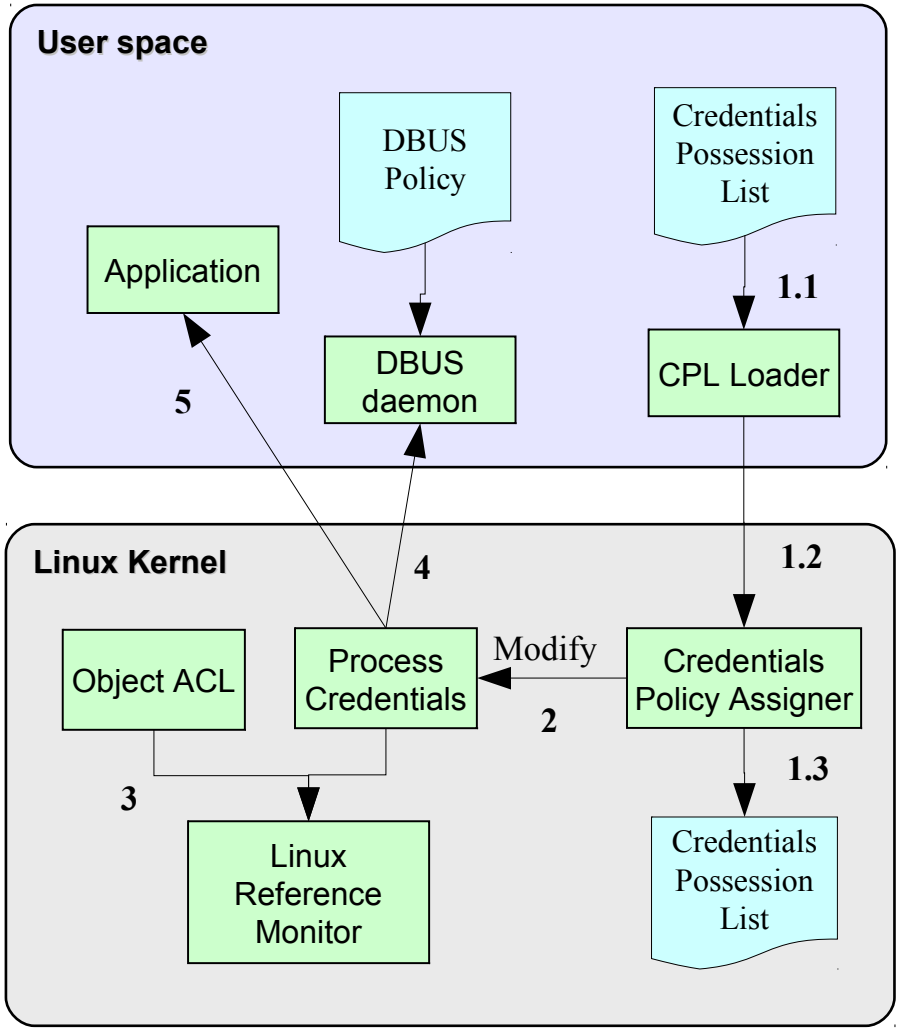
Access Control components - Installation

1. Application arrives to the Aegis Installer together with Aegis Manifest
2. Aegis installer checks the Aegis Security policy for the information
3. Aegis installer modifies the Credentials' possession list according to the "Intersection rule"
4. Aegis installer possibly modifies D-Bus policy
5. Update runtime policy



Access Control components – Boot and Launch

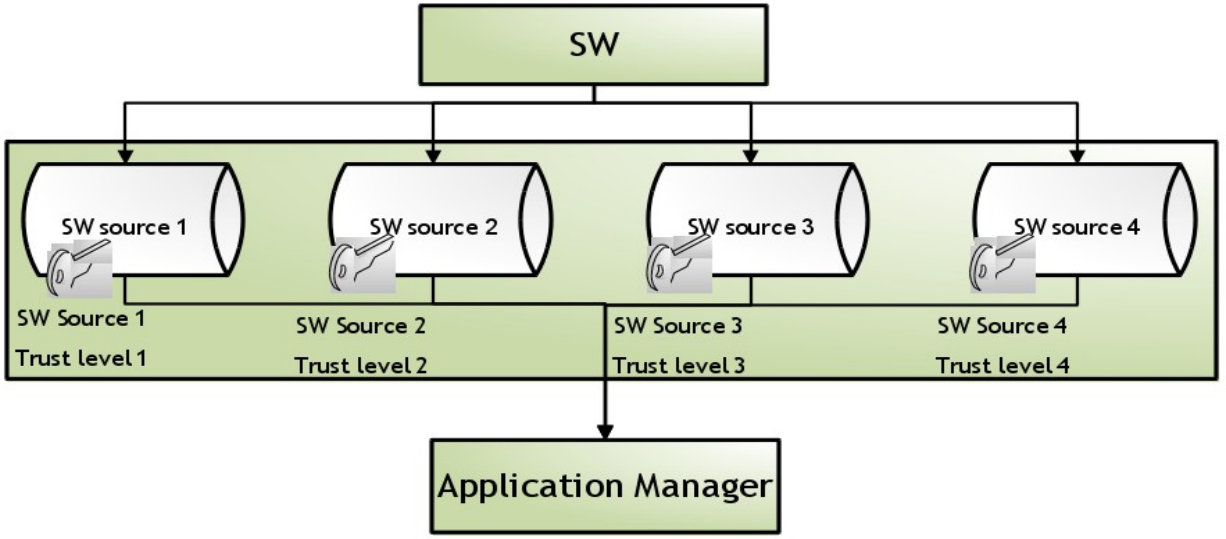
1. Process Credentials Assigner gets the allowed credentials set from the Credentials' possession list
2. Process Credentials Assigner modifies process' credentials (process task structure) according to the received credentials
3. File AC
   - No change
4. D-Bus
   - Check client credentials using libcreds
5. Any Application
   - Checks using libcreds by itself



Maemo Security Policy

- Contains mapping between SW sources and allowed credentials
- Each SW source has an entry in device policy file
  - {SourceID : Trust Level : Public Key : Allowed credentials}
    - {maemo.org : 1 : ABCDEF : user-data, cellular}
  - SourceID: e.g domain name
  - Assigned Trust level
  - Provides list of allowed credentials
  - Public key to verify package signature

- Package from higher trust level cannot be replaced by from lower level
- Package manager verifies if all credentials from Manifest file can be granted



Maemo Manifest File

- Needed for applications requesting or providing resource tokens
- Included in to the package as **pkg-src/debian/<pkgname>.creds**
- Used by package installer to update device policy database

Maemo Manifest syntax

- XML document
- General tags:
  - requested credentials: <request>
  - provided credentials: <provide>
  - credential name: <credential name="credential name">
  - absolute path to the program executable: <for path="path">
- D-Bus service tags:
  - D-bus service name: <dbus name="dbus service name">
  - D-bus type (system or session): <bus="bus type">
  - Credential to bind to a specific d-bus service name <own="credential name">
  - Example:
    - <dbus name="com.nokia.devicelock" own="devicelock" bus="system">
  - D-Bus interface name: <interface name="interface name">

Examples - manifest

- Server Manifest file

```
<aegis>
  <provide>
    <credential name="user-data" />
  </provide>
</aegis>
```

- Client manifest file

```
<aegis>
  <request>
    <credential name="user-data" />
    <for path="/usr/bin/userdatamanager" id="client"/>
    <for path="/usr/bin/userdataclient" id="client"/>
  </request>
</aegis>
```

-

NOKIA

Example – server source code

```
creds_value_t value;
creds_type_t type;
require_type = creds_str2creds("user-data", &require_value);
fd = accept(sockfd, &cli_addr, &clilen);
ccreds = creds_getpeer(fd);
allow = creds_have_p(ccreds, require_type, require_value);
if (allow)
   write(fd, MESSAGE("GRANTED\n"));
else
   write(fd, MESSAGE("DENIED\n"));
```

Privacy protection

NOKIA

Protected Storage

- Provides integrity and confidentiality protection services for applications
  - Runtime and offline protection
- Used to protect integrity of TCB components
  - Security policies, certificates, configuration files
- Storage types
  - Private / shared
  - signed / encrypted
- Uses cryptography
  - Application specific key: K(AppID, device key)
  - Shared key: K(resource token, device key)
-

Development scenarios

- Normal mode
  - Generate with SDK developer policy update package bound to EMEI
    - Includes personal public key
    - Install policy update package on the device
  - Sign your sw package with your private key
    - Install to device
- Hacker mode
  - Flash developer mode SW image
  - Security is enforced
  - But possible to change policy and allow your sw get needed credentials

Links & Questions

- Public project on
  - http://meego.gitorious.org/meego-platform-security
  -

Thank You

NOKIA

Extra slides

Manifest for Device Security Update

```
<aegis>
  <domain name="MyDomain" rank="30">
    <allow>
      <credential match="*"/>
      <deny>
        <credential name="drm"/>
      </deny>
    </allow>
    <origin>
      <keyinfo>
           mQGiBEO6XBMRBACFyOjxs17kkn0dnzRlMDHFZwcLR3A0xACvC97jbmSvuiH2J1Ku
           R1JkFqCNGv3yzvtjfLMRrNfmIgitOOaPmjK4erQoXM2cyrHlsk/OXLM2aGcR8PGE …
     </keyinfo>
    </origin>
  </domain>
</aegis>
```


NOKIA

DBUS Manifest example - server

- Server

```
<aegis>
<provide>
     <credential name="access" />
     <dbus name="com.maemo.Aegis.example" own="aegis-dbus-server" bus="session">
         <node name="/">
             <interface name="Aegis.Example">
                  <annotation name="com.maemo.secure.Access" value="access"/>
             </interface>
       </node>
     </dbus>
</provide>
<request>
     <for path="/usr/bin/aegis-dbus-server" />
</request>
</aegis>
```

DBUS manifest example - client

- Client

```
<aegis>
  <request>
     <credential name="aegis-dbus-server::access" />
     <for path="/usr/bin/aegis-dbus-client" />
   </request>
</aegis>
```

DBUS generated policy

-

```
<busconfig>
  <policy context="default">
    <deny own="com.maemo.Aegis.example"/>
  </policy>
  <policy creds="aegis-dbus-server::aegis-dbus-server">
    <allow own="com.maemo.Aegis.example"/>
  </policy>
  <policy context="default">
    <deny send_destination="com.maemo.Aegis.example" send_interface="Aegis.Example"/>
    <deny receive_sender="com.maemo.Aegis.example" receive_interface="Aegis.Example"/>
  </policy>
  <policy creds="aegis-dbus-server::access">
    <allow send_destination="com.maemo.Aegis.example" send_interface="Aegis.Example"/>
    <allow receive_sender="com.maemo.Aegis.example" receive_interface="Aegis.Example"/>
  </policy>
</busconfig>
```

More examples

-

```
<aegis>
  <request>
    <credential name="UID::email" />
    <credential name="GID::email" />
    <for path="/usr/bin/aegis-dbus-server" />
  </request>
</aegis>
```