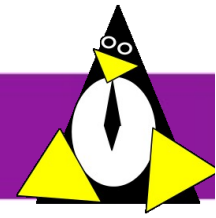
Security model for embedded systems using Smack*

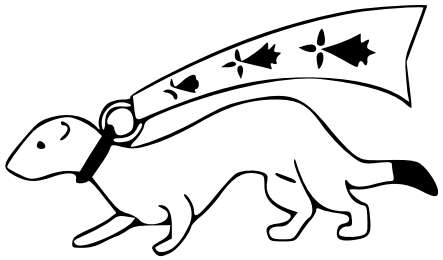
Simple but secure

* **S**implified **M**andatory **A**ccess **C**ontrol **K**ernel



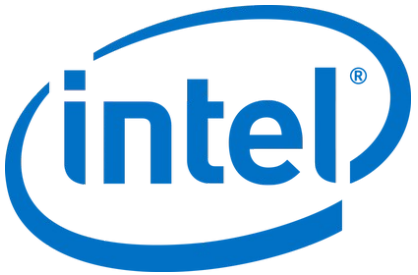
Context

- José Bollo



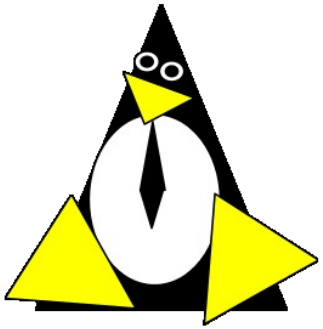
- Eurogiciel

- Intel

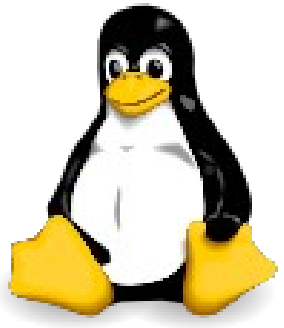


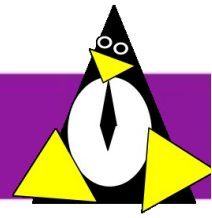
- Tizen

- Smack



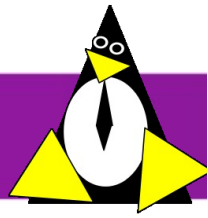
- Linux





Smack overview

- The author of **Smack** is mainly **Casey Schaufler**.
- In Linux **since kernel 2 6 25** – 17 April 2008 – as a **LSM** (Linux Security Module)
- Evoluting since this first days.
- Inside **Tizen** since the first days (2012).
- Use **extended file attributes** to store data relating to files.
- Controlled via a filesystem interface: **smackfs**.
- Controls accesses of processes to files, IPC, sockets and processes (ptrace, signals, ...).



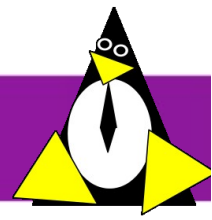
The Smack rules

- Smack's rules have 3 items:
  - the **subject's label**
  - the **object's label**
  - the **access**

Simple !!!

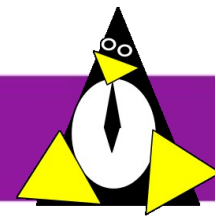
This rule tells to allow **read**, **write** and **execute** access to objects labelled **User** for the processes labelled **System**.

What are labels? What are subjects? What are objects? How to set?



The Smack vocabulary

- **Labels** are just text (of valid ASCII characters) without any special meaning: they are compared to equality (case sensitive: a≠A).
- **Subjects** are running processes: any running process has a smack label.
- **Objects** are **files**, **IPC**, **sockets**, **processes**.
- The label of a running process is called its **context**.
  - The commands `id`, `ps` (option -Z or -M), `ls` (option -Z) are prompting the contexts of the current process, the running processes, the files.
- The grantables **accesses** are: **read** (r), **write** (w), **execute** (x), **append** (a), **lock** (l), **transmute** (t).



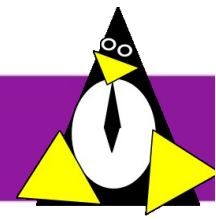
Setting Smack

- How to set context? **You can't!** Except if you have the capability **CAP_MAC_ADMIN**.

```
# chsmack --access label file
# echo -n label > /proc/$$/attr/current
```

- How to set rules? **You can only reduce accesses** for the current thread (inherited by cloning). But if you have the capability **CAP_MAC_ADMIN**, you can change all rules.

```
# echo “subject object rwt” > /sys/fs/smackfs/load-self2
# echo “subject object rwt” > /sys/fs/smackfs/load2
# echo “subject object rwt” > smackload
```

Targets devices

mobile hansets

In-vehicle infotainment (IVI)

television

TIZEN™

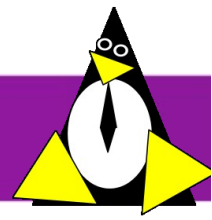
NUCs and boxes

Targets usages

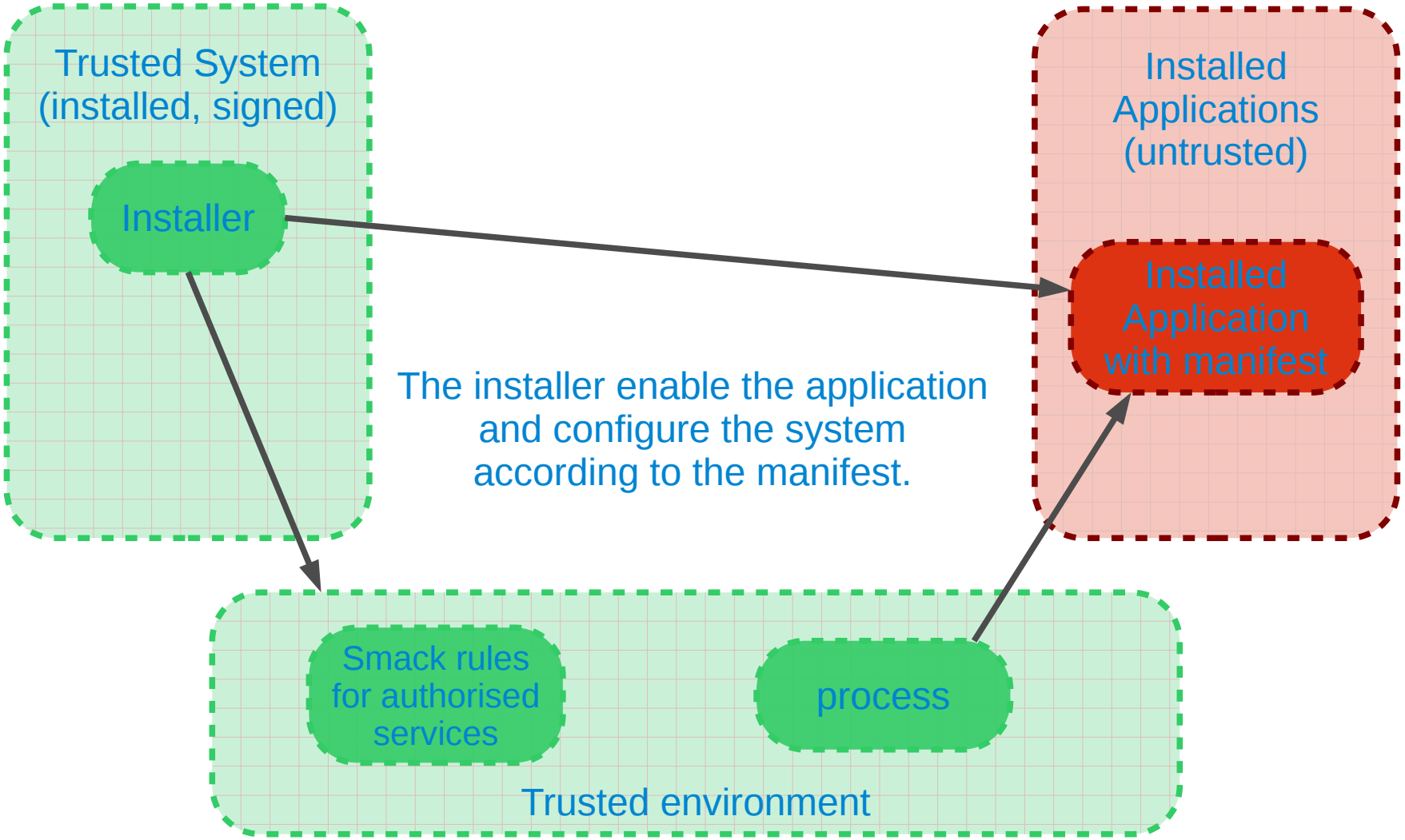
	Single seat	Multi seats
Single user	handsets boxes	
Multi users	tablets - laptops NUC	IVI

IVI is using **Wayland**

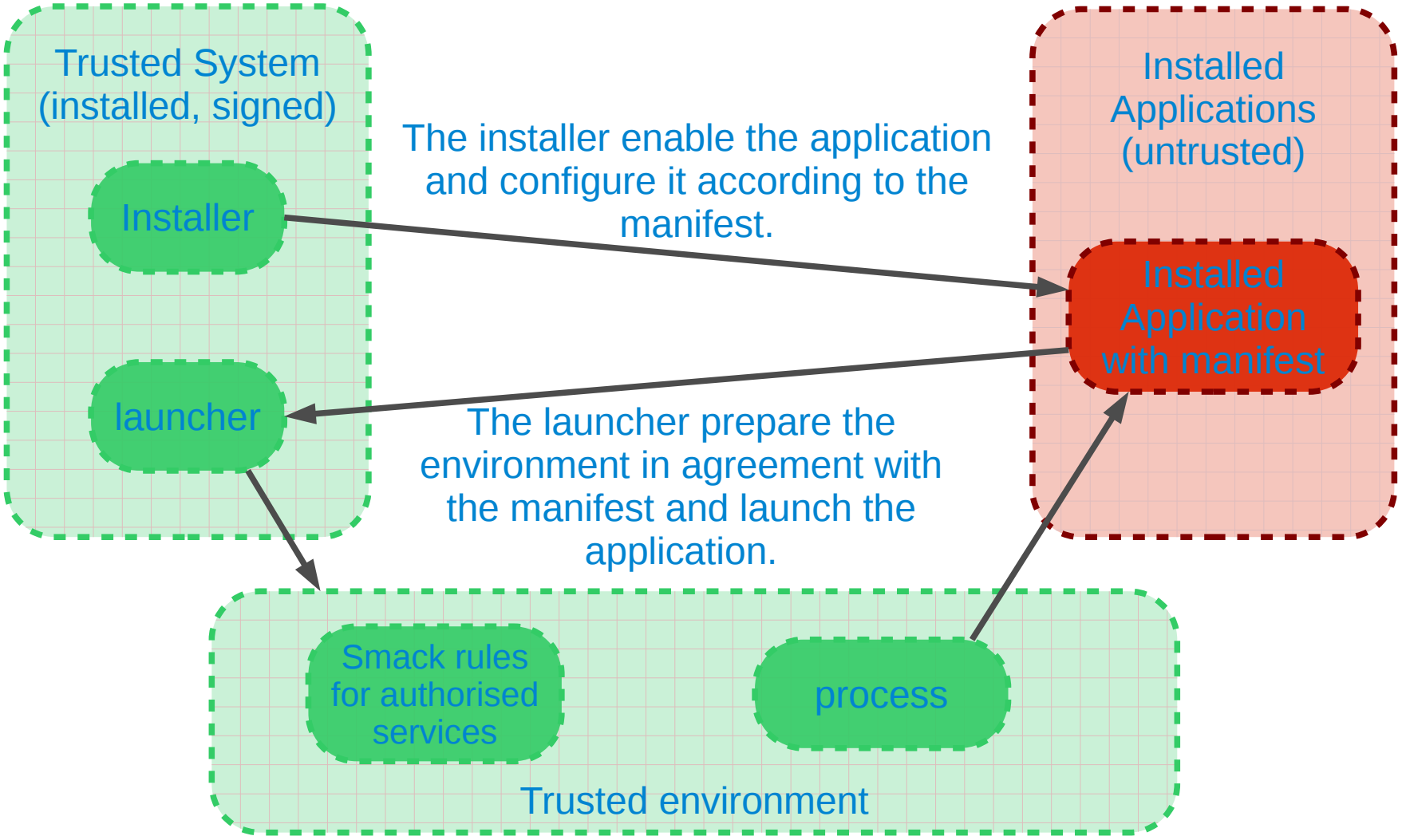
Multi seats is meaning that several users are using the same system through several interfaces.



Installer only model



Installer + launcher model

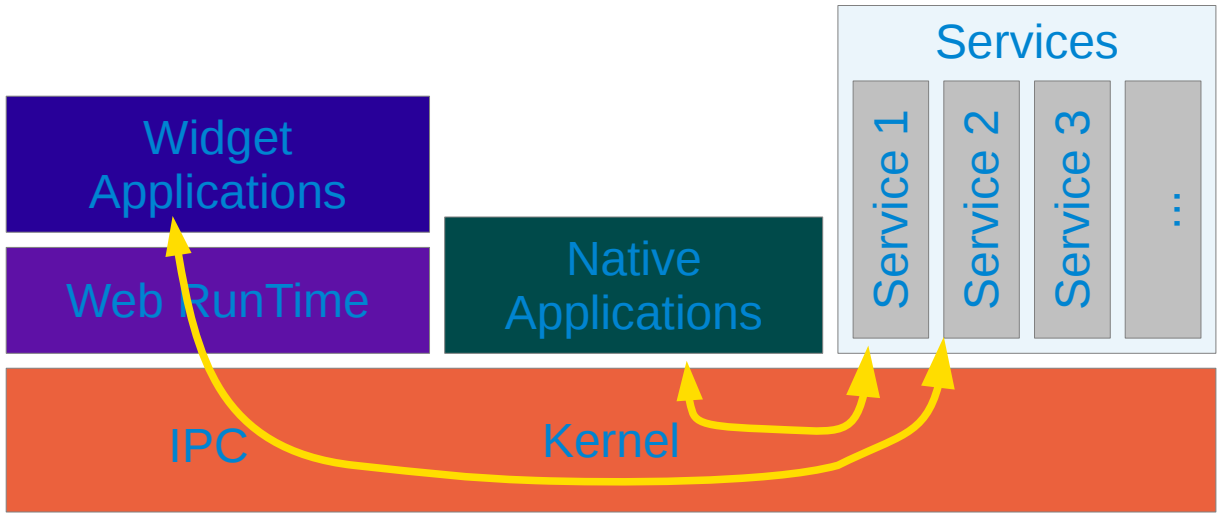


Security of applications

Tizen offers the possibility to install applications that are either natives or widgets (W3C compliant) or a mix of the both.

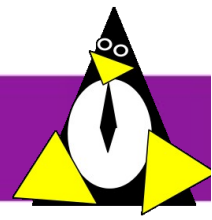
Each application has potentially access to a wide variety of services.

The accessed services MUST be conform to what the manifest of the application is claiming for. That is the condition to have a trusted system, a secure system.

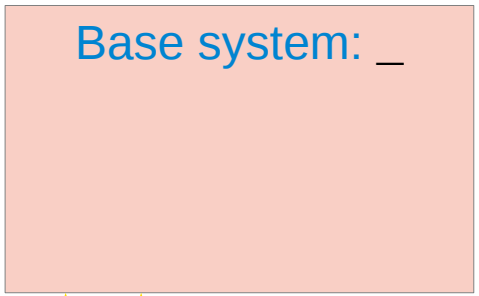


Implementations

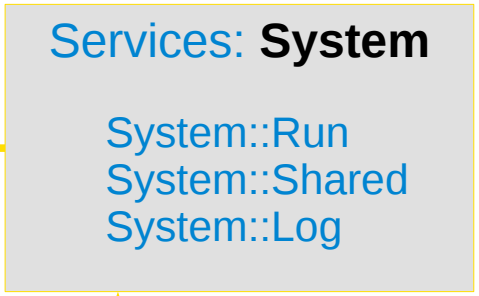
- The problem is difficult due to its power characteristic: controlling N ressources for M kinds of accesses brings to M^N cases!
- For Tizen 2.0 there was many smack rules (for a basic mobile hanset, not less than **33232 rules!**)
  - Each application have a own context label
  - The rules are the spare matrix of all the authorised accesses
- For tizen 3.0 IVI the three-domains model will be used.
  - Basically, three subject labels exist: **_**, **System** and **User**
  - There few more object labels
  - The rules are restricted to the minimum
  - It requires a launcher to achieve the full control of accesses



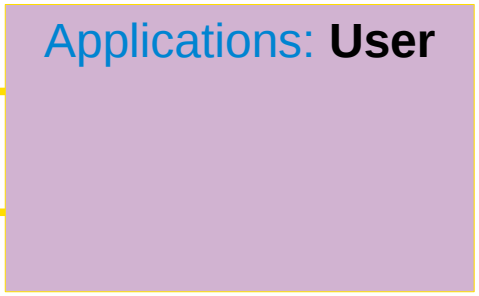
Three-domains model overview



The floor domain provides the foundation upon which the system is built



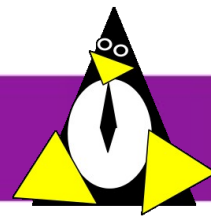
The System domain is comprised of the basic system services and the data they maintain.



The User domain is comprised of the services that interact directly with the person using the Tizen system and the data those services maintain.

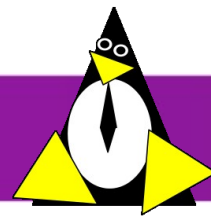
Links

- LSM Smack
  - http://schaufler-ca.com/
  - https://www.kernel.org/doc/Documentation/security/Smack.txt
- Smack utilities
  - https://github.com/smack-team/smack
- Tizen
  - https://www.tizen.org/
  - https://wiki.tizen.org/wiki/Security:Smack
  - https://wiki.tizen.org/wiki/Security:SmackThreeDomainModel



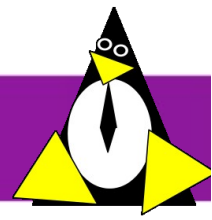
Summary

- It works well and is really simple to learn.
- You can activate it on any Linux kernel.
- The embedded linux distribution TIZEN implements it and its community can help you.
- You can contribute to improve the smack tools and models.



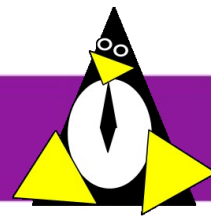
Questions

Thanks



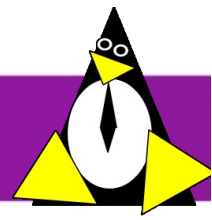
EUROGICIEL

- Open source development and integration:
- Maintainers for tizen.org (Base, Test, Web Framework,... domains)
- Embedded systems for real-time multimédia:
  - Widi/Miracast stack,
  - Wayland/Weston,
  - Webkit2 browser with HW acceleration,
- Application: HTML5/CSS3, jquery, jqmobi, Cordova
- Location : Brittany – France
- http://www.eurogiciel.fr/



Evolutions of Smack

- The author of **Smack** is mainly **Casey Schaufler**.
- In Linux **since kernel 2 6 25** – 17 April 2008 – as a **LSM** (Linux Security Module)
- Evoluting since this first days.
  - Lock access mode (kernel 3.13)
  - Support for multi-rule write to load2 and change-rule (kernel 3.12)
  - Maximum value for CIPSO category change from 63 to 184 (kernel 3.12)
  - Longer Smack labels (24->255) and recursive transmute (kernel 3,5)
  - Transmute access mode (kernel 2.6.38)



Three-domains model rules

Explicit rules 1/2

Subject	Object	Rights
System	System::Run	rwxat
System	System::Shared	rwxat
System	User	rwx
System	^	rwxa
System	_	l
User	System	wx
User	System::Run	rwxat
User	System::Shared	rx
User	_	l

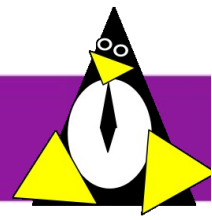
Explicit rules 2/2

Subject	Object	Rights
^	System	rwxa
^	System::Run	rwxat
_	System	wx
_	System::Run	rwxat

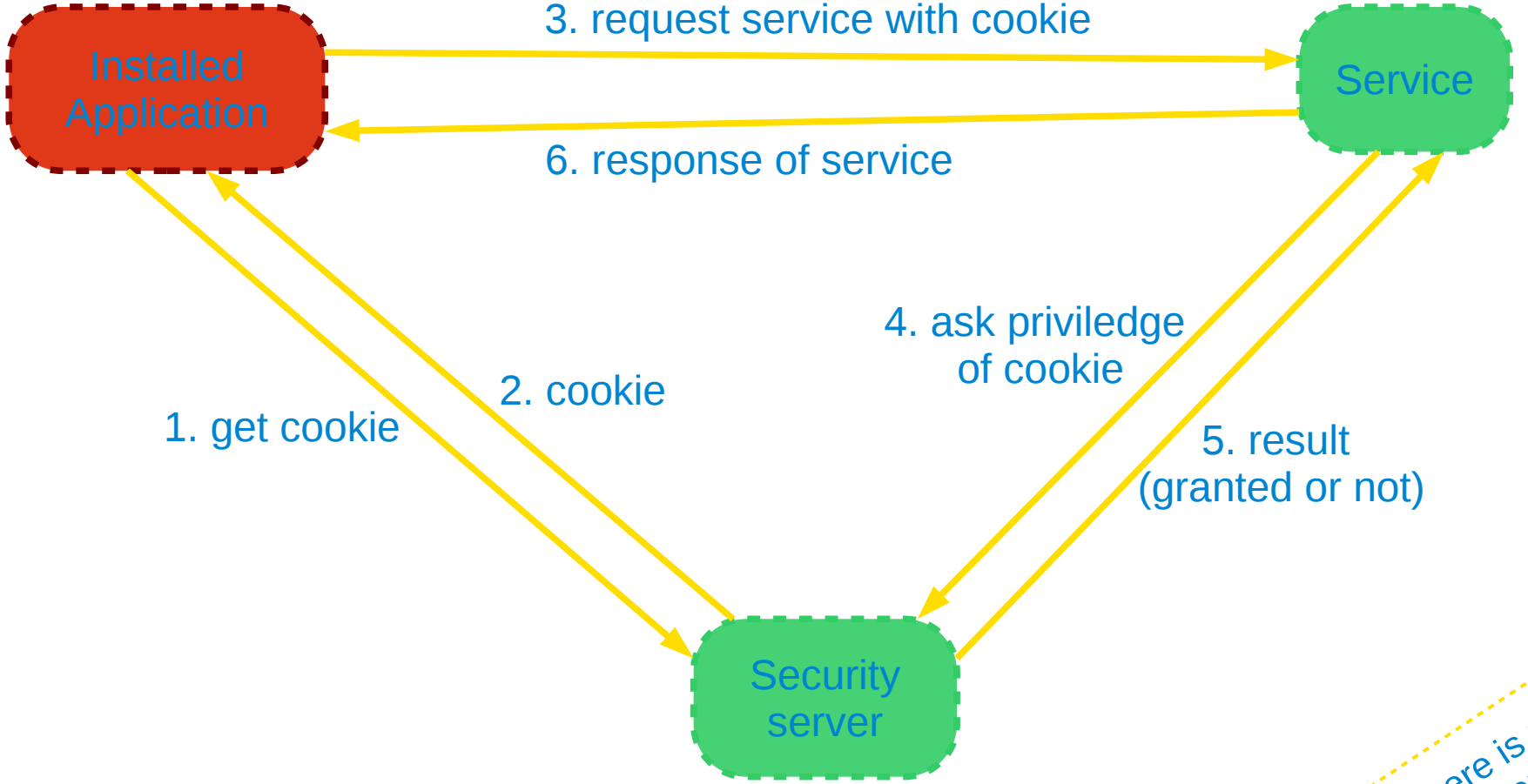
Some implicit rules

Object

Subject	_	^	*	Y
_	rwxatl		rwxatl	
^	rx	rwxatl	rwxatl	rx
*				
X	rx		rwxatl	rwxatl if X=Y



Security server



But there is no Smack here?!