Managing collaboration and competition of multiple WAN services in a residential network

Wouter Haerick, Nico Goeminne, Jan Coppens, Filip De Turck and Bart Dhoedt

Abstract Open service platforms like the OSGi-platform offer a standard, scalable way for service providers to remotely deploy their services inside many residential networks. However, the lack of control by WAN service providers on the home environment together with too complex end-user policy configuration hinder widespread e-deployment of services into the home. Several architectures have been presented for next generation home networks, coping with the deployment, discovery and run-time control of residential services in order to enforce service levels. However, to evolve towards true collaboration scenarios where a service from one service provider can interact with a service from another service provider without configuration inconvenience, or where a service from one service provider can co-exist with an identical service from another provider on the same device, proper security and policy configuration needs to be addressed. This paper contributes therefore to the already presented architectures by discussing secure remote policy configuration in a multi service provider environment. A security framework is proposed based on the OSGi specification that limits not-trusted service providers in their control on other services. The strength of the framework lies in its generic XACML-compliant policy configuration module and its compatibility with existing services. This makes the framework easy to adopt for remote configuration providers, which allows service providers to delegate configuration support to a service aggregation provider.

1 Introduction

The past three years increased effort has been spent by the academic world, industry and standardization bodies to lower the barriers to integrate wired and wireless

Wouter Haerick, Nico Goeminne, Filip De Turck, Bart Dhoedt
University of Ghent, Gent, Belgium, e-mail: wouter.haerick@intec.ugent.be

Jan Coppens
Alcatel-Lucent, Antwerp, Belgium e-mail: jan.jc.coppens@alcatel-lucent.com

home devices in a vendor-neutral way and turn a house into a smart home. Organisations like DLNA and UPnP Forum have succeeded to specify interoperable middleware blocks to share local media services with compliant output devices spread over the home. These efforts have enabled a new way of digital entertainment allowing seamless sharing of music, video and pictures using customer equipment from different vendors. Both organizations have their main focus today on discoverable media services running inside the LAN environment. As a result these services have an inside view of the home network and hence can use the local network information in their auto-configuration and or self-healing processes. In contrast to DLNA and the UPnP Forum, the DSLForum - which also represents the telecommunication industry and service providers, apart from the customer electronics industry - is concerned on how service providers can get a view on the individual home network in order to offer predictable QoS like the locally running services can do, and thus to enable WAN service providers to compete with locally running plug-and-play media servers. The question rises here which WAN service should be allowed to discover which other WAN or LAN services, and more in particular which service should be able to reconfigure or interact with which other services. This paper therefore deals with the problem of how to remotely enforce unidirectional use of a service by another service with no or limited configuration by the end-user, and how to deal with overwritten or conflicting configurations that could possibly degrade the service level from other services in a multi service provider environment.

In the remainder of this paper, we will consider a service provider to be a third party provider which offers a client-server service to a residential network as depicted in Figure 1. The server part could run on the WAN network or could be remotely deployed into the residential network. In both cases, the service provider has interest in knowing which other services are present in the home network, and how the LAN network is behaving, in order to offer predictable QoS. In a multi service provider environment, where multiple WAN service providers offer IP services to a single home network, it gets more complex as configuration requests from one service provider could conflict with previous configuration requests.

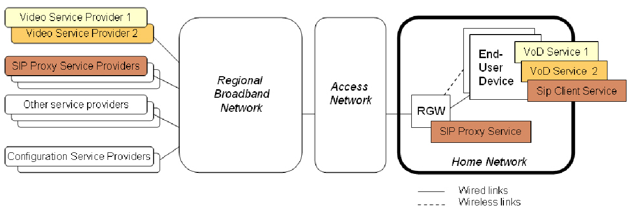


Fig. 1 End-to-end residential service architecture in a multi service provider environment with a single internet access provider

In any case we would like to avoid that service providers start competing for example for shared bandwidth and therefore continuously change each other settings.

In a more general case, we would like to take a look at the concept of a ”configuration service provider” that is in charge of the service configuration of many other service providers. This type of service provider is also introduced in DSL TR-069 WAN CPE Protocol specifications. We contribute to this concept by defining a secure framework to delegate the configuration from a service provider to a configuration service provider. Internet service providers could take up the role of a configuration service provider for all the services that are offered through their internet service and access gateways.

In the margin of the standardization organisations, a short note can be made on ”the battle of end-user devices”. When looking at the MP3 player market dominated by Apple’s iPod player, and at the rivaling Microsoft Zune player, it becomes clear that the battle on the end-user device is not purely about selling more devices, but about installing marketshare for their respective music service providers iTunes and Zune.net. The same battle is ongoing between Skype certified devices (Skype protocol) and MSN Live compatible devices. These devices contain software that acts as (1) device management software, (2) client software to the WAN service provider and (3) as local media player. The inherent configuration issues of multiple service provider environments are however today often avoided by offering devices, pre-configured by the manufacturer, to work only with a single service provider.

However, the general purpose terminals are evolving to support multiple services, and thus different service providers become able to provide each a set of services on one single device. This leads to a situation were different parties are involved in the management and configuration of home network resources, as well as in the service deployment. Hence, one should be aware that:

- Subscribing to different services from different service providers often requires a complex re-configuration of the terminal or the home gateway. While reconfiguring a device, the integrity of the local network and other running services must be guaranteed to ensure that installed services will not degrade or stop working.
- Configuration conflicts can arise as a result of applications which are able to provide themselves a certain configuration without intervention of a configuration provider or a local user. Also, configurations made by the end user can conflict with previous settings executed by service providers.

The next section of this paper describes related work on home network architectures and remote configuration. Then, an OSGi-based security architecture is proposed. First, the generic XACML-compliant policy enforcement components are introduced. These components comply with the ISO security model on authentication and authorization. Afterwards, two architectural options are described to deal with multi-service provider security using the underlying XACML components. In a next section, a remote security interface is proposed that follows the TR-069 requirements on WAN CPE remote configuration. The complete architecture is illustrated with a proof-of-concept implementation in the following section. For two WAN service providers, one offering a video service and the other offering a SIP proxy service, the usage of the security framework is discussed. A last section contains the conclusions.

2 Related work

Several architectures have been presented for next-generation residential gateways [2],[3],[4] as core component of the smart home. While each of these architectures use their own naming for the subcomponents, the different components can be mapped to one of following three layers: a driver layer, a common access layer and a service layer. The common access layer consists of both hardware and native software and provides access to hardware devices inside the LAN or WAN. The native software could for example provide functionality to guarantee QoS. The driver layer acts as a translator between the common access layer and the higher level service layer. The service layer consists of hardware independent services that can remotely be deployed by a service provider. These architectures however do not detail an additional layer to protect the local services from non-trustworthy service providers.

With respect to security, specifications are discussed in the UPnP Forum [8], the Home Gateway Initiative [5] and also in the latest OSGi release 4 specifications [6]. The DSLForum specifications for remote CPE management [7] refer to SSL, IPsec and WS-Security as security mechanisms. Although final documents are available for UPnP security and OSGi security, it lacks of practical implementations. A main reason could be that the configuration of security settings did not get the focus it possibly needs to make it convenient to use. With our work, which includes a proof-of-concept implementation, we aim to contribute with convenient security configuration components. To increase the convenience, it is described how the security settings can be delegated from local end-users or service providers to configuration providers. In addition, we focus on multi-provider environments, where WAN service providers can possibly overwrite configuration settings that could degrade other services. With respect to multi-service broadband architectures we would like to refer also TR-101 from DSLForum that provides a standardized approach to an Ethernet-centric multi-service broadband architecture as well as a QoS and multicast blueprint.

3 Secure multi-provider architecture

We consider an end-to-end network with multiple WAN service providers offering an IP-based service to one ore more wired or wireless devices in the home. Two alternative architectures will be explained that limit service providers in the actions that can be performed from a remote WAN-location. The first architecture combines an XACML-based policy framework with multi-service-provider security enforcement to hide certain services from certain service providers. In fact, the framework does not restrict the visibility towards the locally running services but enforces very stringent rules so that unauthorized service providers will never be able to make configuration changes to other services. A second architecture is proposed that indeed changes the visibility, and also protects the life-cycle actions to avoid that any service can start, stop or uninstall a service with too stringent security restrictions. Before discussing the two alternative architectures to deal with the multi-service provider issues, the underlying XACML policy components are discussed.

3.1 XACML-based security components

We adopt the ISO security model for authorization and authentication (Figure 2) and apply it to a service-oriented home architecture that is built upon the OSGi framework.

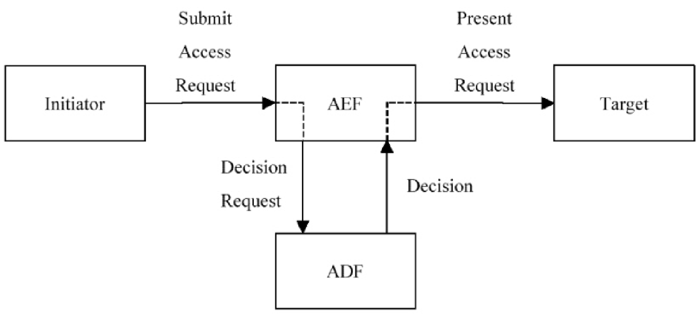


Fig. 2 ISO 10181-3 Access Control Framework

In the ISO model, the Access Control Enforcement Functions (AEFs) mediate access requests and submit decision requests to Access Control Decision Functions (ADFs). The ADFs make the final decision to allow or deny the access of the target by the initiator. The AEF requires Access Control Information. This information typically includes (1) Initiator information, (2) Request information and (3) Target information.

In order to take a decision, the ADF should first authenticate the initiator. If the authentication is successful, the ADP should enrich the Access Control Information with contextual Access Decision Information. With this information, the ADF can police based on predefined policy rules.

The decoupling of the AEF and ADF allows representing policies in a generic format while the access control enforcement remains target specific. This way an AEF for OSGi services and another AEF for Internet access can both share common, generic policies that use the same policy language.

To control the access to the WAN, to locally running OSGi services and even to local non-OSGi services, distributed enforcement is needed at the firewall, OSGi service registry and at other services (eg a print server). Although the enforcement points are distributed, a central policy repository makes the administration a lot easier, and allows for more consistent reasoning about possible policy conflicts. For such a central policy repository, it is important to adopt a standard policy language. Only then the policies can be interpreted and shared by different applications in the home. With respect to access control, this leads to the standardized specifications of XACML. XACML is standardized by the Organization for the Advancement of Structured Information Standards (OASIS) and describes both a policy language and an access control decision request/response language (both written in XML). The policy language is used to describe general access control requirements, and has standard extension points for defining new functions, data types, combining logic, etc. The request/response language lets you form a query to ask whether or not a given action should be allowed, and interpret the result. The response always includes an answer about whether the request should be allowed using one of four values: Permit, Deny, Indeterminate (an error occurred or some required value was missing, so a decision cannot be made) or Not Applicable (the request can't be answered by this service). The power of XACML lies in its generic and powerful format to describe access policies for any application. Each policy corresponds with a single access control policy and is expressed using multiple rules. Policies can be stored in different files, stored locally or distributed, or can be bundled inside a PolicySet document.

The diagram below shows a total view of the end-to-end security architecture for secure service delivery in OSGi-based home environments. The OSGi-based security bundles, which run on the residential gateway, consist of 3 sub-components: (1) A TR-069 Security Management Agent (MA), (2) an Access Control Service and (3) a Security Configuration Service.

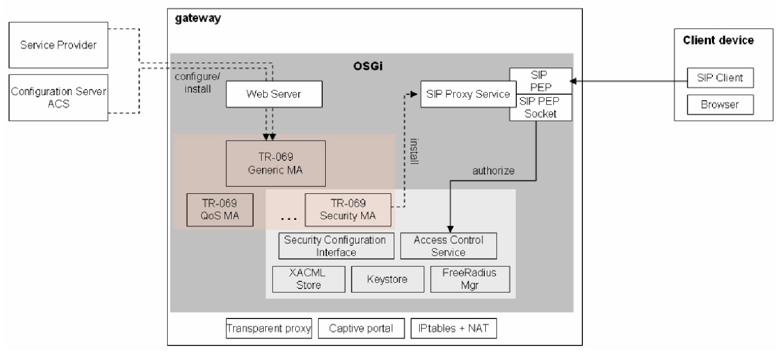


Fig. 3 XACML-based security component architecture with central gateway running OSGi

3.1.1 TR-069 Security Management Agent

For each bundle that gets installed via the TR-069 security management agent, the integrity of the bundle is verified by checking the signed jar file. If the key used for the signature is known to the security framework, this means it is stored in the key store, then the bundle gets access to all methods offered by the Security Configuration Service. These access rights are stored using the XACML syntax. If required, these settings can be overwritten in a way that only a subset of protected methods is accessible by the installed bundle. This fine grained access control mechanism thus allows to block service request on a method-level instead of on the service-level. This bundle depends on a java TR-069 library and introduces a new new TR-069 security object ”Device.Services.Management.ServiceProviders”. This object is an extension of a BundleManager object tracking the different services/bundles of each service provider. The new TR-069 object allows already trusted service providers to add/remove their certificates and install/uninstall/share/unshare their services with services of other service providers, increasing the configuration convenience for the end-user. The TR-069 ServiceProviders object could also act as a TR-069 object proxy for any TR-069 configuration request. In that case the proxy needs to extract the additional security parameters (retrieved by the SOAP header or SSL certificate) enabling the TR-069 ServiceProviders object to identify the origin of the request. Based on the origin parameters, the TR-069 ServiceProviders object from the proxy can select the corresponding TR-069 configuration object. Based on the origin, the TR-069 ServiceProviders object decides if the requested configuration to that specific TR-069 target object is permitted. The communication is typically allowed if the origin of the request is also the owner of the targeted object.

3.1.2 Access Control Service

The access control service provides a generic access interface to any OSGi service that is protected with a Policy Enforcement Points (PEP). Note that a PEP from the XACML specifications corresponds with AEF from the ISO security model. Bundles that do not have such a PEP can be automatically modified using an access control adder script. The latter creates a new service interface that adds authorization to each method, updates the Activator class-file to register the new service interface into OSGi and adapts the meta-inf file to include the necessary java packages for XACML support. The access control service has only one method in its interface:

```
public boolean requestAccess(Subject s, String obj, String action, List args)  throws
AccessDeniedException;
```

By supplying the proper information of the Subject - for example the symbolic name of a BundleSubject or the SIP-id of a SIPSubject in case of a SIP proxy service - and the information on the target service including the action to perform, the access control service makes a decision whether or not to allow access to the Subject. Therefore, an XACML access request is constructed and matched against the XACML policies in the local policy store. For the implementation we considered the following requirements:

- Policy Initialisation: Initial policies of trused services should automatically be added to the policy store the first time a service starts up in the OSGi framework. These initial configuratons are of utmost importance to achieve a high convenience. At any time modifications to the policies can be performed by the local user or by the remote configuration server.
- Highly flexible policies: A (modified) policy can be made up of simple predefined functions like string equations or regular expressions or custom made functions. These custom made functions can refer to methods exposed by other OSGi services. Using custom functions one can easily construct accounting policies for billing services.

The Access Control bundle depends on the com.sun.xml library, the javax security auth library, the Policy Access Point bundle, the Policy Authentication Point bundle and the credential bundle.

3.1.3 Security Configuration Service

The Security Configuration Service offers the interface to all underlying security mechanisms like the XACML policy store, the key store, the RADIUS database and the IPtables firewall. This interface is protected with a PEP which makes it only accessible for trusted, signed bundles that have access rights for the individual methods.

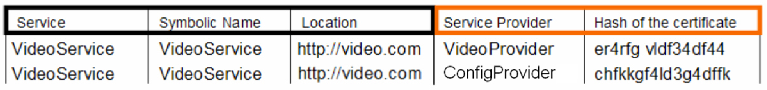
3.2 Architectural option 1: TR-069 Management Agent with multi-service-provider security enforcement

Service providers can control services that run on a home gateway using the TR-069 protocol, sending messages over SOAP/HTTP. In order to enforce strict security rules, each service provider participating in the security framework should authenticate itself to the TR-069 management agent. Three different options are possible:

- two-way SSL is used between the service provider and the home gateway; during the SSL handshake the certificate of the service provider is stored by the Management for later use;
- WS-Security is applied to the SOAP messages including the service provider certificate; the security information is stripped from the SOAP header by the Management Agent and stored for later use;

- the certificate is sent to the Management Agent as one of the TR-069 datafields. These data is securely sent (at least using a signature to verify that the message has not been modified by an untrusted party) and forewarded to the service management bundle.

No matter which option is used by the Management Agent to retrieve the origin of the service provider that sends configuration requests, the MA should forward the security information (or a link to the place where this information is stored) to the service management bundle. The latter component will then verify if the security information corresponds with a trusted party by looking up the certificate in the key store. If the service provider is trusted then the Multi Provider policy is evaluated to verify if the service provider is allowed to change the policies/configuration of the target object included in the configuration request. The multi provider security file should include the following information, which expresses for each service which service provider is allowed to change the (XACML) configuration.



Service	Symbolic Name	Location	Service Provider	Hash of the certificate
VideoService	VideoService	http://video.com	VideoProvider	er4rfg vldf34df44
VideoService	VideoService	http://video.com	ConfigProvider	chfkkgf4ld3g4dffk

Fig. 4 A proposal for a multi provider policy format

Figure 4 depicts a video service that not only can be configured by its owner but also by the configuration provider. This multi provider policy file can only be updated by the service owner, or by the local administrator (through a confirmation pop-up).

Figure 5 shows how we extend the XACML-based archicture from the previous section with a modified Service Management Bundle to allow interaction with the multi provider policy file.

A service provider that wants to configure a service on the home gateway now only has visibility on those services for which it has the grants (as stated in the multi provider configuration file). The actual access to the individual service methods is further controlled by the XACML policies. The result is fine-grained access control used in favor of multi provider security.

Figure 6 shows how the multi service provider security components filter the view a service provider has on the home gateway. Service Provider A is a trusted service provider and has the grants to configure its own videoService but also to modify the Resource Sharing Framework [1] configuration to make resource reservations in case additional CPU is required to display a videostream. Apart from those two services it can also see the untrusted services ”X10” and ”back-up”. Service Provider B can only configure its SIP service and the two untrusted services. While this configuration restricts the use of a service by another service bundle, it does not restrict the visibility of the services offered by the video service or any other local bundles.

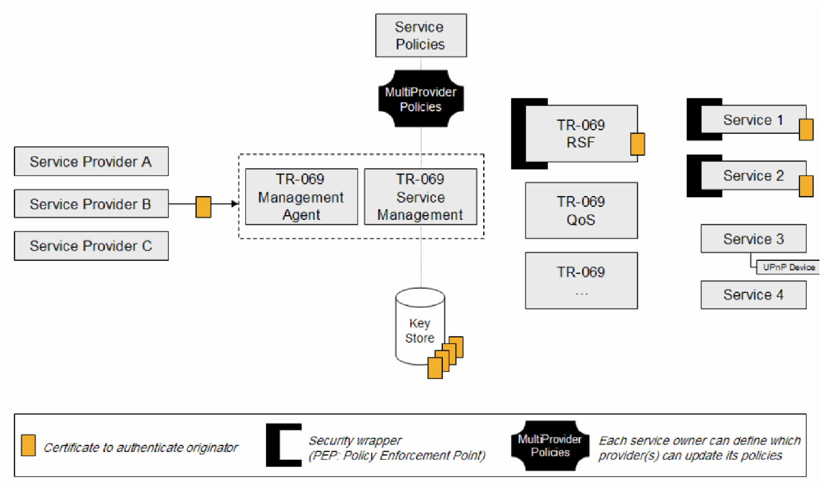


Fig. 5 Architectural option 1: Secured Management Agent to support multi provider configuration

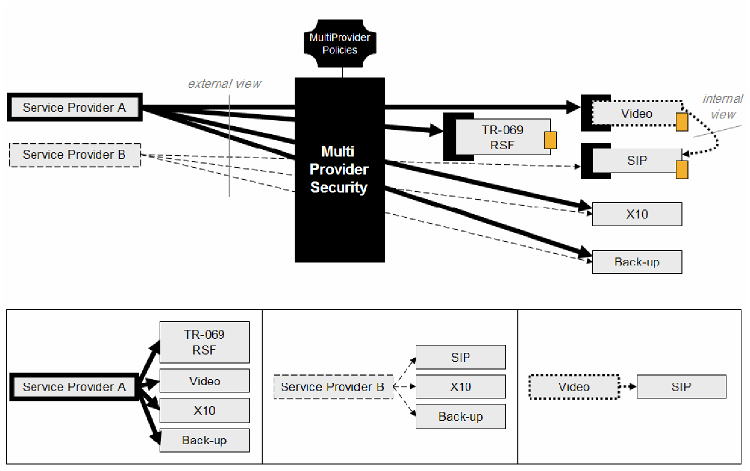


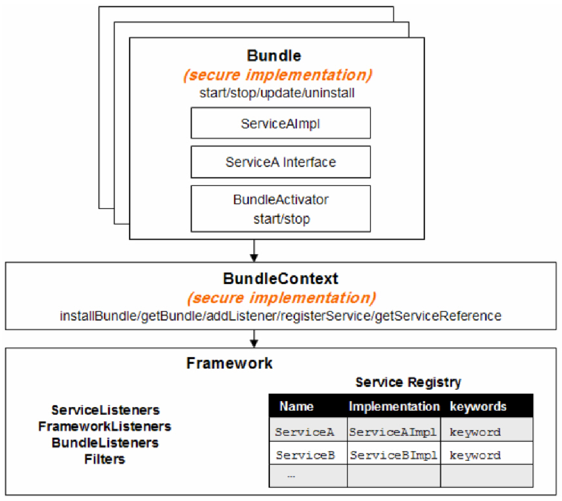
Fig. 6 Restricted external and internal visibility

3.3 *Architectural option 2: Modification of the core OSGi framework to add bundelContext security*

An alternative solution to obtain multi provider security is explained in this section but requires changes to the core OSGi framework. This security solution can thus not be added as a separate bundle by a remote configuration server. It must be included in the OSGi framework jar-file and therefore should replace the existing

OSGi framework. As a direct consequence of the integration into the framework, it has the advantage that the security solution can not be replaced easily by another bundle at runtime. The alternative solution we propose will prevent that a trusted bundle will be replaced with a malicious bundle because it is intertwined with the core OSGi objects BundleContext and Bundle. Therefore, it will no longer be possible for a random bundle to stop, update or uninstall another bundle.

The BundleContext object has the global overview on the OSGi registry and thus on all the locally registered bundles and listeners (see code listing 1 below). The actual life-cycle controlling methods are defined by the Bundle interface (see code listing 2 below). By modifying the BundleContextImplementation and BundleImplementation each of these methods can be linked to the XACML compliant security framework in order to evaluate if the calling party has the permissions to perform the life-cycle method.



Name	Implementation	keywords
ServiceA	ServiceAImpl	keyword
ServiceB	ServiceBImpl	keyword
...		

Fig. 7 Architectural option 2: Modifications to the core OSGi framework for multi provider security

In the implementation of the BundleContext object and the Bundle object, each of the sensitive operations (installation of a bundle, get Service References) should be extended with a condition that first evaluates the permissions of the calling bundle. Compliant with release 4 of the OSGi specifications, the origin of the calling bundle can be verified against the bundlename or the bundlelocation.

The XACML compliant service policies define which actions should be made accessible to which other bundles. For OSGi frameworks that are compliant with OSGi release 4, the implementation of the bundle interface method "hasPermission" could be linked to the Service Management bundle of the security framework to evaluate the request for permission.

Code listing 1 - An extract from the BundleContext Interface

```
public interface BundleContext {
public abstract String getProperty(String s);
public abstract Bundle getBundle();
public abstract Bundle installBundle(String s) ...;
public abstract Bundle installBundle(String s, InputStream is) ...;
public abstract Bundle getBundle(long l);
public abstract Bundle[] getBundles();
public abstract void addServiceListener(ServiceListener sl, String s) ...;
public abstract void addServiceListener(ServiceListener sl);
public abstract void removeServiceListener(ServiceListener sl);
public abstract void addBundleListener(BundleListener bl);
public abstract void removeBundleListener(BundleListener bl);
public abstract void addFrameworkListener(FrameworkListener fl);
public abstract void removeFrameworkListener(FrameworkListener fl);
public abstract ServiceRegistration registerService(...);

public abstract File getDataFile(String s);
public abstract Filter createFilter(String s) ...; }
```

Code listing 2 - An extract from the Bundle Interface

```
public interface Bundle {
public abstract int getState();
public abstract void start() throws BundleException;
public abstract void stop() throws BundleException;
public abstract void update() throws BundleException;
public abstract void update(InputStream inputstream) ...;
public abstract void uninstall() throws BundleException;
public abstract Dictionary getHeaders();
public abstract long getBundleId();
..
public abstract boolean hasPermission(Object obj); ..
}
```

4 Remote configuration interface

The security settings offered by the OSGi java security bundles can be exposed to a remote management server in a generic way adhering to the TR-069 protocol. The goal of the security interface is different for the configuration service provider, service providers and home user.

The benefits for the configuration service provider of a secure remote interface should include:

- centralized set-up of trust relations by the configuration service provider;
- centralized policy management by the configuration service provider;
- provide a generic interface to update policies.

Service/Bundle providers should benefit from centralized trust relations and be able to delegate policies and their management to a central policy store owner. Home users should benefit from new trusted services without any additional security configuration.

The security interface we propose is transparent for the TR-069 Management Agent (MA) (Figure 8. Hence, this MA has no knowledge about the security data model but is only involved in translating TR-069 messages in generic java calls. In particular for security configuration, a security data model has been defined that allows configuration of (1) physical credential/policy stores, (2) add credentials (certificates, passwords, etc.), (3) add policies and (4) configure visibility.

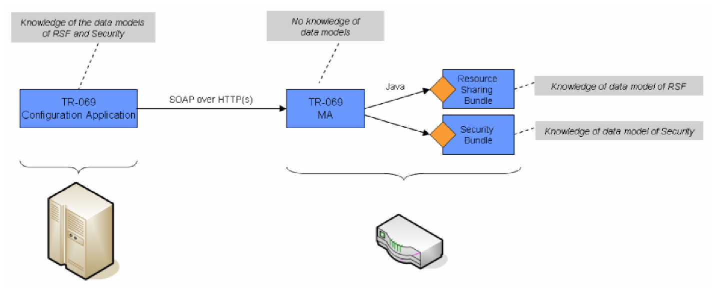

Knowledge of the data models of RSF and Security
No knowledge of data models
TR-069 Configuration Application
SOAP over HTTP(s)
TR-069 MA
Java
Resource Sharing Bundle
Security Bundle
Knowledge of data model of RSF
Knowledge of data model of Security

Fig. 8 Transparent TR-069 Management Agent.

Each TR-069 configurable entity is required to expose its functionality through the generic TR-069 java interface (orange diamonds in Figure 8. However, the implementation of the interface is different for each of these entities. The following three methods should be supported:

```
      setParameterValues(TR09Object[] args) returns TR69Result[]
        getParameterValues(String[] keys) returns TR69Object[]
getParameterNames(String parameterPath, boolean nextLevel) returns
                          TR69Parameter[]
```

The first function, setParameterValues, sets the parameters, given by their name (full path), to a certain value. This function can be used to adjust a policy, a credential or a visibility. The argument is an object that combines a key (parameter name) with a certain value.

The second function, getParameterValues, retrieves the values of the listed parameters, given by their names (full path). The agument is an array of strings containing parameter names with their full path.

The third and last function of the TR-069 interface, getParameterNames, returns a list of all parameters with their name (with full path) and an indicator which indicates whether or not the parameter can be changed. The boolean argument of this function indicates if the parameters of the next branch(es)/level also need to be included in the result.

5 Proof-of-concept implementation

The targeted setup of the demo is shown in figure 9. The setup contains a home network, representing a home environment of a potential customer, and a broadband network representing the real internet with numerous service providers. The home network has a desktop PC with a softphone, a browser and an OSGi framework installed on it and the gateway runs a captive portal, web server, authentication server and another OSGi framework. The goal is to illustrate how a video service can be configured by its service provider - or a configuration provider - to be paused by a trusted VoIP/SIP proxy as soon as a valid phone call is received.

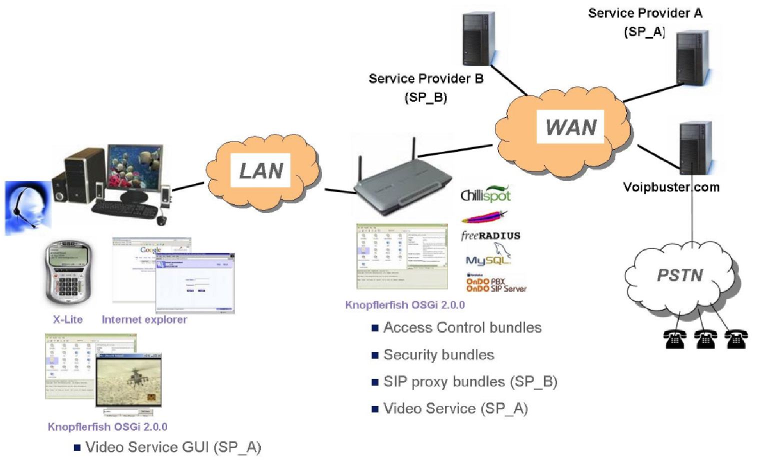


Fig. 9 Collaboration scenario: Configuration of a remote video service to be interrupted in case of incoming VoIP/SIP calls.

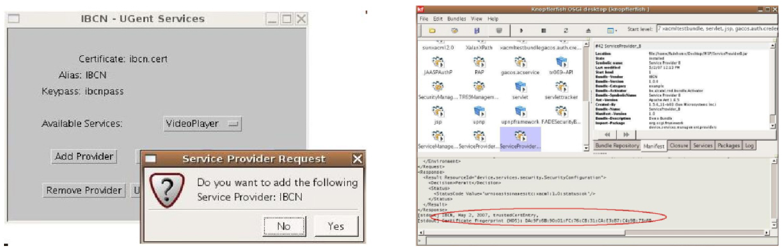
Two service providers are part of the proof-of-concept: service provider IBCN-Ugent and service provider RnD.

A standard non-protected video service from provider IBCN is parsed by two scripts two add the required security support. The first script, named the Access-Control-Adder tool, adds fine-grained access control to the video service to allow security enforcement on method level. The second script adds a signature of the service provider to the jar-file, together with an XACML default policy file. The video service is implemented as a UPnP controlpoint for the UPnP video player provided by VideoServiceGUI.jar. The controlpoint exports this (Access Controlled) service to the OSGi framework and so enables other bundles to control the UPnP video player. The use of these actions is controlled by the XACML based security framework.

Service provider RnD offers two services that would like to interact with the video player from service provider IBCN. The two services are VCRRemote and a SIP Proxy service.

Using the TR 069 ”Device.Services.Management.ServiceProviders” object, the service providers can add/remove their certificates and install/uninstall/share/unshare their services with services of other service providers. We have opted for architectural option 1 to have a security framework that is decoupled from the OSGi implementation version. We used the Knopflerfish implementation of OSGi.

First, the certificates of the trusted service providers need to be added to the central certificate store. When a service provider wants to add his certificate, a pop-up is shown asking to approve the addition of the certificate to your keystore (Figure 10). Once accepted, one can start installing services from the corresponding service provider on your gateway.



```
#42 ServiceProvider_B
Location             file:/home/fadehome/Desktop/NSP/ServiceProviderB.jar
State                installed
Symbolic name        Service Provider B
Last modified        5/2/07 12:13 PM
Start level          1
Bundle-Vendor        IBCN
Bundle-Version       1.0.0
Bundle-Category      example
Bundle-Activator     be.alcatel.rnd.bundle.Activator
Bundle-SymbolicName  Service Provider B
Ant-Version          Apache Ant 1.6.5
Created-By           1.5.0_11-b03 (Sun Microsystems Inc.)
Bundle-Name          ServiceProvider_B
Manifest-Version     1.0
Bundle-Description   Demo Bundle
Import-Package       org.osgi.framework
                     device.services.management.providers
```

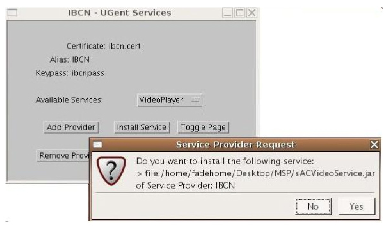
```
  </Environment>
</Request>
<Response>
  <Result ResourceId="device.services.security.SecurityConfiguration">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
  </Result>
</Response>
[stdout] IBCN, May 2, 2007, trustedCertEntry,
[stdout] Certificate fingerprint (MD5): DA:9F:6B:90:01:FC:76:CB:31:CA:E3:87:C4:98:73:6B
```

Fig. 10 After user approval (one click) the PEP from the Security Configuration Service permits the installation of a trusted certificate.

When installing a service of a known service provider, the Security Framework will ask to confirm the installation of the new service/bundle. Attempts of unknown service providers to install services are ignored. When a (signed) service is installed, it can also make use of the Security Framework. This makes it possible to create own policy and/or credential stores containing service dependant data. In the demo scenario, first the certificates of the two service providers IBCN and RnD are added to the local keystore. Afterwards a VideoService from IBCN and a VCRRemote from RnD are installed. The installed services are signed and thus have access to the Security interface. When the VideoService starts, it will create a new policystore and write the default access policies to it (Figure 11). A SIP-Proxy service, installed on the gateway by provider RnD, could now request the VideoService to pause the video player when detecting an incoming phone call.

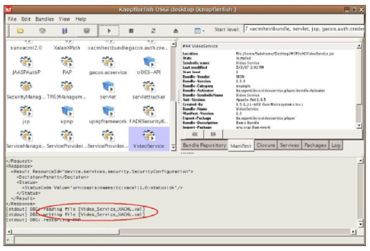
The default policies however make the VideoService initially only accessible for the VideoRemote service of IBCN. Consequently, attempts of the VCRRemote or SIP-Proxy service to use the VideoService will fail (figure 12).

The policies of the VideoService will now be remotely updated according to an offline agreement between IBCN and RnD to share their services in a collaboration scenario. Upon agreement, service provider IBCN can configure its services to be-



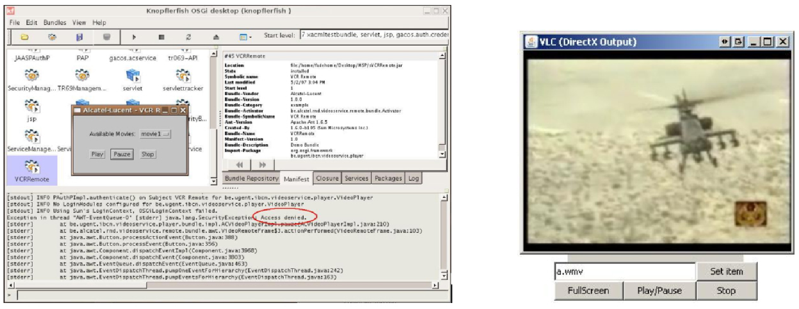
Certificate: ibcn.cert
Alias: IBCN
Keypass: ibcnpass

Do you want to install the following service:
> file:/home/fadehome/Desktop/MSP/sACVideoService.jar
of Service Provider: IBCN



```
</Request>
<Response>
  <Result ResourceId="device.services.security.SecurityConfiguration">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
  </Result>
</Response>
[stdout] DBG: reading file [Video_Service_XACML.xml]
[stdout] DBG: writing file [Video_Service_XACML.xml]
[stdout] DBG: restarting PAP
```

Fig. 11 At initial start of the trusted video service, all default policy settings are added to a central repository made available to a configuration provider via the TR-069 interface.

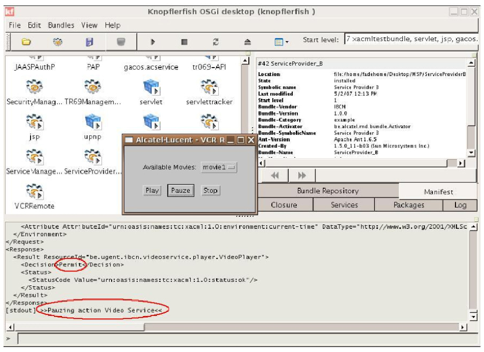


```
Last modified          5/2/07 3:04 PM
Start level            1
Bundle-Vendor          Alcatel-Lucent
Bundle-Version         1.0.0
Bundle-Category        example
Bundle-Activator       be.alcatel.rnd.videoservice.remote.bundle.Activator
Bundle-SymbolicName    VCR Remote
Ant-Version            Apache Ant 1.6.5
Created-By             1.6.0-b105 (Sun Microsystems Inc.)
Bundle-Name            VCRRemote
Manifest-Version       1.0
Bundle-Description     Demo Bundle
Import-Package         org.osgi.framework
                       be.ugent.ibcn.videoservice.player
```

```
[stdout] INFO PAuthPImpl.authenticate() on Subject VCR Remote for be.ugent.ibcn.videoservice.player.VideoPlayer
[stdout] INFO No LoginModules configured for be.ugent.ibcn.videoservice.player.VideoPlayer
[stdout] INFO Using Sun's LoginContext, OSGiLoginContext failed.
Exception in thread "AWT-EventQueue-0" [stderr] java.lang.SecurityException: Access denied.
[stderr]        at be.ugent.ibcn.videoservice.player.bundle.impl.ACVideoPlayerImpl.pauze(ACVideoPlayerImpl.java:210)
[stderr]        at be.alcatel.rnd.videoservice.remote.bundle.awt.VideoRemoteFrame$3.actionPerformed(VideoRemoteFrame.java:103)
[stderr]        at java.awt.Button.processActionEvent(Button.java:388)
[stderr]        at java.awt.Button.processEvent(Button.java:356)
[stderr]        at java.awt.Component.dispatchEventImpl(Component.java:3968)
[stderr]        at java.awt.Component.dispatchEvent(Component.java:3803)
[stderr]        at java.awt.EventQueue.dispatchEvent(EventQueue.java:463)
[stderr]        at java.awt.EventDispatchThread.pumpOneEventForHierarchy(EventDispatchThread.java:242)
[stderr]        at java.awt.EventDispatchThread.pumpEventsForHierarchy(EventDispatchThread.java:163)
```

Fig. 12 The default configuration does not allow the VCR Remote service to access the video service

come shared with RnD . From now on, noth the VCRRemote and the SIP proxy service of service provider RnD can use the pauze function of the VideoService, provided by IBCN. Figure 13 illustrates the XACML permit-decision message.

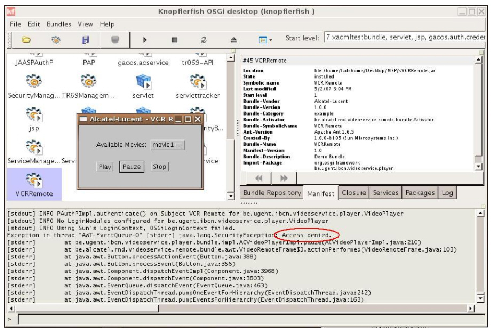


```
Bundle-Vendor          IBCN
Bundle-Version         1.0.0
Bundle-Category        example
Bundle-Activator       be.alcatel.rnd.bundle.Activator
Bundle-SymbolicName    Service Provider 3
Ant-Version            Apache Ant 1.6.5
Created-By             1.5.0_11-b03 (Sun Microsystems Inc.)
Bundle-Name            ServiceProvider_B
```

```
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time" DataType="http://www.w3.org/2001/XMLSc
  </Environment>
</Request>
<Response>
  <Result ResourceId="be.ugent.ibcn.videoservice.player.VideoPlayer">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
  </Result>
</Response>
[stdout] >>Pauzing action Video Service<<
```

Fig. 13 VCR Remote from service provider RnD is granted to pause the video service. The XACML permit-decision message is shown.

When the agreement comes to an end, IBCN can again unshare its services for RnD as illustrated in Figure 14.



#45 VCRRemote

Location	file:/home/fadehome/Desktop/MSP/sVCRRemote.jar
State	installed
Symbolic name	VCR Remote
Last modified	5/2/07 3:04 PM
Start level	1
Bundle-Vendor	Alcatel-Lucent
Bundle-Version	1.0.0
Bundle-Category	example
Bundle-Activator	be.alcatel.rnd.videoservice.remote.bundle.Activator
Bundle-SymbolicName	VCR Remote
Ant-Version	Apache Ant 1.6.5
Created-By	1.6.0-b105 (Sun Microsystems Inc.)
Bundle-Name	VCRRemote
Manifest-Version	1.0
Bundle-Description	Demo Bundle
Import-Package	org.osgi.framework be.ugent.ibcn.videoservice.player

```
[stdout] INFO PAuthPImpl.authenticate() on Subject VCR Remote for be.ugent.ibcn.videoservice.player.VideoPlayer
[stdout] INFO No LoginModules configured for be.ugent.ibcn.videoservice.player.VideoPlayer
[stdout] INFO Using Sun's LoginContext, OSGiLoginContext failed.
Exception in thread "AWT-EventQueue-0" [stderr] java.lang.SecurityException: Access denied.
[stderr]        at be.ugent.ibcn.videoservice.player.bundle.impl.ACVideoPlayerImpl.pauze(ACVideoPlayerImpl.java:210)
[stderr]        at be.alcatel.rnd.videoservice.remote.bundle.awt.VideoRemoteFrame$3.actionPerformed(VideoRemoteFrame.java:103)
[stderr]        at java.awt.Button.processActionEvent(Button.java:388)
[stderr]        at java.awt.Button.processEvent(Button.java:356)
[stderr]        at java.awt.Component.dispatchEventImpl(Component.java:3968)
[stderr]        at java.awt.Component.dispatchEvent(Component.java:3803)
[stderr]        at java.awt.EventQueue.dispatchEvent(EventQueue.java:463)
[stderr]        at java.awt.EventDispatchThread.pumpOneEventForHierarchy(EventDispatchThread.java:242)
[stderr]        at java.awt.EventDispatchThread.pumpEventsForHierarchy(EventDispatchThread.java:163)
```

Fig. 14 The video service is no longer accessible for services from provider RnD (Access Denied).

6 Conclusions

WAN Service providers need (at least limited) control on the residential network and the locally running services, to be able to offer end-to-end predictable service levels. Only then they are able to compete with local plug-and-play services which usually have a full view on the local network and its resource usage. In order to limit the view of WAN service providers on the residential network, a security framework is needed. This paper has discussed n XACML-based security framework, with two architectural options to extend the framework for multi-service provider environments. It was the aim of the security framework to allow convenient remote configuration of any service by a remote configuration provider, and to avoid conflicting service provider configuration. With a proof-of-concept implementation it is illustrated that the framework not only eases configuration of a single service, but also allows to configure in a generic and secure way collaboration scenarios between service providers. The latter implies that the security framework also allows to deny untrusted service providers to extend their view on the home network beyond the view on their own service. The framework has the additional advantage that it remains compatible with existing OSGi services by simply extending these services with a 100% automated script to add a signature and access control extensions. These scripts aim to further ease adoption of the proposed security approach by WAN service providers.

Acknowledgements This work has been funded by the Institute for the Promotion of Innovation through Science and Technology in Flanders (Belgium) within the scope of the FADE-Home project.

References

1. Haerick W., Goeminne N., Cauwel K., De Jans G., De Turck F., Dhoedt B., Demeester P., Bracke S., Acke W., Bouchat C., *Success in home service deployment: zero-touch or chaos?* Proceedings of the 44th FITCE Congress 2005, Vienna, Austria, 1-3 September 2005, pp. 36-44
2. Corcoran, P. M., *Mapping home-network appliances to TCP/IP sockets using a three-tiered home gateway architecture.* IEEE Transactions on Consumer Electronics, Aug. 1998, 44(3), pp. 729736.
3. Valtchev, D., et al., *Service Gateway Architecture for a smart home.* IEEE Communications Magazine, Apr 2002.
4. Nguyen T., Bouwen J., *The Next-Generation Residential Gateway.* The Journal of the Institution of British Telecommunications Engineers, JulySeptember 2001, 2(3), pp. 134138.
5. Home Gateway Initiative, *Home Gateway Technical Requirements: Release 1.* The http://www.homegatewayinitiative.org/publis/HGI_V1.0.pdf, July 2006.
6. OSGi Alliance *OSGi Server Platform Release 4,* October 2005
7. DSLHome Technical Workgroup *CPE WAN Management Protocol. TR-069,* May 2004
8. Universal Plug and Play Forum *UPnP Device Architecture v1.0 2002,* http://www.upnp.org