

Policy driven formation of federations between personal networks

Joachim Zeiss, Luis Sanchez and Sandford Bessler Member IEEE

Abstract—The federation concept extends the communication between trusted devices or nodes of a personal network (PN) to interactions between PNs belonging to different users. Federation formation, participation control and resource disclosure are controlled by policy rules running in each personal network. The paper describes the architecture and the challenging introduction of semantic technology in an event and messaging oriented system. The work has been performed within the Magnet Beyond IST project.

Index Terms: semantic policies, N3, personal network, context-awareness, PN federation, access and privacy rules.

I. INTRODUCTION

A personal network is a relatively new concept [1] that allows a user to transparently interconnect all his personal devices independently of their location (e.g. in the personal area, at home, at work or in his car). While Personal Networking is focused on the communication between personal devices only, many communication patterns need to extend the boundaries of the Personal Network and involve the secure interaction of multiple people having common interests for various professional and private services. Examples include: collaborative working, virtual meetings, conference, resource sharing for private and professional services, family networks, virtual classrooms and distant learning, inter-vehicle networks or emergency networks. We called this extension a federation of two or more personal networks.

The main goal of federations is to share resources such as files or services. Therefore it can be expected that security, privacy and access control play a major role in different phases of a federation life-cycle. The underlying security mechanisms are provided by the PN authentication and secure associations between nodes while access control is supported by enforcement of user defined policies.

In this sense, federations however have to consider new aspects of trust between PN users, participation and membership mechanisms in the formation phase and fine grained access control to resources of another PN, one the federation is established. Most of federations' scenarios are pervasive, in which users move from one environment to

another, change devices, so that we cannot foresee all the possible interactions between users, resources, services. Only a semantically rich representation of all the parameters can ensure a common understanding between these previously unknown entities [3]. The selected approach to be presented in the rest of the paper is based on semantic policies [6][7].

Let us first identify the need of policy rules in the following example scenarios:

- Typical examples of predefined, long-lasting federations are projects with more or less known members, and shared resources such as address books, documents, tools. (Re)-starting the federation can be associated to remote meetings, mediated by public infrastructure networks. Other useful long-lived, trusted federations can be established between the members of a family.
- Ad hoc PN federations are thought to be formed occasionally between users who may not know each other, but are in proximity of each other: This kind of scenarios pose most challenges that we try to meet using semantic web techniques.

The paper describes work in progress about the use of semantic policies to control the federation process. Our main contributions are:

- define an architecture to integrate declarative policies and a reasoning component (the policy engine) in a messaging and event based system.
- find mechanisms to group and select relevant rules, specific to the federation formation phase, request origin or user context.
- investigate the performance of Euler and CWM reasoners on mobile devices.

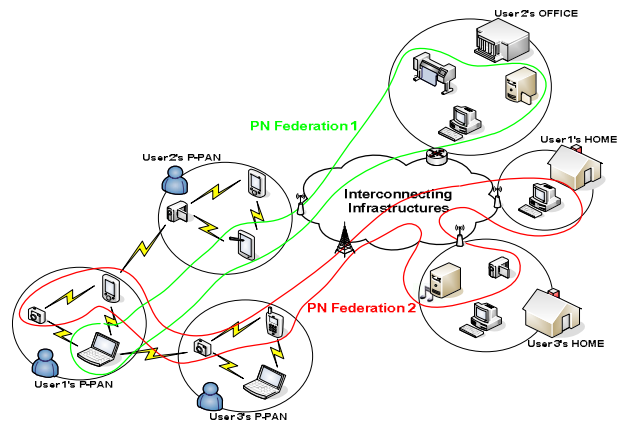


Fig. 1. PN Federation deployment scenario

The rest of the paper is organized as follows: section II presents the PN-F architecture and life cycle. The main section

Joachim Zeiss is with the Telecommunications Research Centre Vienna, Austria, e-mail: zeiss@ftw.at

Luis Sanchez is with the University of Cantabria, Spain, email: lsanchez@tmat.unican.es

Sandford Bessler is with the Telecommunications Research Centre Vienna, Austria, e-mail: bessler@ftw.at.

III describes the semantic layer and policy engine integration, whereas section IV presents first realization considerations and is followed by concluding remarks and further research plans.

II. PN-F ARCHITECTURE

A PN federation (PN-F) can be defined as a secure impromptu, situation-aware or beforehand agreed cooperation between Personal Networks of different people for the purpose of achieving a common goal or service by forming an efficient collaboration.

A. PN-F life cycle

The PN federation is a challenging concept, involving major research challenges related to the definition and management of the federations, service profiles and discovery, security, context and networking. We shortly introduce the three identified phases, as different policy rules have to be checked in each phase.

PN-F Participation: the objective of this phase is the agreement between the PN-F Creator (i.e. the PN that starts the PN-F) and one candidate PN-F Member (a PN aiming at participating on the PN-F). The basis for this agreement is the secure exchange of different PN-F profiles.

This phase starts when the Creator advertises the public part of the *PN-F Profile* containing basically the goal/purpose of the federation together with the necessary information to identify the federation and to address its Creator (represented by its Federation Manager, FM). Additionally, it can also contain the PN-F formation rules and the list of PN-F invitees (does not have to actually correspond with the PN-F

members). Two possible ways of advertising the PN-F has been identified, Invitation-based (i.e. the Creator takes the initiative) or Publish-based (the Candidates ask for joining). Once the Candidate is aware of the PN-F (i.e. it has received the public part of the PN-F Profile), and after a security association has been established between both PNs in order to authenticate and protect the interaction, it will conform its *PN-F Participation Profile*, mainly consisting on the resources that it makes available to this federation, and securely sends it to the Creator. The Creator then checks whether the Candidate fulfils the federation policies and if this is the case, the private part of the PN-F Profile, consisting on the complete list of PN-F Members and a group key is securely forwarded to the new PN-F Member.

PN-F Formation: in this phase, the communication means between any two PN-F Members are deployed. The main issues solved during this phase are addressing, routing and security.

PN-F Use: This phase comprises both the discovery and provision of services offered to other PN Members. The authenticity provided by the networking solutions implemented below the PN-F service level allows the deployment of access policies for the shared resources.

B. PN-F Architecture and functional entities

Fig 2 shows the generic architecture of a PN-F together with its main functional entities and the relations between them for PN-F formation.

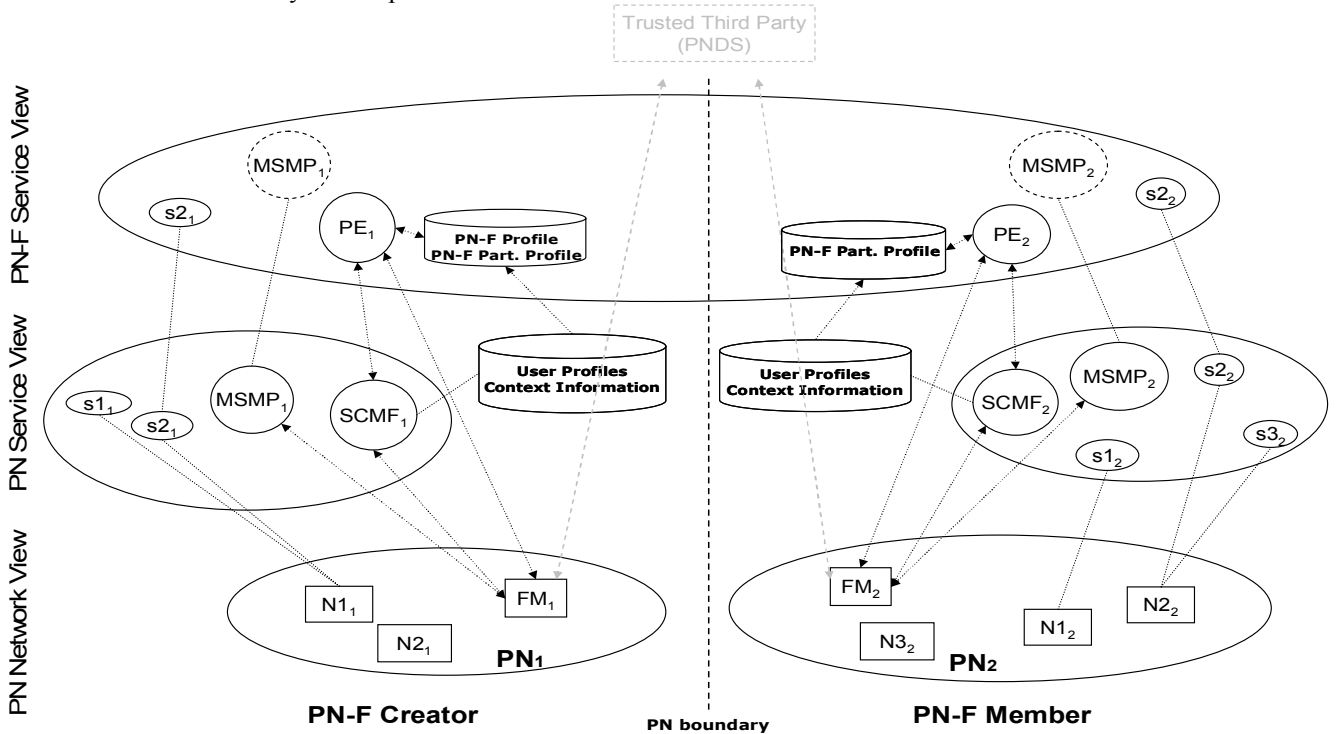


Fig. 2. PN-F Architecture and main functional entities

These functional entities are:

- Federation Manager (FM): It is responsible for managing

all the interactions between two PNs during the PN-F Participation phase, mainly focused on the exchange of

PN-F profiles. It controls the respective PN state on the PN-F state machine.

- Secure Context Management Framework (SCMF): It is a distributed framework that provides access to all the PN related context information. One of its responsibilities is to store the different profiles from the PN-Fs that the corresponding PN is involved in.
- MAGNET Service Management Platform (MSMP): It controls the service discovery and access within the PN. The FM will interact with it when the PN-F Participation profile is to be created.
- Policy Engine (PE): Acts as an interpreter and reasoner of the rules declared on the PN-F profiles. When a service is requested under the auspices of a PN-F, it takes the rules from the different PN-F profiles plus any other required context information to enable access control enforcement.
- Personal Network Directory Service (PNDS): Takes the role of a trusted third party. It can be used to verify the identity of the peer PN on the different phases of the PN-F and it can also host the publication of PN-Fs' and PNs' details.

III. SEMANTIC POLICY LAYER

A. Policy languages

Today's identity-, role- and certificate-based access control scheme are limited to static situations and known principals.

The authorization decisions are performed mostly in a central decision point (PDP) using centrally managed policies. Once the access is granted, no conditions are posed during the usage and after the usage of service and data. Privacy and access control are handled differently, although the same language and mechanisms could be used.

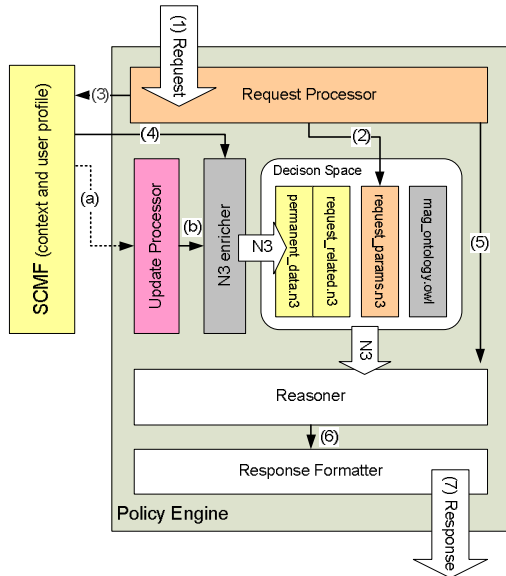


Fig. 3. Architecture of the police engine

In case of context aware services, this could be handled differently [4], since both the client and the service would need an evaluation engine and policies: the former - to disclose its context data, the latter to perform access control.

The situation is similar for personal networks engaged in a

federation: the decisions and the policies are distributed.

Considering the languages used to express various policies, we distinguish between markup languages such as XACML and P3P and languages with ontology support. We have selected the language Notation3 or N3 [5] - a shorthand easy readable non-XML serialization of RDF models. Its strength lies in the ability to reason over easily definable formulae which might include build-in logical, mathematical, time of day interpreting or string manipulation functions. N3 makes it easy to implement a distributed freely extensible policy system. Complex relationships can be defined in a simple way. N3 policies can be applied to many aspects of service negotiation, access and usage. Its underlying semantic technology allows a declarative, flexible and dynamic definition and enforcement of policies at runtime. N3 tremendously reduces the cost of parsing definitions on mobile clients as it does not require an XML parser. It reduces footprint of the policy engine and speeds up processing.

B. Architecture of the Policy Engine

Since most of the network and service components of the PN are message and event based, whereas the N3 reasoning is declarative, we introduce in the ontology the “request” and “result” classes, similar to the REI language [6]. Thus, the policy engine becomes a stateless request response based server that deals with any kind of requests expressed in N3. In order to select from a large number of rules, those that are relevant to the application, we provide a semantic definition in the request itself – yes, the request might describe itself – and a decision space for the policy engine (described in the next section). The request might even describe the response it is expecting by using N3 rules. The incoming request and the local context management system (SCMF) are the only data sources. Fig. 3 describes the Policy Engine components:

- The Request Processor reads the request (1) and inserts them into the decision space. It queries the SCMF (3) for required context, user profile and policy data.
- The Update Processor receives update events on context and user profile information kept permanently in the decision space.
- The N3 Enricher transforms plain xml formatted output of SCMF responses and notifications into N3.
- The Decision Space consists of files or N3 String objects containing the Magnet ontology in OWL plus instances of context, user profile and policy data related to the user.
- The Reasoner is a N3 rule engine. It is invoked with all data found in the decision space. It returns N3 triples.
- The Response Formatter transforms the N3 results.

C. Rule processing description

Within the federation architecture (Fig. 2) the policy engine (PE) is being used by the FM, to control federation formation and usage, and by the security module (CASM) of the SCMF, to control external access to context and user profile data.

The following N3 declarations illustrate the rules for a scenario, where a user (i.e. a federation member) :joachim asks to join the federation :myFed. The FM of the creator passes this JoinRequest to the PE. If the latter answers with

“<>:result :granted”, then, joining the federation is allowed.

The join request has two mandatory fields to denote the requestor and the federation he would like to join:

```
:aRequest a :JoinRequest;
      :has_requestor :joachim;
      :for_federation :myFed.
```

Additionally, optional context or profile data that are not yet accessible via SCMF queries might be added. The federation is defined by the creator in a PN-F profile which contains information about participating members and policies that apply during creation and in-use phases. For the following example, location, friendship relation and proximity policies are assumed.

Before the rules themselves are applied, the roles of the sender and receiver are mapped to the three message exchange possibilities: creator-member, member-creator and member-member (see Fig. 4). Additionally, federation member checks are applied.

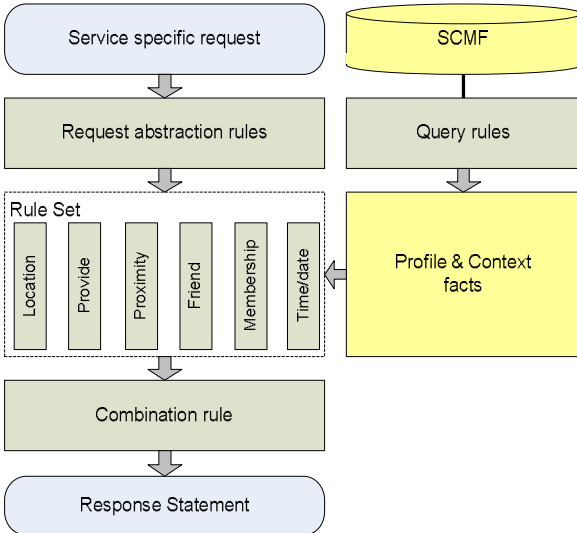


Fig. 4. Internal processing of requests and rules

The request abstraction mechanism maps to the generic roles of a sender and receiver understood by any elementary policy rule:

```
{ ?req :has_requestor ?member.
  ?req :for_federation ?pnf.
  ?pnf :has_member ?member
} => {<> :sender ?member}.
```

```
{ ?req :for_federation ?pnf.
  ?pnf :has_creator ?creator
} => {<> :receiver ?creator}.
```

In N3, “?” preceded identifiers are universal quantifiers and the expression “{A. B. C} => {D}” means $A \wedge B \wedge C$ implies D. To build a policy, we need the following elements (Fig 4):

- A set of elementary rules in the combination rule
- A query rule, telling the PE what data to obtain from the context system to be able to fulfil the request.
- A combination rule

The elementary rules define conditions to be met by context and user profile information. These policy elements are held in a set of rules which might be combined to fulfil the particular needs of a policy. Currently we have defined six elementary

rule sets as shown in Fig 4. If a request should be granted or denied might be determined by the location of any of the parties, by the proximity to each other, by being a friend or member in a certain group, may depend on services one is willing to offer, or on date and time information.

In the following, we illustrate some of these rules. The location rule checks location matches for sender or receiver:

```
{ <> :sender ?s. <> :receiver ?r. <> :rule ?p.
  ?p a :LocationPolicy. ?p a ?q.
  ?s :has_location ?ml.
  ?ml :at_x ?mx. ?ml :at_y ?my.
  ?r :has_location ?cl.
  ?cl :at_x ?cx. ?cl :at_y ?cy.
  ?mx math:equalTo ?cx. ?my math:equalTo ?cy
} => {?s :complies_to ?q}.
```

The proximity rule checks whether sender and receiver are near:

```
{ <> :sender ?s. <> :receiver ?r. <> :rule ?p.
  ?p a :ProximityPolicy. ?p a ?q.
  ?s :near ?r
} => {?s :complies_to ?q}.
```

The friend rule checks the friend list (address book or other buddy list) of the creator for its relationship to the requestor:

```
{ <> :sender ?s. <> :receiver ?r. <> :rule ?p.
  ?p a :FriendPolicy. ?p a ?q.
  ?s foaf:knows ?r
} => {?s :complies_to ?q}.
```

Finally, the combination rule combines the rules above. A “:complies_to” triple determines that the related policy must be met. If a policy is missing in this formula it does not need to be fulfilled. The formula produces the final result of the request:

```
{ <> :sender ?s.
  ?s :complies_to :ProximityPolicy.
  ?s :complies_to :FriendPolicy.
  ?s :complies_to :LocationPolicy
} => {<> :result :granted}.
```

The desired response output expressed as reasoning goals is defined as follows:

```
{<> :sender ?member } => [].
{<> :receiver ?creator } => [].
{<> :result :granted } => [].
{?s :complies_to ?p } => [].
```

These statements could come along with the request. Hence, the requester may determine verbosity, content and logical details of the response. She might ask to include meta policy information which tell how to deal with missing information or contradicting information. Current meta policy definitions are:

```
:metaPol a :MetaPolicy;
      :modality :positive;
      :resolve_conflict :deny_preceeds;
      :default :denied.
```

The :modality informs if a request will be :granted (:positive) or :denied (:negative). In case :granted and :denied occur at the same time, the :resolve_conflict specifies whether the deny information overrides the grant information or vice versa. The :default predicate tells if a request should be granted or denied if no information is available.

The actual response to the join request in our example is:

```
<> :sender :joachim. <> :receiver :sandford.
:joachim :complies_to :LocationPolicy.
```

```
:joachim :complies_to :ProximityPolicy.  
:joachim :complies_to :FriendPolicy.  
<> :result :granted.
```

The expression <> represents the N3 document itself, in our case all of the N3 triples evaluated. The last line of the output above is the important one. It reads: “The request has the result granted.” According to the meta-policy this request is supposed to be denied if this line is missing. Any information which is part of the context data could be added to the response. The policies defined by the federation creator could add rules to do so. Furthermore, the answer might contain obligations or declinations as terms of usage.

A special type of rule is the query rule that is created when a new policy rule is submitted to the system. In this case the policy engine evaluates the new policy and determines the queries to be issued towards the SCMF to obtain the necessary context and user profile data. The required combination and query rules are stored in the user profile and obtained by the PE ahead of invoking the reasoner.

D. Semantic fortification of context and profile data

In an ideal world all context and user profile data would exist in terms of semantic web definitions, both classes and instances. However, for pragmatic reasons, this is not the case in Magnet Beyond. All the context and user profile data stored in the SCMF needs to be queried. The query result is represented in plain XML. Data that rarely changes can be retrieved and enhanced with semantic meta-information by the PE. The Update processor (see Fig. 3) sets notification triggers on value changes making sure that data inside the decision space of the PE remains consistent. These data are user name PN ID or address information. Other data changes frequently and needs to be retrieved and semantically enhanced every time they become subject to a policy request. These data are location or proximity of a user.

IV. REALIZATION ASPECTS

For the realisation of the reasoning component different implementations are available, namely CWM, Pychinko and Euler. The CWM is a fully-fledged N3 rules engine, whereas Pychinko and Euler implement only a subset of the N3 language. CWM is significantly slower than the lightweight implementations Pychinko and Euler. The latter produces only “N3” output and cannot recursively evaluate semantics of data sources defined in an N3 statement or form. All rule engines are available in Python. Euler is also available in Java, C# and Prolog. As the PE is supposed to run on a mobile client, we have to make a trade-off between semantic expressiveness against performance. Therefore, CWM can only be used for pre-processing RDF complex N3 files on a desktop machine. Euler has been selected for the PE realization since it has the smallest footprint (pychinko needs to install CWM modules as well) and is available in many languages. For simple N3 statement processing the python implementation of Euler has proved to be 10 times faster than CWM on a Nokia 770.

V. CONCLUDING REMARKS AND FUTURE WORK

The concept of a federation between personal networks involves semiautomatic interactions between users. We think that in this environment, especially in ad hoc scenarios, policies are the right approach to solve privacy and fine grained access control to PN services and resources. More than that, the nature of personal networks requires a symmetric, distributed system of reasoners and policies instead of legacy centralized decision points.

While the first integration of simple reasoners such as Euler provides optimistic results concerning performance, there are still a number of challenges to be met:

- The probably biggest challenge is to find a simple way for non-expert users to create and manage their own rules. The users have to understand the effects of their decisions.
- Since we deal in Magnet Beyond with a powerful context management system, we want to further explore ways by which context can make the policies more flexible.
- To investigate the scalability of the PE through its use in other scenarios besides PN federation: for example the context aware service scenario discussed in [4] requires the local PE to check context disclosure policies against the remote service access policy. The scalability of the PE design requires that several federations and context-aware service invocation coexist and that the rule selection works even with a large number of policies.

ACKNOWLEDGMENT

This work is partially funded by the IST Project Magnet Beyond. MAGNET Beyond will develop user centric business model concepts for secure Personal Networks in multi-network, multi-device, and multi-user environments. MAGNET Beyond has 32 partners from 15 countries, among these highly influential Industrial Partners, Universities, Research Centres, and SMEs. www.istmagnet.org

REFERENCES

- [1] I.G.M.M. Niemegeers, S.M. Heemstra de Groot, “FEDNETS: Context-aware Ad-hoc Network Federations”, International Journal on Wireless Personal Communications, Vol. 33, June 2005, pp. 319-325.
- [2] IST Magnet, <http://www.istmagnet.org>
- [3] A. Toninelli, R. Montanari, L. Kagal, O. Lassila, A semantic context-aware access control framework for secure collaborations in pervasive computing environments, Proceedings ISWC 2006, pp 473-486.
- [4] S. Bessler, J. Zeiss, Semantic Modelling of Policies for context-aware services, WWRP 17, Heidelberg, 2006
- [5] T.B. Lee N3 Language Tutorial, <http://www.w3.org/2000/10/swap/doc/>
- [6] L. Kagal and Tim Berners Lee, Rein: Where policies meet rules in the semantic web, Technical report, MIT, 2005.
- [7] V. Kolovski, Yarden Katz, James Hendler, Daniel Weitzner and Tim Berners Lee, Towards a Policy-Aware Web, Semantic Web and PolicyWorkshop, Galway, Ireland, 2005.