A Dynamic VPN management architecture for Personal Networks[1]

Wassef Louati and Djamal Zeghlache
Institut National des Télécommunications
Wireless Networks and Multimedia Services Department
9 rue Charles Fourier - 91011 EVRY CEDEX - France
Emails: { wassef.louati, djamal.zeghlache }@int-evry.fr

***Abstract*— This paper proposes a Dynamic Personal VPN management architecture to allow Personal Network (PN) users to initiate establishment of tunnels for PN networking and management of VPN services and parameters on demand. In this architecture, a service-oriented policy-based VPN establishment and a network-based VPN model are adopted. The VPN management plane interacts with naming and service discovery frameworks to handle service location, mobility and achieve configuration flexibility. A number of frameworks have been selected for their flexibility and extensibility to conduct the analysis and design an overall architecture for VPN based PN networking. This paper combines an Intentional Naming System (INS) used to resolve names and assist mobility and the Netconf protocol acting as a management protocol. An AAA infrastructure provides access control and accounting. Results of an implementation and a performance evaluation of the proposed architecture are reported.**

***Index Terms*—Personal Networks, Dynamic VPN, Management plane, INS.**

I. INTRODUCTION

The *Personal Network* (PN) (figure 1) concept extends the PAN (Personal Area Network) paradigm by encompassing remote personal devices and nodes in the network architecture. It is the subject of wide research [1] and is at the heart of ubiquitous computing. A PN is composed of a *Private PAN* (P-PAN), which is a bubble surrounding the *PN user* and characterized by a common ownership (e.g. PDA, mobile phone, etc.), and a set of *clusters* located outside of the P-PAN. A cluster is a network of devices and nodes (i.e. devices implementing IP protocol) characterized by a common trust relationship and located within a limited geographical area (e.g. home cluster, office cluster, etc.). Each cluster has a number of gateway nodes that enable connectivity to the outside world. These gateways can include functions such as NATing and Firewalling. From a network point of view, a PN can be viewed as a collection of geographically scattered clusters (the P-PAN is a special cluster) which can communicate via *infrastructure networks* (such as Internet, UMTS, WLAN, etc.) and via ad hoc segments. In this paper we will focus on IP infrastructure networks. From a *service* point of view, a PN is perceived as the collection of available public and private services provided by personal and foreign (i.e. not personal) devices reachable by the PN user (figure 2).

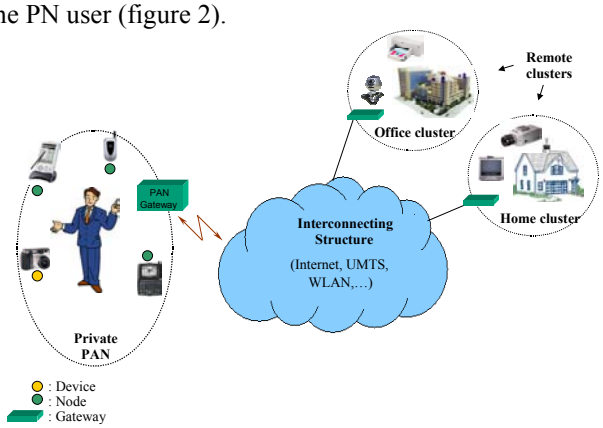

Fig. 1. The Personal Network concept

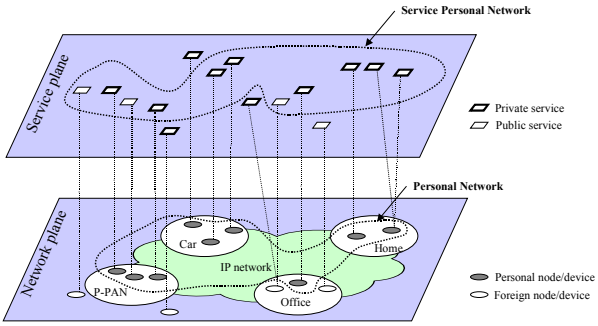

Fig. 2. PN plane architecture

Since *Virtual Private Networks* (VPN) have been extensively used to provide a way to transfer secure and private data traffic using the public infrastructure networks, they represent an efficient solution to secure inter-cluster communication for PNs. *Dynamic VPN* can dynamically adapt the VPN parameters according to context such as the VPN user needs and location. Since PN clusters, nodes and services face a constantly changing environment and highly dynamic configuration according to context (time, place, circumstances, etc.), dynamic VPN *establishment, tear down and maintenance* capability appears as a requirement for PN. This is actually the scope of the reported study in this paper exploring how dynamic VPNs can support PN services and PN users. More specifically, the objective is to propose a *Dynamic Personal VPN* (DP-VPN) management architecture satisfying PN requirements such as *mobility* support, *security* and *flexibility*.

[1] This work was partially funded by IST Integrated Project MAGNET.

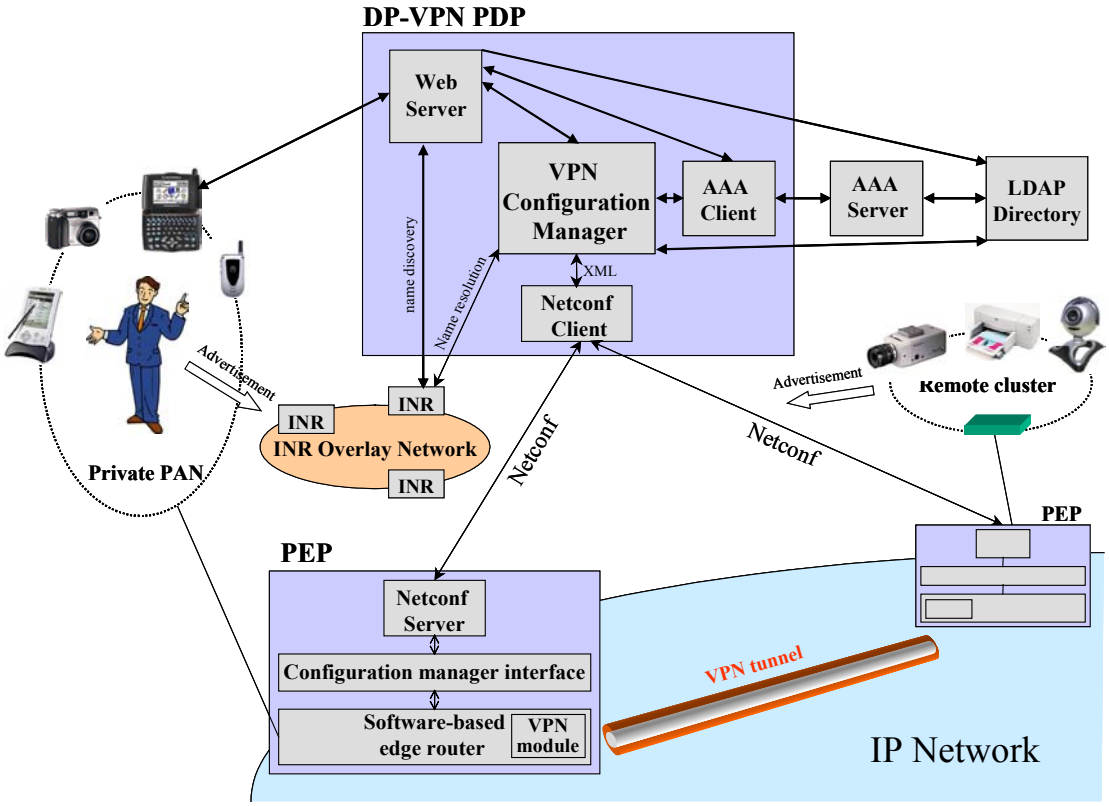

Fig. 3. The DP-VPN architecture

To achieve this goal the PN architecture should include several protocols and frameworks and facilitate interaction between architecture components.

The VPN management plane will interact with a flexible naming system handling mobility, service discovery and service location. In order to meet the requirements of flexibility and ease of use, the Intentional Naming System (INS) [2] was selected. The Netconf management protocol [3], supporting exchange of configuration information between VPN management and networking, is adopted in this work. An AAA infrastructure is employed to provide access control and accounting within the PN. A service-oriented policy-based management approach and a network-based VPN model are adopted.

The organization of this paper is as follows. Section 2 describes the DP-VPN architecture, its components and the DP-VPN establishment request mechanisms. A scenario of DP-VPN applicability is demonstrated in section 3. Finally, the results of an implementation and a performance evaluation of the architecture are reported in section 4.

II. The Dynamic Personal VPN architecture

A. *Architecture overview*

Dynamic Personal VPN places the VPN services management under the control of the PN user and provides support for PN cluster/node/service mobility. The approach consists of empowering the PN user with some VPN management responsibilities in agreement with the VPN service provider. A Service Level Agreement (SLA) is established to determine which parameters and actions the PN user can control and manage. Management of VPN services includes creating, editing and deleting services and their related parameters (such as security, QoS, source/destination of flows, time duration of tunnels, etc). This can be achieved by including a Dynamic-VPN technology in the service provider network such as in [4]. One example of such VPN services is the dynamic bandwidth management service found in [5] that allows PN users to dynamically request changes of bandwidth allocated to their VPNs.

The mobility scope of this paper is limited to large time scales. The contribution addresses mostly low mobility cases and does not cover micro-mobility challenges in PNs. It is assumed that changes in PN node or service addresses (or precise locations) and changes in edge routers (or cluster point of attachment) are handled by mobility management and fast forwarding. The management plane will however interact with an adequate naming system that provides discovery services, flexibility and expressiveness for describing devices and nodes and reduce the need for direct handling of addresses and precise location information.

Figure 3 depicts a Dynamic VPN management architecture for Personal Networks based on policies. The architecture includes a DP-VPN Policy Decision Point (PDP) and Policy Enforcement Points (PEPs) [6]. The PDP acting as a DP-VPN manager is part of the VPN service provider whereas the PEPs are generally the routers that will be automatically configured by the DP-VPN PDP to support the VPN management. The Netconf protocol [3] is used for communication between PDP and PEP. For simplicity, the figure has been structured in

blocks representing the key functional modules (DP-VPN PDPs and PEPs). To interact with the providers, users rely on a Web Server providing a Web-based user interface for DP-VPN services and resources discovery. The *VPN Configuration Manager* (VPN CM) is the main component of the PDP and ensures the management of the VPN configuration rules and their transmission to the appropriate routers (or PEPs). The VPN CM has a Web interface support for configuring and managing VPNs. This support can be provided by a Web Server that can be accessed by a node in the Private PAN supporting a Web explorer (e.g. PDA, laptop, GPRS phone, etc.).

A Lightweight Directory Access Protocol (LDAP) in the architecture provides directory-enabled management of the VPN profiles.

Adopting the end-to-end VPN model requires the integration of a VPN gateway in a capable cluster node. This is generally not very feasible or practical since these gateways are often mobile and/or may not support the required functionality. The access link between the gateways and the Service Provider edge routers are not directly manageable. This favors the adoption of network-based VPN technologies in the DP-VPN architecture (i.e. VPN tunnels are initiated and terminated at the Provider *edge routers*). This assumes that a secure connection between the gateways and the edge routers is possible.

In the DP-VPN architecture, a policy-based VPN connectivity is adopted to configure the routers with VPN parameters. Establishment of network-based VPN between two PN nodes will be based on security policies relying on service names in the naming system rather than specific IP addresses. The VPN policies are represented basing on the Policy Core Information Model (PCIM) [7], and will be expressed in eXtensible Mark-up Language (XML).

Prior to, enforcement of the policies in PEPs, the VPN CM acts as a policy server that transforms policies into the correct policy format in an overall *name-to-address resolution* operation. If the IP address of a node changes, the PN user or the VPN administrator would not have to change manually the policies.

The architecture provides also *edge router discovery* through the naming system that is used to find the two edge routers involved in given VPN when networking PN nodes. Following discovery, the VPN CM can configure each edge router with specific VPN policies and appropriate security keys and security associations. In other words, edge discovery will define the end points of the tunnels to be set up by the VPN CM. This approach, based on edge routers, should scale well as each router can have its own private policies and security association entries.

The choice of *software-based* routers in this architecture is due to their flexibility to implement interfaces and entities such as configuration interfaces and VPN establishment modules.

B. Naming and resource discovery in the architecture

As names can be organized independently of addresses, PN architectures can benefit from using naming systems to describe PN devices and nodes. Names would be used as temporary substitutes to addresses and the naming system would be updated in case of node or service mobility. Appropriate naming system for PN should include resource discovery, service location and configuration flexibility. The Intentional Naming system [2] meets these requirements and has been selected in this work to provide the functions needed for PN networking.

1) INS overview

INS is a naming, resource discovery and service location system. It is an expressive, responsive and robust system for dynamic and mobile networks of devices. Since *intentional names* are resource proprieties descriptions, applications use them to describe what they are looking for (i.e., their intent), not where to find objets (i.e., not hostnames). INS implements intentional names using expressions called *name-specifiers*. The two main concepts of the name-specifier are the attribute and the value. An attribute is a category in which an object can be classified, for example its 'color'. A value is the object's classification within that category, for example, 'green'. Together, an attribute and its associated value form an *attribute-value pair*. Name-specifiers are a hierarchical arrangement of attribute-value pairs in a tree to allow for more flexibility in describing intent. Figure 4 shows an example of a name-specifier that describes a public-access digital camera, whose resolution is 640*480 and located in the second floor of some private premise. The text representation of this example is:

[service=camera [type=digital] [resolution=640*480]] [accessibility=public] [location=INT [floor=2]].

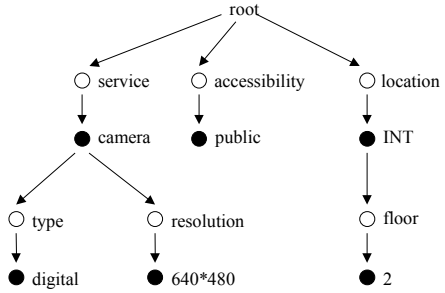

Fig. 4. A graphical view of an example name-specifier

The INS architecture (see figure 5) is composed of *INS applications* and an *Intentional Naming Resolver* (INR) network. The INR is an entity that routes client requests to the appropriate locations by maintaining a mapping between resource names (service descriptors) and their locations in the network. This mapping is stored in a data structure called a name-tree. Any PN node or infrastructure node can potentially act as a resolver. The INRs self-configure into an application-level distributed *overlay network* to exchange name information, to be accessed later from any resolver. The INRs are populated with name information and data through INS applications that can be clients or services. Clients use an intentional name, rather than an IP address, to request a service without having to worry about how the end-nodes that serve the request are connected. INS applications can

communicate with the INR overlay in the following ways:

Advertisement and Discovery: Each INS service advertises periodically to its INR neighbor, its intentional name and its associated name-information records such as IP address of the node providing the service. The INR propagates the information to its neighbors. For resource discovery, a client can send a name query to an INR to learn about matching names that exist in the network.

Early binding: A client can ask an INR to resolve an intentional name to its network location (IP address), like in the Domain Name System (DNS) where access point names are resolved into host addresses. The application can then send data directly to the network location using the resolved destination node address during the session. Note that INS includes also the late binding functionality (sending immediately data to an intentional name). Late binding is not addressed in this paper and the proposed DP-VPN architecture.

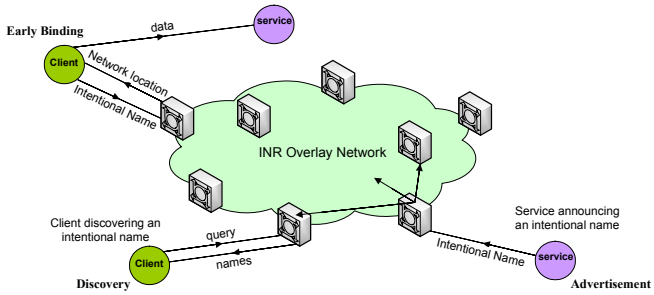

Fig. 5. INS architecture

2) Role of INS in the DP-VPN architecture

The INS, resource and service discovery reside in the service plane and can assist the dynamic and distributed establishment of DP-VPN. The actual tunneling, routing and VPN mechanisms are part of the network level. INS can enable establishment of DP-VPNs without the need to specify explicitly tunnel endpoints to set up the desired connectivity. It suffices to name the device, node, service or application and send a name resolution request to the naming system that shall return the corresponding IP address.

Assuming that the INS is an integral part of PN devices, there are several benefits of using INS to support the DP-VPN architecture:

a) Naming

Besides using names to identify devices and nodes, INS can use *service names* to describe a specific service offered by a device. PN device and service names should be unique within the PN space to avoid any internal conflicts.

To scale the system, INS proposes to partition the namespace into several *virtual spaces*. More formally, a virtual space is defined as an application-specified set of names that share some attributes in common. In other words, it is a logical grouping of services. In the context of Personal Networks, all devices and services in a PN could be considered as a virtual space. The virtual space could play the role of the PN identifier in the global network. This identifier can be obtained from a third authority for users to identify their PN using a unique global identifier. INRs involved in a personal network (i.e. using the PN virtual space) would form an overlay sub-network. The virtual space is designated as 'vspace' in the paper and used in all representations of intentional names.

INS/Twine [8] is a scalability improvement over INS that consists of partitioning resource descriptions among a subset of INRs. INS/Twine is recommended for large-scale wide-area resource discovery and consequently for the DP-VPN architecture. INS/Twine does not require the use of virtual spaces anymore but appending a simple attribute to names could identify the PN to which nodes and clusters belong.

b) Expressiveness

INS enables the PN user to express the source and destination endpoints of the VPN using intentional names that describe the intent of the application (with partial descriptions sometimes accepted), rather than specific endpoints (IP addresses). Since INS provides an expressive service description, which gives adequate flexibility to clients in searching an appropriate service, the PN user is not compelled to know the IP addresses of the endpoints. When those addresses are queried via names, the INR can resolve the names and send back the address information to the querying entity.

c) Resource name information

The PN user (playing the role of an INS client) can find out, via a Web Interface, related to an INR, which resources are around and available either locally or remotely. For example, a remote PN user 'personX' can discover the names of all devices located in his home by requesting the names matching to the query [cluster=home][vspace= personX]. The user makes use of the camera name to trigger or request establishment of the VPN.

PN users can also query more name information records about the services, other than IP addresses, such as transport/application protocols and transport port numbers. Such information can be useful to the PN user who would like to control or set VPN parameters for a dynamic service (e.g. dynamic bandwidth management service).

Note that if a PN node has a local private address, it should advertise the names of its services with the global routable address assigned by the cluster NAT. The INS application advertising the names must obtain this information (i.e. the corresponded global address to the local address of the node) from the NAT lookup table.

d) Edge discovery

Assuming that programmable and open edge routers are available and can provide services for the PN, INS applications can be integrated in these edge routers. This can be useful for discovering edge routers that are involved in the VPN establishment. When a cluster connects to an edge router, following resource and service discovery, the gateway

can pass its *cluster name* composed of the cluster identifier and the PN identifier (e.g. [cluster=office] [vspace=personX]) to the INS application located in the edge router. The incoming cluster name is concatenated with the edge IP address in a name record that will be advertised by the INS application to the INR network (Figure 6).

A PN device/service name includes normally the cluster name. Once a cluster is attached to the interconnecting structure, the cluster nodes can announce their names to the INR network. When two nodes from different clusters want to communicate through a VPN, the VPN CM examines their names, deduce the cluster name (figure 7), and consequently the edge routers involved in this communication by asking an INR in an early binding operation to resolve the names into IP addresses.

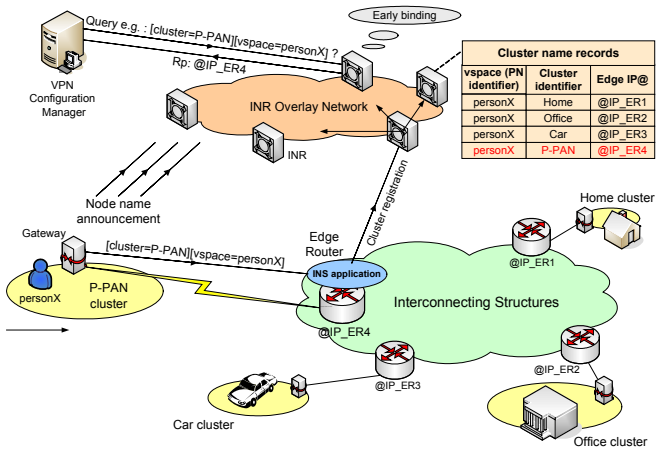

Fig. 6. Cluster registration and edge discovery

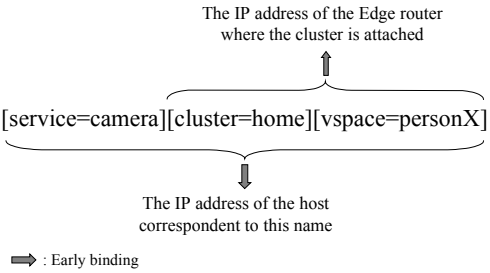

Fig. 7. PN node locating process

e) Mobility

The proposed framework is capable of assisting mobility management in the PN architecture. When a cluster moves, it passes its name to the new visited edge router which will announce this name to the naming system. Therefore, the mapping between the cluster name and its associated (serving) edge router IP address will be updated.

In addition to the cluster mobility support, the architecture handles *host and service mobility* since intentional names are included in the VPN rules.

For example, a VPN rule, which is applied to the flow coming from the home camera in Marry's PN and going to her PDA, will take the following format:

IF { (source)[service=camera][cluster=home][vspace=Marry] and (destination)[service=PDA][cluster=P-PAN] [vspace=Marry] } THEN { Encrypt and Tunnel }

Therefore, expressing the flow's source and destination by an intentional name will make them independent of their locations (reachable through a name rather than a specific endpoint). Indeed, when the source or destination host moves, it announces from its new location its intentional name (and its new associated address) to its new INR neighbor. This information is immediately propagated through the INR network. When the VPN rule is applied or enforced again, the names will be resolved to their updated IP addresses.

This can also assist service mobility. A service may need to move from one node to another to follow a PN user. For example, a PN user who receives usually a home camera transmission on his laptop through a DP-VPN, can elect to receive it on his PDA rather than his laptop by announcing a similar name than the one announced by the laptop (e.g. [service=camera] [entity=receiver] [cluster=P-PAN] [vspace =personX]).

C. Management and access control protocols

1) Management protocol: Netconf

The dynamic aspects of establishment, maintenance, update, and tear down of VPN tunnels are controlled via management protocols requiring efficient and standardized exchange of control data. Since the use of XML-based management is strongly recommended nowadays for portability across heterogeneous platforms, particularly for software-based ones providing easy XML manipulation, this work has focussed on an XML-based solution.

Netconf was developed by the IETF Network Configuration working group [3] chartered with the standardization of a new protocol using XML technologies for the configuration management of IP based network devices. The protocol uses a simple RPC-based (Remote Procedure Call) communication paradigm between a client and a server. RPCs, encoded in XML, are sent from a client to a server. The server executes the called function and returns the result. Typically, the client is an application running as part of a network manager and the server is part of a network device.

In the context of our architecture, the PDP includes the Netconf client. The server is part of the PEP. The communication path between the client and the server is build on top of the Blocks Extensible Exchange Protocol (BEEP) [9], an application protocol for XML transport. As a peer-to-peer protocol, BEEP provides an easy way to implement Netconf. BEEP can provide transport security to secure Netconf communications. This is an important feature since VPN-specific configuration information would be communicated by Netconf.

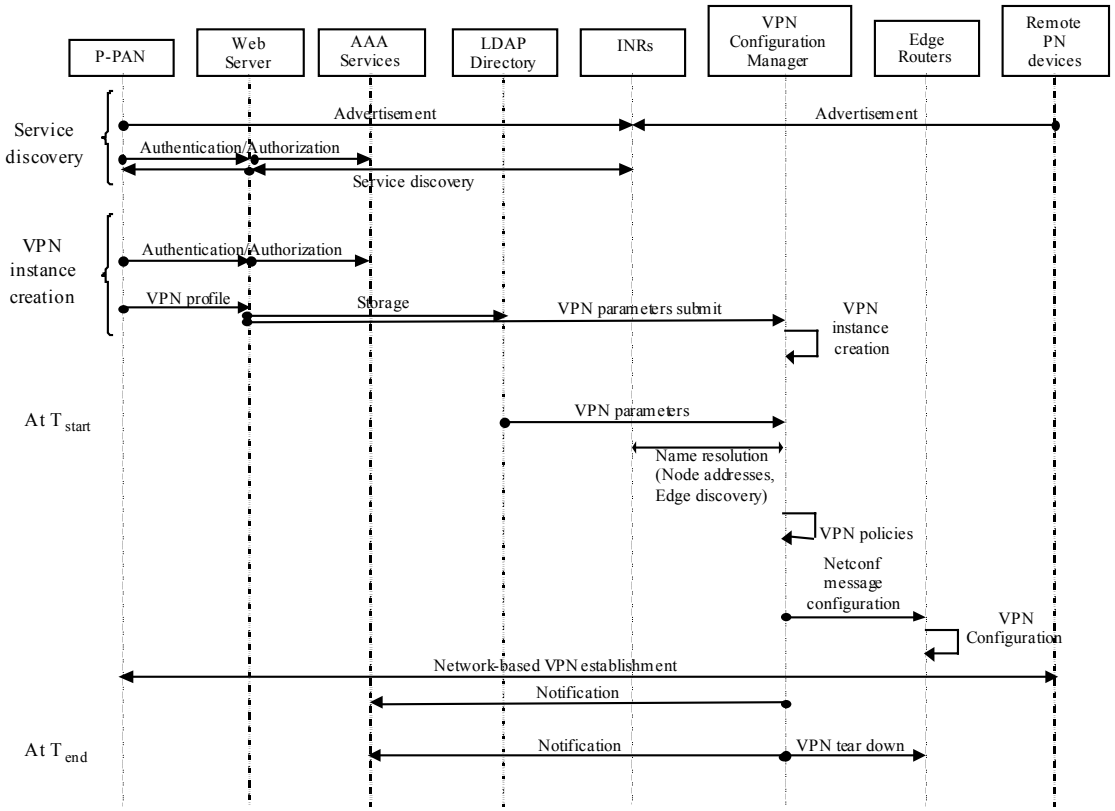

Fig. 8. The scenario diagram of the DP-VPN establishment

2) AAA services

An AAA infrastructure (Authentication Authorization Accounting) is required in the DP-VPN architecture. The user must be authenticated before allowed to learn about their PN resources or access dynamic VPN services. Once PN access is granted, allowed VPN services and parameters must be indicated to the users and information on their usage must be collected. The LDAP Directory can be used to keep track of user authentication information (e.g. user ID and passwords), authorization information (e.g. SLA parameters) and accounting information of the VPN service to be delivered. The use of resources of other PNs increases the importance of AAA services.

In the AAA infrastructure, a client/server model is employed. The clients are nodes that perform network access control and the servers are nodes that handle AAA requests. An AAA protocol is used for the exchange of authentication, authorization and accounting data between client and server. Diameter [10] is one suitable protocol that can be extended to an AAA application for LDAP.

To perform PN access control, the AAA client requests authentication and authorization from an AAA server related to the LDAP Directory. The server is also responsible for storage, analysis and administration of reported measurement information. In order to support usage-based billing, the PDP notifies the AAA server when a VPN service is activated or deactivated. The VPN parameters are also relayed to modules responsible for dynamic QoS based accounting.

D. Dynamic VPN request processing

Figure 8 shows the scenario of the Dynamic Personal VPN establishment. First, PN services and devices periodically advertise their intentional names to the INR network to describe what they provide. Those resources can be discovered anywhere by the PN user by consulting a Web Interface related to an INR.

The Web Server is also accessed by the PN user to create VPNs. The AAA server verifies the PN user's credentials in order to authenticate and authorize access to VPN services. Once the PN user is authorized, he/she can create a VPN profile towards their remote devices by specifying flow source and flow destination endpoints (expressed by intentional names), the VPN start time (T_{start}), the duration, the repetition or frequency, security and often used (or target) QoS parameters. This profile is stored in the LDAP Directory and an object instance managing this VPN is created by the VPN CM. Note that the VPN activation time (T_{start}) may be equal to the VPN instance creation time.

At T_{start}, the object instance retrieves the VPN parameters from the LDAP Directory, asks an INR to resolve the intentional names to IP addresses (so as to obtain the node IP addresses and the serving edge router IP addresses) and converts the VPN parameters into VPN configuration policies. Note that the policies can be for VPN security and/or VPN QoS purposes. For the security purpose, the VPN CM will also prepare the secure keys (symmetric keys could be used) and security association parameters. Finally, the CM creates the proper XML configuration document that will be

transmitted using the Netconf protocol to the appropriate routers to achieve their VPN configuration. For the VPN connectivity, a software-based edge router implements a VPN module and integrates a "*configuration manager interface*" [11] that dynamically inserts/removes/ modifies the VPN parameters at run time according to the XML configuration. Therefore, the VPN tunnel is established between edge routers and the AAA server is notified.

At the VPN deactivation (due to a node disconnection or at VPN expiration time), the VPN CM tears down the VPN tunnels in a Netconf operation, by removing the VPN configuration from the correspondents edge routers. At the same time, the AAA server is notified about the VPN deactivation.

III. A SCENARIO EXAMPLE OF DP-VPN APPLICABILITY

Figure 9 presents a scenario example showing the DP-VPN applicability in a medical surveillance case. In this scenario, a doctor "Dr_x" wants to establish a daily dynamic VPN to a hospital "H", in order to supervise his patients via her/his PDA, by means of cameras and medical equipment fixed in their rooms (thus, all those devices are included in Dr_x's PN). First, the Doctor accesses the Web Interface to learn about all equipment fixed in the patients rooms. For example, to discover all available cameras, he can query the services matching the intentional name: [service=camera] [cluster=hospitalH] [vspace=Dr_x]. Then, the doctor creates a profile of a VPN that begins every day at 9 p.m. for one hour, during a week, between a particular camera and his PDA (identified by the intentional names: [service=camera] [room=100] [cluster=hospitalH] [vspace=Dr_x] and [service=PDA] [cluster=P-PAN] [vspace=Dr_x]). Hence, the doctor's PDA will be reachable at any location, independently from allocated IP address (and type, i.e. local or global address). The doctor can also access the "Dynamic Bandwidth Management" Web interface in order to allocate high bandwidth for images coming from remote medial equipment, so that he can guarantee real-time supervision.

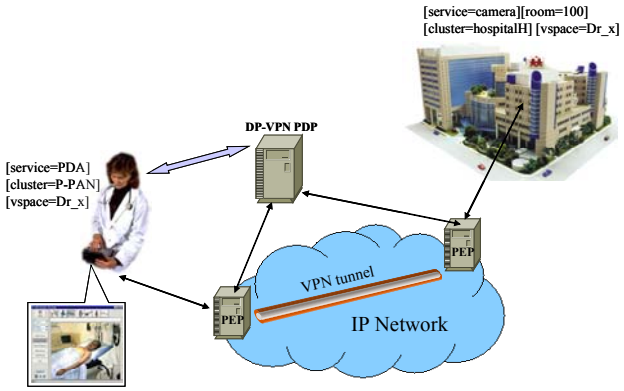

Fig. 9. A scenario example: medical surveillance

IV. PERFORMANCE MEASUREMENT

The proposed architecture was implemented and tested. The developed modules (VPN CM, Netconf client/server, etc.) and the used software (INS, BEEP, etc) are all in Java. The PDP and the INS framework use 2,6 GHz Pentium IV machines with 2 Gb RAM. Two multiprocessors edge routers using Click software router [12] implementing IPSec [13] function are used. The testing environment is a Gbps Ethernet LAN and all machines are running Red Hat Linux 9.0.

Selection of the IPSec for secure VPN establishment is motivated by the fact that it is a well established framework for security and tunneling. IPSec ensures confidentiality, integrity, and authenticity of information transferred across public IP networks. The policy based security mechanisms of IPSec also fit well with the policy based VPN architecture proposed in this paper. The IPSec Tunnel mode can be used to achieve network-based VPNs, with the edge routers acting as IPSec gateways. Finally, as each edge router may support multiple clusters from different PNs, multiple VPN tunnels may be required between the same two physical edge routers. The IPSec Security Parameter Index (SPI) field can be used to distinguish which packets belong to which tunnel.

System performance is assessed for experienced latency for VPN configuration of edge routers. This time is the sum of the time to retrieve the VPN parameters from the LDAP directory, the time of the names-to-address resolution (if names are used instead of addresses), the time of the Netconf communication and the time spent on configuring the edge routers with the VPN parameters.

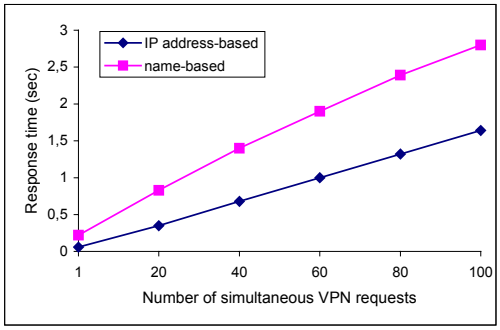

Fig. 10. Performance result

A first experiment determines the response time variation according to the number of simultaneous VPN requests. The goal is to test the system at full load to evaluate scalability performance. The time measured corresponds to the time spent to establish all the VPNs. In this experiment, there is no edge discovery. The end points are passed statically to the VPN CM. Hence, resolution and discovery time is not included in the collected results. A second experiment treats the case where all endpoints are expressed with intentional names to assess the additional latency introduced by the proposed DP-VPN architecture. This additional time is caused by the name-to-address resolution and is reported in Figure 10. For example, a single request takes 60 ms without name resolution whereas it requires 220 ms when using names. The increased response time for many simultaneous requests (e.g. 2,8 seconds for 100 requests) is caused by the existence of many object instances managing the VPNs created by only one VPN CM. Distributing those instances to multiple VPN CMs would reduce this response time. Note that the

interaction with the LDAP database also causes significant delay.

V. Conclusion

This paper presents a Dynamic VPN management architecture for Personal Networks. In the architecture, a PN user is able to make dynamic requests for configuring parameters describing a VPN service. Using naming systems such as INS, PN users can express VPN source and destination endpoints using intentional names, instead of IP addresses, that describes their services. This allows VPN rules to be based on services rather than specific addresses. In addition, INS enables transparent host/service location and offers automated and flexibility in VPN configuration via tunnel endpoint discovery. Besides applicability to the personal network context, this architecture can be perceived as a general *service-oriented VPN management architecture*, potentially useful in pervasive computing environments.

References

[1] « My personal Adaptive Global NET », IST-MAGNET consortium, IST-MAGNET project webpage, http://www.ist-magnet.org/

[2] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, J. Lilley. « The design and implementation of an intentional naming system ». In Proceedings of the 17th ACM Symposium on Operating Systems Principles, pages 186–201, Kiawah Island Resort, South Carolina, December 1999. ACM Press.

[3] Enns, « NETCONF Configuration Protocol », Internet Draft: draft-ietf-netconf-prot-05, February 2005

[4] A. R. McGee, S. R. Vasireddy, K. J. Johnson, U. Chandrashekhar, S. H. Richman, M. El-Sayed, « Dynamic virtual private networks », Bell Labs Technical Journal,Volume 6, Issue 2, 2001. Pages 116-135.

[5] Y. El Mghazli, T. Nadeau, M. Boucadair, K. Chan Nortel, A. Gonguet Alcatel, « Framework for L3VPN Operations and Management », Internet Draft: draft-ietf-l3vpn-mgt-fwk-05, March 2005.

[6] X. Guo, K. Yang, A. Galis, X. Cheng, B. Yang, D. Liu. « A Policy-based Network Management System for IP VPN ». Proc. of the IEEE International Conference on Communication Technology (ICCT'03), Beijing, China, April 2003.

[7] B. Moore, E. Ellesson, J. Strassner, « Policy Core Information Model », RFC 3060, February 2001.

[8] M. Balazinska, H. Balakrishnan, D. Karger. « INS/Twine: A scalable peer-to-peer architecture for intentional resource discovery ». In Proceedings of the First International Conference on Pervasive Computing, pages 195–210, Zurich, Switzerland, August 2002. Springer-Verlag.

[9] E. Lear, K. Crozier, « BEEP Application Protocol Mapping for NETCONF », Internet Draft: draft-ietf-netconf-beep-04, March 2005.

[10] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, J. Arkko, « Diameter Base Protocol », RFC 3588, September 2003.

[11] W. Louati, B. Jouaber and D. Zeghlache , « Configurable software-based edge router architecture », in Proc. of the 4th Workshop on Applications and Services in Wireless Networks, Aug 2004.

[12] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. « The Click modular router ». ACM Transactions on Computer Systems, 18(3):263–297, Aug. 2000.

[13] S. Kent, R. Atkinson, « Security Architecture for the Internet Protocol », RFC 2401, November 1998.