Distributed and Parallel Databases, 12, 217–234, 2002
© 2002 Kluwer Academic Publishers. Manufactured in The Netherlands.

A Reference Architecture for the Certification of E-Services in a Digital Government Infrastructure

F. ARCIERI
Università di Roma 2 "Tor Vergata", Roma, Italy

G. MELIDEO
E. NARDELLI nardelli@di.univaq.it
Dipartimento di Informatica, Università di L'Aquila, L'Aquila, Italy; Istituto di Analisi dei Sistemi ed Informatica, C.N.R., Roma, Italy

M. TALAMO talamo@mat.uniroma2.it
Autorità per l'Informatica nella Pubblica Amministrazione and Università di Roma 2 "Tor Vergata", Roma, Italy

Recommended by: Boualem Benatallah and Fabio Casati

Abstract. Certifying the execution of a service is a critical issue for an e-government infrastructure. In fact being able to document that an e-service was actually carried out, given the legal value that is often attached to data managed and exchanged by public administrations, is of the utmost importance. This is made more complex in cases, like it often happens in the public administration sector, where e-services are based on legacy systems managed by autonomous and independent organizations. In this paper we discuss the introduction, within the standard three tier architecture for e-services, of an architectural subsystem providing certification functions. This architecture features both physical and functional independence from the application level and is made up by new control components providing a highly efficient solution for certification requirements. Our solution has been successfully tested in real-world systems developed in Italy to support digital government functions.

Keywords: digital government support, legacy application interoperability, certification of e-services, technologies and infrastructures for e-services

1. Introduction

The pervasive spreading of the WWW has created a tremendous opportunity for providing services over Internet. The whole area of e-services is becoming an intensely researched field (e.g., see [11] or the special issue [22]), with a wide technological and economical impact. In this context, the digital government sector is quickly gaining importance, given its "potential to profoundly transform citizens' conceptions of civil and political interactions with their governments" [13]. Interoperability [23] and service interconnection with legacy systems [14] are just two examples of research issues with a wide impact in the digital government arena.

An Information Technology (IT) infrastructure supporting e-government services has to certainly allow an efficient monitoring of service execution, that is to be able to check the status of progress of the distributed transaction(s) activated by the monitored service.

Monitoring requires therefore the capability of tracing, analyzing, certifying and documenting what is going on in the distributed system. Monitoring is also important for two additional reasons:

- *contractual*: every party involved in e-service execution has an interest in obtaining a certification of what happened,
- *quality*: service provider is interested in knowing how well the system is performing towards end-users.

However, e-government services are defined and developed in a complex organizational, architectural and technological scenario. In fact, the various public administrations and agencies that are involved are usually autonomously and independently managed. Therefore, even if they had to cooperate to reach a common overall goal, they have developed and run their own computer-based information systems in a completely independent way.

Supporting intragovernmental processes is therefore a hard-to-realize yet critical component of any IT infrastructure for digital government. Within this component, certification is a main function. Consider, for example, a citizen requesting a certificate about a real estate ownership. In a digital government framework, such a request is most probably presented at a service center which is the unique interaction point for all kinds of citizens' requests. This center will then issue such a certificate on the basis of actual information received from the agency responsible for this kind of data. It is therefore important to be able to prove that what was eventually printed by the service center is what was actually sent by the agency. Moreover, if the citizen pays a fee for such a certificate, a sum corresponding to the overall number of certificates successfully printed and delivered to citizens will have to be transferred to the agency. Certification issues are also important in all cases where an e-service requires the interaction of several organizations, e.g. supply chain management [12].

In this paper we first recall the development of real-world e-government systems where we have addressed the certification issue and then present and discuss our solution, which has led to a successful realization of the e-services. These systems have all been developed within an overall initiative, coordinated by AIPA ("Autorità per l'Informatica nella Pubblica Amministrazione"), the italian authority for IT in Public Administration (PA), aiming at developing inter-organization cooperative information systems supporting e-government [17].

Our solution is based on the introduction, within the standard three tier architecture for sites providing e-services, of an architectural subsystem providing certification functions. It features both physical and functional independence from the application level and is made up by new control components providing a highly efficient solution for certification requirements.

It is not the focus of this paper to discuss security issues, which is another very important aspect of digital government infrastructures. For a detailed discussion of such issues as authorization and access control functions in the context of an IT infrastructure for e-government see [15].

The paper is structured as follows. Firstly, in Section 2, we present the context and motivations at the root of our research. Then, Section 3 discusses the reference architecture adopted in our discussion. Next, in Section 4, we present the fundamental technique supporting certification in intragovernmental processes supporting digital government. Section 5 presents application areas where we have tested and refined our solution. Finally, Section 6 concludes the paper. A preliminary short version of this paper, just sketching main issues and giving a high level description of our approach, appeared as [10].

2. Historical background and research motivations

The origin of our research is in legislation [16] enforced in Italy in the years 1993–1994, requiring to set up an information system for cadastral data exchange by means of telematics communication among Ministry of Finance, Municipalities, and Notaries.

At that time computer based systems were already used by the above organizations to manage cadastral data and procedures. Hence each of these organizations was already able to provide (a rough form of) its own e-service to end-users.

The overall challenge was therefore to design the architecture of an IT-based system allowing: (i) to control and certify the exchange of information between independent PA organizations, each with their own hardware and software systems, and its different organizational procedures, and (ii) to ensure that exchanged data was kept coherent in the different PA organizations, even under updates.

No technical solution was available at that time, since either they required to have homogeneous hardware/software systems, or they put one organization in control of the other ones, or both.

We investigated the situation and were able to define, through a series of prototypes developed during 1995–1997, an architectural solution [1] which fully satisfied organizational requirements and supported efficient exchange of data while their coherence is maintained. SICC ("Sistema di Interscambio Catasto Comuni") is a system which allows the exchange of cadastral data among the principal entities in Italy interested in the treatment of cadastral information: the Ministry of Finance, Municipalities, Notaries, and Certified Land Surveyors [6, 20].

Further development efforts refined this solution. A prototype (SCT[1]) was developed to investigate technical and organizational issues to enable a true reuse of geographical data by many users and many organizations [2, 3] and a distributed systems (SIM[2]) was realized to provide e-government services to people living in mountain areas [4] while allowing a full certification of provided service(s).

More details on all development efforts briefly sketched in this section are given later in Section 5.

3. Reference architecture

The reference architecture for our discussion of certification issues supporting intragovernmental processes for digital government is a distributed one (see figure 1). This is compliant with more recent trends and requirements in Public Administrations and e-government actions, where decision capabilities are more and more decentralized and put at the appropriate local level.

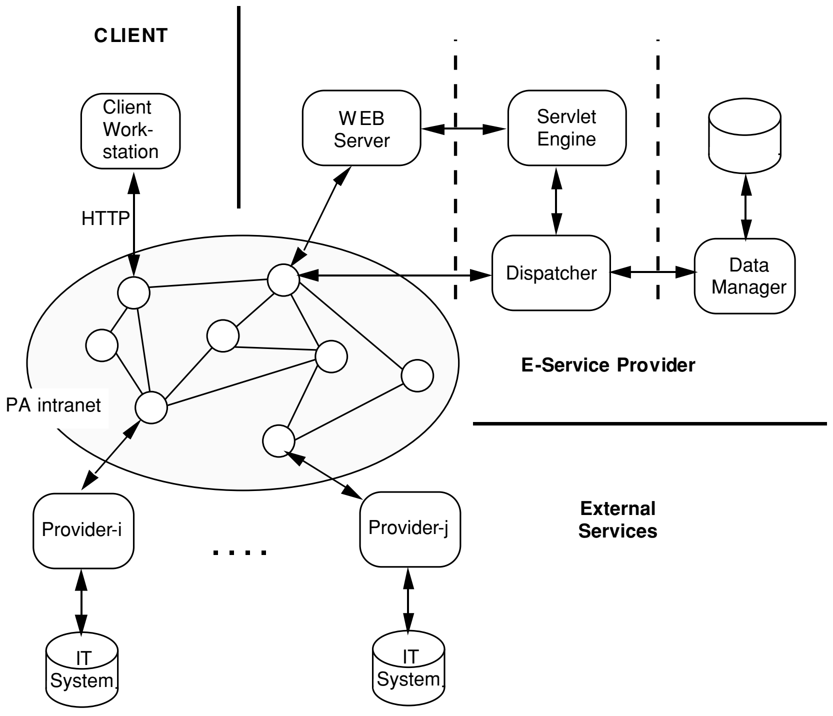

Figure 1. Overall reference architecture.

A number of *external services* are attached to the Public Administration intranet. These services are still managed within Public Administration, but are external with respect to both the client and the e-service provider. They are not directly accessible to every client present over the PA intranet, mainly for reasons of authentication, that is because the client's authenticity is, in general, difficult to establish. This difficulty derives from the fact that the access to the italian PA intranet is allowed also to local municipality networks. All end-users connected to a given local municipality network appear to the PA intranet as the same client, at the connection level. Hence a more complex authentication procedure would be needed for each request if these could arrive directly from an end-user to an external service provider.

Also, parameters for a given external service are, in general, difficult to be properly set-up for a generic client. This is the case, for example, of an external service distributed over many physical sites allocated in different geographical locations, where each site manages data according to a distribution policy internal to the organization. The specified policy can provide for a distribution according to functions or to geographic competence and it is clearly desirable to shield end-users from the need of directly knowing these aspects.

Therefore, even if it seems from figure 1 that a client is physically able to access an external service directly, it is not the case from a functional viewpoint. The client must access a specific web-site whose role is to act as an *e-service provider* for clients. Through this e-service provider, external services are then made available to all clients on the intranet. The site of this e-service provider may also be itself a provider, for its own services, to clients present over the PA intranet.

3.1. Main interactions among subsystems

In a typical example of a service request an end-user connects, through its web client, to the web server of the e-service provider, selects a specific service and activates it providing needed parameters.

Within the provider's site, structured according to the usual three tier architecture, the service request of the client is forwarded as one (or more) application message(s) to an application server, executed within a *servlet engine*. Note that our solution does not require the provider to have such an architecture and can be implemented independently from it. It is clearly much more easier, from the system implementation and maintenance viewpoints, to deal with architectures following the clean approach based on the three tiers.

The application server, once it has received the application message(s) related to the service request, forwards the message(s) to a new architectural component we have introduced, called the *dispatcher*. This is a specific control component, having the task of executing some preliminary checks on service requests and of delivering them to the appropriate external service provider (see also Subsection 3.3). More specifically, the dispatcher:

- identifies the user issuing a given request, its work-station and its access privileges with respect to requested services, e.g., some users may have a read access only, while others may be entitled to read/write data;
- decides and activates security levels adequate for the requested service, e.g., some services might require to transfer some portions of data, of sensitive nature, in an encrypted form;
- checks completeness and correctness of service requests, i.e., the presence of all required parameters and the fact their values are within prescribed ranges
- identifies which basic e-service providers are affected by the end-user's request and forwards them proper sub-requests.

Interaction, for certification purposes, between the servlet engine and the dispatcher and among all e-service providers is made possible by prefixing exchanged application messages with a suitable *application header*, composed by two main parts:

- *message identifier*: with a constant length, allows to identify characteristics of an application message without resorting to external resources; it is present in every application message between a client and a service provider;
- *service parameter*: with a variable length, contains data needed for a correct activation of the required service.

In terms of network protocol layers the application header can be thought of as a protocol lying at the application layer (the same as, e.g., HTTP or FTP) and using the underlying TCP services of the transport layer. Hence the application header is layered on top of TCP like TCP is layered on top of IP.

The structure of application header depends on the kind of intragovernmental process which is being supported and it is defined during its design phase. In any case, it has to contain: identifiers for the end-user and the workstation he/she is working on, a time stamp, transmission parameters, service and function requested. The presence of the application header constitutes a first security level, since it allows to raise an alarm whenever a non-compliant flow is detected.

3.2. Functions and role of the Servlet engine

A service request and its answer makes up a bi-directional *information flow* between a client and a server. Remember that usually over any network working according to Internet protocols an information flow is made up by a number of application messages (e.g., HTTP messages). The servlet engine builds a proper application header for each application message received before forwarding it to the dispatcher. In figure 2 it is recalled the layered structure of the interaction over the Internet between an end-user and a service provider, where each communication happening at a given layer is actually carried out trough services of lower layers. Remember, in fact, that each application message corresponds, on the TCP layer, to one or more TCP segments which, in turn, may be further broken into a number of IP packets. With reference to the same layered structure of figure 2, an example of how a generic request for a service is transformed while going down through the communication layers is shown in figure 3. This figure also shows the insertion of the application header, happening at the application layer. A simplified view of an IP packet traveling on the network illustrating the application header inserted by the

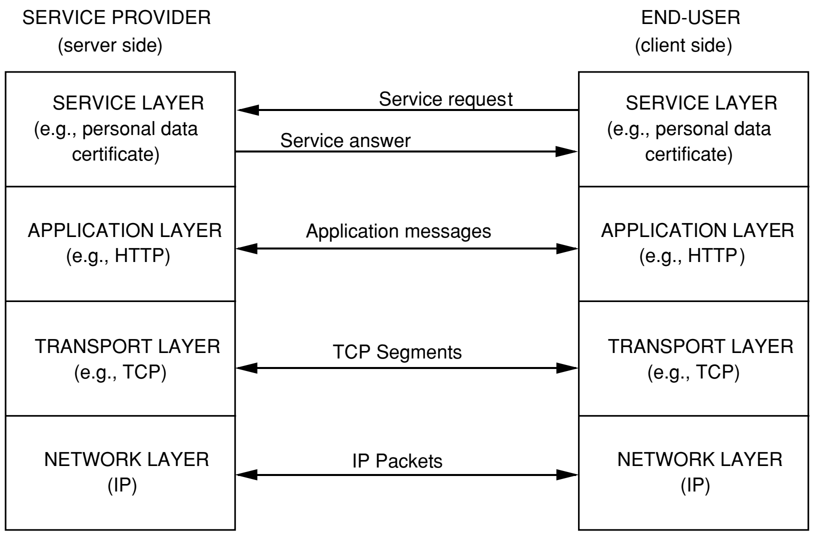

Figure 2. Layers of communication over the network.

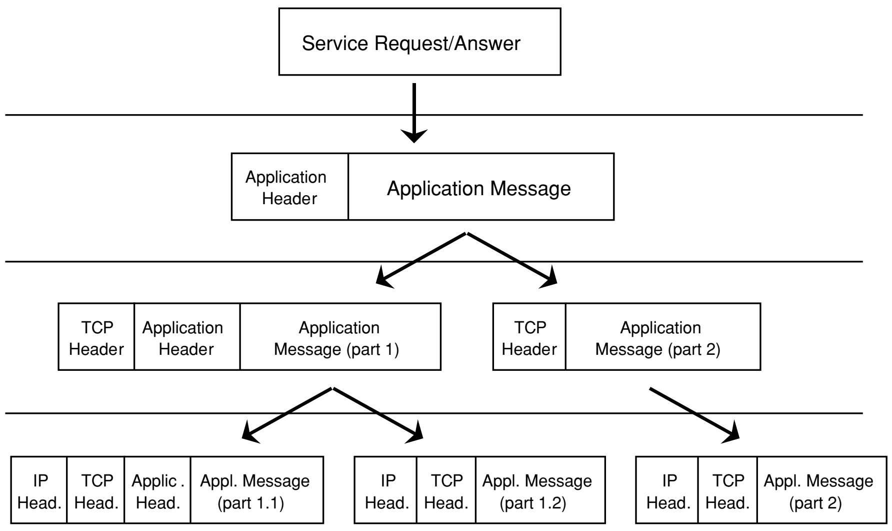

Figure 3. Example of transformation of a service request through the communication layers.

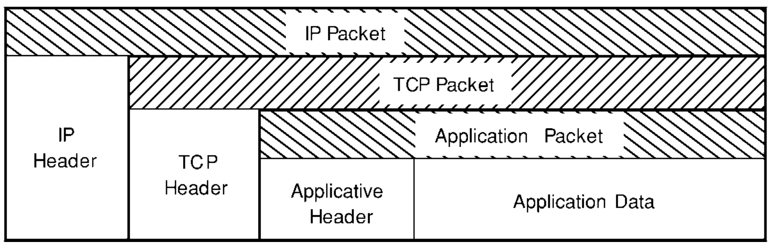

Figure 4. The structure of an IP packet showing the presence of the application header.

servlet engine is shown in figure 4. There, the term “Application Packet” denotes (a part of) an application message exchanged by any application layer protocol as, e.g., HTTP or FTP.

Note that our approach can be integrated with current proposals under development for uniform invocation of remote service, e.g., SOAP [19] or WSDL [21]. In fact the application header is simply a format, agreed within the organizational structure of the PA intranet, supporting certification of exchanged services. It can be encapsulated within SOAP or WSDL invocation protocol, if these become standard. Also, the approach is conceptually valid for supporting certification services even over a public network like the Internet. Clearly, the role played in our current approach by the application header will have then to be implemented through the primitives of protocol(s) that will have been chosen for this purpose.

3.3. Functions and role of the dispatcher

3.3.1. Functional description. The dispatcher subsystem always examines the application header to check authorization levels and to verify correctness and completeness of data in the header: in such a way the end-user can be advised, in case of missing or wrong parameters before forwarding its request to the supplier, with an improvement in overall quality of service.

The Dispatcher analyzes application messages by means of the same algorithmic techniques applied by network probes (see subsection 4.1), and is able to quickly check whether an end-user is requiring a service for which he/she has the proper authorization. This check can be configured to a fine granularity level, allowing to discriminate down to a specific users (among all using a given workstation) and to a specific sub-service. For example, it is possible to enable some users to execute update functions of a given service, while other can only activate display of data. Dispatcher configuration parameters can be changed remotely, even during its normal operation mode, without interrupting it.

Once the dispatcher has positively executed the above described checks the request is forwarded either to the third, or lowest, tier of the architecture (if only internal data management services are needed), or to an external e-service provider (if another e-service is needed to satisfy the end-user's request). In case of negative result of these checks, the end-user is advised using an answer message produced through the customization, with the specific request parameters, of a generic message taken from a set of stored messages defined as part of the dispatcher configuration parameters. An alarm signal can also be sent, in such a case, to a central control server, pre-defined in the dispatcher's configuration. This is a server acting as central controller for service management. If the request cannot be satisfied but it does not fall within any of the pre-defined categories of irregular requests then the dispatcher can establish a communication with the central control server to receive the appropriate answer, to be sent back to the end-user.

Answers from external services are sent back from external service providers to the dispatcher and from this forwarded to end-users without further processing. Note that these return information flows do not have an application header, hence a more complex processing is required for certification purposes, discussed in subsection 4.1.

3.3.2. Discussion. The introduction of the dispatcher has the great benefit of allowing the system to deal with client's authorization and authentication issues outside the e-service application programs. The separation of these aspects of e-services means greater flexibility and independence in changing the application logic. In the above example regarding users enabled to update and users enabled only to read, the service need not know which users among the ones using a given workstation are enabled to use update functions. Since our solution is transparent with respect to messages exchanged both on the end-user's side and on the external service provider's side, no aspect of the client-server interaction has to be changed to implement our certification solution. The consequence is that, even if the service is an old legacy one, no change to its current implementation is ever needed to manage this aspect.

We stress the fact that for an IT infrastructure supporting e-service certification to work efficiently, basic functions be independent and abstracted from the actual e-service invoked or executed. Also note that in our approach no wrapping of services takes place but only a monitoring of the interaction between client and provider for the aim of documenting what actually happened. This monitoring happens at the application level, which is the most effective choice. In fact, monitoring at the lower connection levels (i.e., TCP/IP) would mean the impossibility of giving sense to what is recorded, while doing it higher up, at the service level, would mean to encapsulate services, incurring in the unavoidable slowdown of this type of approach. Of course, our monitoring at the application level is based on actions physically carried out at the lower connection levels, but this is simply due to the fact that application protocols are defined and realized in the Internet through the use of lower layer protocols, like TCP/IP.

4. Certification

The main technical difficulty for certification purposes is that all e-services involved are, from the viewpoint of the certification process, like black boxes and cannot be internally changed.

The only approach is therefore to monitor and to keep track of input and output flows. But input and output flows of IT services taken as black boxes are nothing more than sequence of IP packets.

Then the challenge is to be able to understand, by only checking IP packets flowing through various points over the intranet, what is going on with respect to a specific service. Note that even if we are dealing, in these digital government cases, with the PA intranet, in general all sorts of packets can be found. So the goal is to be able:

- at specific points to collect all IP packets corresponding to specific application flows related to a specific service,
- to reverse engineer streams of IP packets so to reconstruct application level messages passed through monitored points,
- to correlate what has been identified at those specific points so that it can be understood which is the status of the overall distributed transaction,
- to check that threads of application messages (as detected at server sites) corresponding to a given service request are compliant with application rules defined by the service provider for that specific pair of end-user and sub-service,

This approach clearly requires algorithms for IP packet classification with very high efficiency, otherwise it introduces too large of a slowdown in the interaction among the basic components. We have a very good algorithm that solves this problem [18], featuring a small constant processing time to answer positively or negatively the query asking to classify the given IP packet with respect to a set of possible distinct cases (or *classes*). Such a constant value is independent from the size of such a set (i.e., from the number of classes) and directly proportional to the number of “coordinates” used to characterize the universe of the classes. Some examples of coordinates are: ‘Source IP Address’, ‘Destination IP Address’, ‘Transport Protocol’, and ‘IP Version’ and an example of class definition is given by a specific value for ‘Source IP Address’, an interval of values for ‘Destination IP Address’, a set of values for ‘Transport Protocol’ and a specific value for ‘IP Version’.

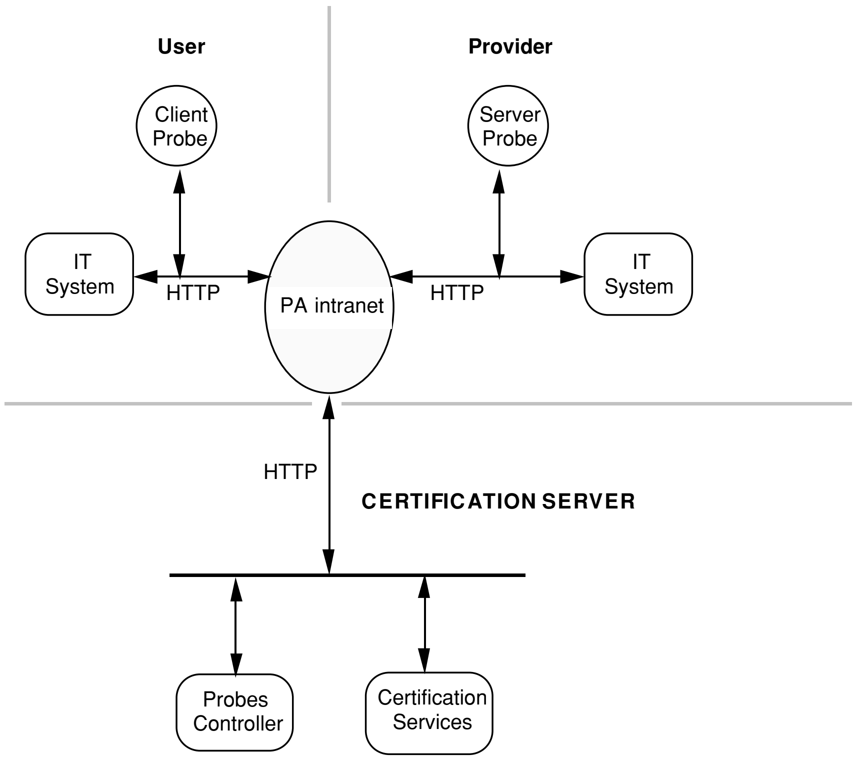

Figure 5. Architectural solution for certification.

Our architectural solution for certification, documentation and accounting of information flows between clients and servers is based on *network probes*[3] (see also figure 5). They operate both on server side and on client side and, once properly set-up, monitor all and only that traffic flowing between them and the communication network for which they have been explicitly configured. In figure 5 network probes are shown at the site of the e-service provider (server probe) and at the site of its end-user (client probe). Note that, of course, configuration of network probes directly derives from agreements formally established between organizations involved in the exchange of services.

4.1. Network probes

4.1.1. Functional description. Network probes are devices featuring high security levels, such as absence of terminal devices to access internal resources (e.g., keyboard, mouse, screen, . . .), uninterruptable power supply, software components for automatic faults check, diagnosis and alert, a "watchdog" subsystem for resetting and re-starting the probe whenever low-level network malfunctions are present for an abnormally long time. Probes can be remotely controlled and configured, even during their normal operation mode—and without interrupting it, by means of both the SNMP protocol and a proprietary language using the UDP protocol.

Probes are able to detect all IP packets passing on the portion of the network they are controlling, to efficiently identify all those related to application services for which they have been configured, to select only those referring to specific kinds of transactions, and to reconstruct application threads relative to services they have to monitor.

Note that probes neither have to collect and process all data passing on the network link they are monitoring, nor have to reconstruct exchanged flows for all servers on the network. They are configured for the specific services listed in the formal agreements signed among the involved organizations and process just IP packets referring to them. Hence this approach is fully scalable without performance degradation. Also note that probes allow a much more efficient support for certification than logging every exchange flow at the dispatcher, since this choice would incur in a high computational overhead for the dispatcher itself.

By suitably processing IP packets, which usually travel over the network without following a fixed route and order, probes in fact reconstruct their exact sequence, hence the individual application messages exchanged by the monitored services. Probes are also able to establish the completeness degree of exchanged information flow, since they can detect missing IP packets.

4.1.2. Information flow reconstruction. The first step in probes activity is the recognition of acknowledged TCP segments from IP packets. Remember that the TCP level guarantees to the application level an ordered reliable byte stream. Hence, when the application sends a message, TCP ensure its delivery to the receiver as one or more TCP segment. Remember also that a TCP segment may be delivered at the IP level as more than one IP packet. Well known problems for this goal are:

- the fragmentation of a TCP segment in IP packets depends on both the static characteristics of the underlying physical network (e.g., the value of MTU = Maximum Transmission Unit) and on the dynamic ones (e.g., the current value of the congestion window);
- routes followed by IP packets referring to the same TCP segments and their arrival time are out of control of the services managing the exchange of TCP segments;
- IP packets may be retransmitted, due to congestion in any point of their route, even when it is just the acknowledge of their arrival that was lost, and can even be retransmitted with partially overlapping content.

For all these problems, though, the really critical issue is the speed of packet classification, guaranteed by our algorithm [18]. For all other issues well known and publicly available solutions exist.

The second step is the reconstruction of the information flow from the acknowledged TCP segments picked up by the probes. This happens in two phases: first, synthetic application-oriented data is extracted from the segment and sent to the certification server and, second, aggregation of synthetic data referring to the same (client, server) pair is established to document information flow. The first phase is the critical one since segments have to be processed very efficiently. It is done by each probe by executing straight-line programs written in an "ad-hoc" defined language.

```
start Ethernet // The first function of the program is the main function.
     ...// Declaration of variables
     main(); // Goes to the subfunction named 'main'
end

start main
     get(0,ip_client); // Typed get:  ip_client is a byte(4), then gets 4 bytes
     get(0,ip_server); get(0,srcp); get(0,dstp);
     mover(0,24); // Steps 24 bytes forward
     get(0,type); get(0,tot_len); get(0,data_len);
     test(type) // The syntax 'in 7[2]:  action;' means:  if the data declared to be
             // tested has 7 as value of the first two bytes then execute 'action'
          in 1[1],in 6[1],in 11[1]:  client-flow(); // same action for three tests
          in 2[1],in 7[1],in 12[1]:  ...
          ...
          default :  Log("Type error"); exit; //'exit' terminates analysis of this TCP segment
     end
     suspend 0; // This process is suspended until more data are available.
     main[]; // This is equivalent to the GOTO statement.
end

start client-flow
     test(srcp)
          in "H00H50":  // Port is the right one
               test(data_len)
                    in 0-2[4]:  ; // No sufficient data available, do nothing and return.
                    default:  get(1,method,2); check-appl-method();
               end
          default:  Log("The port is not HTTP (80)");
     end
end

start check-appl-method
     test(method)
          in "PO":  ... // Recognized 'POST' method (HTTP);
          default:  return 0; // No more actions on this connection
     end
     //Extract synthetic data from the application message
     ...
     identify_username();
     ...
     // Write synthetic data in a buffer to be sent to the certification server
     buftosend << key_session @ type_trans @ ver_pkt @ type_pkt @ ip_client @ len @ lcbyte;
     // Sends the buffer.  The parameters are:
          // - the buffer to be sent
          // - two chars classification flag
          // - a compression flag (0/1)
          // - a DES flag (0=no encritpion, 1-255 encript using n-th key)
     write-in-queue(buftosend,"WC",0,0); // Asynchronous send
end
```

Figure 6. A simplified excerpt of a program for extracting synthetic application-oriented data from TCP segments.

In figure 6 it is shown a simplified, but real, example of a fragment of such a program. The program assumes that lower level functions of the probe have picked-up an acknowledged TCP segment, i.e., a confirmed part of an application message, and have made data available to the program through two channels. The first one (channel 0) contains information about the connection (fully identified by four parameters: source IP address, destination IP address, source port, destination port), while the second one (channel 1) is the data buffer containing all data in the acknowledged TCP segment. The `test` operator is the critical step in performance terms: its execution, still using the same technique of [18], runs in time proportional to the number of bytes to be tested (which is always a small constant), using a space proportional to the product (#-bytes)(#-distinct-values).

The reconstruction of the 'forward' information flow (i.e., from the end-user to the e-service provider) is easier for network probes, since at least one of the IP packets corresponding to the application message(s) contains the application header. On the contrary, the 'backward' information flow is much harder to reverse engineer for two reasons: (i) none of the corresponding IP packets contains an application header, (ii) the flow is usually decomposed in many application-level messages, each referring to a small part of the whole flow. An example of the latter problem is encountered when the backward information flow consists in a form, filled through a series of interactions among the end-user and the service, where at each interaction step only some of the fields present in the form receive a value, hence each application message sent back to the service only contains part of the overall answer.

The first problem may be overcome through a deeper analysis of the data part of packets, to extract needed information. The second problem requires network probes to be configured by providing them: (i) the structure of information flows to be detected and legal values for each of the structure's component, and (ii) the actions to be executed whenever specific value and/or sub-structures are detected. The first element is therefore *declarative* knowledge while the second one is *procedural* knowledge specifying processing actions to be executed as a consequence of the detection of specific patterns. Given this information a network probe is able, by collecting all IP packets referring to the same information flow, to incrementally build and reconstruct the whole structure exchanged.

4.2. Probes controller

4.2.1. Functional description. Summary information items filtered out by network probes as result of their real-time analysis of IP packets according to their configuration parameters are sent to a central *probes controller*.

Once summary data reach the probes controller, it processes them and establishes a correlation among those referring to the same service flow. This correlation defies spoofing attacks. Spurious IP packets might, for example, be introduced in the network as a fake server's answer to a client's request. But even if their TCP/IP coordinates are an exact replication of the correct ones and packets are prefixed with the correct application header, they will be recorded only by the probe on the client side. Since the probe on the server side has not seen them, the probes controller is not able to match information recorded on the client side and the attack is repelled.

Note that with this approach network traffic overhead is very low since only summary elements, at the application level of the communication, and only those relative to the service for which the probe has been configured, are sent back to the probes controller.

Probes also compute, on the payload part of packets, a suitably defined hash function, whose outcome is sent, together with other information, to the probes controller. In such a way the controller is able to check also the correct exchange of payload data between provider and end-users, hence to provide a certification of the integrity of exchanged data.

Correlations among summary information sent by probes are also stored in a relational DBMS for reporting and statistics purposes. Using them various kind of reports can be produced, documenting from the single execution of a service to all services (possibly grouped in classes) received by a given end-user or delivered by a given provider. Of course the probes controller has to be able to interact with and to receive information from many probes on the network, but this is as well defined and accepted in the formal agreements signed by the involved organizations.

With this solution a complete documentation for each service is built and can serve as the official basis for certification of services. Whenever a legal value is associated to data there is in fact the need of certifying the actual supply of the requested service and its correct receipt by the client. This certification action is carried out by a *certification server* on the basis of correlation among exchanged data items established by the probes controller.

4.2.2. Discussion. An important aspect of our solution based on network probes is that, since it is both physically and functionally independent from applications inter-operating over the network, it allows certification and documentation services to be allocated, if desired, to a third-party, independent from both the servers and the clients involved in the exchange of e-services. Note that this does not mean to send externally all IP packets of PA since the third party is anyhow within the PA's intranet. But with this choice the technical leverage point to enact, control, and enforce regulation policies for e-government services is obtained. 'If', 'when', and 'what' are of course issues of competence of higher-level policy makers, but it is important to stress that this technical solution is transparent with respect to the inner structure and operation of the involved providers. Hence it does not incur the risk of being unusable in practice due to technical mismatches and difficulties.

From the economical viewpoint, one should note that what has to be explicitly added to the technological infrastructure that is anyhow needed for the provision of e-services to end-users are the network probes and the certification server. Their cost is almost negligible with respect to the overall costs incurred for the operation and management of the involved e-services.

Note also that documentation about information flows is useful not only for certification purposes: it is critical for performance tuning and monitoring actions and for establishing and controlling the actual levels for quality of services [7].

5. Implementation and applications

In this section we first give some more details on implementation and performance related aspects of the proposed architecture and then present real-world systems implementing it.

5.1. Implementation

5.1.1. Dispatcher. The dispatcher is able to process, on the average, 500 application messages per second even when it is managing more than 500,000 distinct (*end-user*, *sub-service*) authorization pairs. Sometimes, it may happen that for some authorization pair it is needed to access the external service provider to receive further information required to complete the evaluation of the request. Under this more complex framework the dispatcher is still able to process 100 application messages per second against a set of authorization pairs of the same size as above.

5.1.2. Network probes. Activity of network probes is very efficient and neither disturbs nor slows down operations of the communication layers. They are physically placed on network links directly leading to or coming from client and server nodes and each of them is able to process in real-time over 500 TCP connections active on the link it is watching. The maximum size of data lost by a probe is 0.05% of the whole volume of watched data passing over the assigned link, for a 10 Mbit/s Ethernet link fully working at its maximum speed. With the same assumptions, in the case of a 100 Mbit/s link the maximum size of data lost is 0.1%.

From the reverse engineering of IP packets probes extract suitable data items able to certify and to document information passed through the link under their surveillance. This summary information, about 200 bytes for each TCP segment passed over the link, is temporarily stored locally for later delivery to a central server supervising activity of all probes (probes controller, see below).

5.1.3. Probes controller. Communication protocol between probes and the central controller is designed so that its operation does not affect efficiency of probes watching functions.

For confidentiality reasons the transmission of summary information to the probes controller is encrypted according to the PGP scheme, using either private or public keys, according to configuration parameters. Different keys may be used for summary information referring to different services. Moreover, for increased security and to prevent "man-in-the-middle" kind of attacks, all communications between probes and the controller are encrypted.

Transmission is periodically activated according to the schedule defined in probe configuration. Assuming a daily average traffic of 100,000 TCP segments to be monitored over the watched link, a probe is able to store locally summary items for as long as 100 days, to ensure recovery from a long period of communication faults or malfunctions. Local storage may be as well encrypted, if specified by configuration parameters.

5.2. Applications

As discussed in Section 2 the first system where the solution here described was implemented is SICC ("Sistema di Interscambio Catasto Comuni"), which allows the exchange of cadastral data among the principal entities in Italy interested in the treatment of cadastral information: the Ministry of Finance, Municipalities, Notaries, and Certified Land Surveyors [1, 6, 20].

The system is accessible nation-wide through a Web-based interface since September 1998. The effectiveness of its use is demonstrated by the number of administrative transactions successfully completed through the system in year 2001: they are 800,000 per month, corresponding to 60% of the overall transactions related to cadastral services accomplished every year in Italy. Note that certificates related to ownership, location, geometry and value of real estates, are mandatory in estate's selling transactions and their issue is subject to a fee, paid for by the buyer. In the month of September 2001 the total value of Ministry of Finance's income deriving from certificates issued through this system corresponds, on a yearly basis, to the 20% of the overall sum cashed by the Ministry for cadastral services, which is estimated for the year 2001 to amount to 78 million euros (roughly 70 million US dollars[4]).

Afterwards we defined SCT ("Sistema di Comunicazione Dati Territoriali"), that is a prototype developed to investigate technical and organizational issues to allow lowering of costs related to geographical information systems development and maintenance, to enable a true reuse of geographical data by many users and many organizations and thus foster the development of a market for territorial data [2, 3].

The design and prototyping of SCT system was carried out by a consortium led by Telecom Italia and comprising three more large companies (namely, ESRI, Finsiel and IBM). Such a project is one of the steps taken by AIPA in the joint cooperation framework (namely, "Intesa Stato-Regioni-Enti Locali per la Realizzazione dei Sistemi Informatici Geografici di interesse generale") with the Region Association and the Municipality Association[5] for coordinating efforts aimed at supporting the growth of the sector of geographical information systems.

The third and most important example is SIM ("Sistema Informativo della Montagna"), that is a distributed systems providing e-government services to people living in mountain areas [4].

SIM acts as a mediator between citizens living in those areas and IT-based services in the fields of cadaster, labour and pension, public registry of personal data. Its design started in 1998 and the overall financial effort has been, until now, of about 52 million euros (roughly 47 million US dollars). It is currently being used in about one thousand operating centers, all over Italy, serving more than 10 million inhabitants, more than 4000 of the about 8000 Municipalities and covering more than 54% of the italian territory. Its extension to the whole country is currently being planned.

Finally we would like to notice that the development of two newest italian e-government services, where the fourth author is the overall IT infrastructure supervisor, coordinated by the Ministry of Internal Affairs, namely the *Electronic Identity Card* and the *Integration of Municipal Public Registries of Personal Data*, is being driven according to the solutions described in this and related papers [1–6, 8, 9].

6. Conclusions

In this paper we have considered certification issues arising in intragovernmental process supporting digital government services. Certification of exchanged e-services is a critical activity in any application areas, but is of the utmost importance for e-government since very often exchanged data have a legal value and play a legal role.

The proposed solution is based on the introduction of two novel kinds of components, namely *dispatchers* and *network probes*, which allow the efficient and effective handling of certification issues. It also provides the technical platform on which to base control policies, if these policies are deemed necessary. Our solution directly derives from successful experiences in the development of a number of real-world systems. A first abstraction in terms of a formal computational model is described in [8], while further developments of the underlying formal model for data coherence maintenance among independent organizations are presented in [5] and [9].

Our approach requires a careful modification to be used in the case of public networks such as Internet. In fact, very often in this case services are provided on top of a secure communication layer (e.g., SSL). The conceptual architecture here described can still be used, but radically different technical solution have to be implemented for the probes.

Acknowledgments

We would like to thank Elettra Cappadozzi and Aurora Girolamo for many useful and interesting discussions during the development of the work here described. Numerous and detailed comments and observations from all referees have greatly helped in improving the presentation. Last, but not least, we would like to thank the editors of this special issue for their support.

Notes

1. Sistema di Comunicazione Dati Territoriali.
2. Sistema Informativo della Montagna.
3. Patented, 1997.
4. At the exchange rate of about 0.9 US dollars per 1 euro.
5. Conferenza Stato-Regioni and Conferenza Stato-Regioni-Città.

References

1. F. Arcieri, E. Cappadozzi, P. Naggar, E. Nardelli, and M. Talamo, "Access key warehouse: A new approach to the development of cooperative information systems, in 4th Int. Conf. on Cooperative Information Systems (CoopIS '99), Edinburgh, Scotland, UK, Sept. 1999, pp. 46–56. Extended version published in the Int. J. of Cooperative Information Systems, vol. 11, nos. 1/2, pp. 175–200, 2002.
2. F. Arcieri, E. Cappadozzi, E. Nardelli, and M. Talamo, "Geographical information systems interoperability through distributed data exchange, in 1st International Workshop on Databases, Documents, and Information Fusion (DBFusion'01), Magdeburg, Germany, May 2001, Preprint n.8/2001, Fakultät für Informatik, Universität Magdeburg.
3. F. Arcieri, E. Cappadozzi, E. Nardelli, and M. Talamo, "Distributed territorial data management and exchange for public organizations," in 3rd International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS'01), San Jose, CA, USA, June 2001, IEEE Computer Society Press, pp. 42–51.
4. F. Arcieri, E. Cappadozzi, E. Nardelli, and M. Talamo, "SIM: A working example of an e-government service infrastructure for mountain communities," in Workshop on Electronic Government (DEXA-eGov'01), Conf. on Databases and Expert System Applications (DEXA'01), Sept. 2001, Munich, Germany, IEEE Computer Society Press.
5. F. Arcieri, E. Cappadozzi, G. Melideo, E. Nardelli, P. Naggar, and M. Talamo, "A formal model for data coherence maintenance," in Int. Workshop on Foundations of Models for Information Integration (FMII'01),

10th Workshop in the series Foundation of Models and Languages for Data and Objects (FMLDO), Viterbo, Italy, Sept. 2001. Lecture Notes in Computer Science, vol., Springer-Verlag: Berlin.

6. F. Arcieri, C. Cammino, E. Nardelli, M. Talamo, and A. Venza, "The Italian cadastral information system: A real-life spatio-temporal DBMS," in Workshop on Spatio-Temporal Database Management (STDBM'99), Edinburgh, Scotland, UK, Sept. 1999. Lecture Notes in Computer Science, vol. 1678, Springer-Verlag: Berlin, pp. 79–99.
7. F. Arcieri, F. Fioravanti, R. Giaccio, E. Nardelli, and M. Talamo, "Monitoring QoS performance of cooperative services in a digital government framework," in 3rd Int. Workshop on Software and Performance (WOSP'02), July 2002, Roma, Italia.
8. F. Arcieri, R. Giaccio, E. Nardelli, and M. Talamo, "A framework for inter-organizational public administration network services," in Int. Conf. on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet (SSGRR'01), L'Aquila, Italy, Aug. 2001, IEEE Computer Society Press.
9. F. Arcieri, G. Melideo, E. Nardelli, and M. Talamo, "On the dynamics of an infrastructural approach supporting coherence maintenance for inter-organizational collaboration," in Int. Symp. on Business Strategy Based Software Engineering (SoftwareTrends'01), Sept. 2001, Gersau, Switzerland, NetAcademy Press.
10. F. Arcieri, G. Melideo, E. Nardelli, and M. Talamo, "Experiences and issues in the realization of e-government services," in 12th Int. Workshop on Research Issues on Data Engineering: Engineering E-Commerce/E-Business Systems (RIDE'02), Feb. 2002, San Jose, USA, IEEE Computer Society Press.
11. F. Casati, U. Dayal, and M.-C. Shan, "Report on the VLDB workshop on technologies for E-services (TES)," SIGMOD Record, vol. 29, no. 4, pp. 11–15, 2000.
12. F. Casati, U. Dayal, and M.-C. Shan, "E-business applications for supply chain management: Challenges and solutions," in 17th Int. Conf. on Data Engineering (ICDE'01), Heidelberg, Germany, 2001, pp. 71–78.
13. A.K. Elmagarmid and W.J. McIver, "The ongoing march toward digital government," Guest Editors' Introduction to the special section on Digital Government, IEEE Computer, vol. 34, no. 2, pp. 32–38, 2001.
14. A. Firat, S. Madnick, and M. Siegel, "The cameleon web wrapper engine," in Workshop on Technologies for E-Services (TES'00), Cairo, Egypt, 2000.
15. J. Joshi, A. Ghafoor, W.G. Aref, and E.H. Spafford, "Digital government security infrastructure design challenges," Computer, vol. 34, no. 2, pp. 66–72, 2001.
16. Law Decree n.557 of 30/dec/93 and Law n.133 of 26/feb/94.
17. M. Mecella and C. Batini, "Enabling Italian E-government through a cooperative architecture," IEEE Computer, vol. 34, no. 2, pp. 40–45, 2001.
18. E. Nardelli, M. Talamo, and P. Vocca, "Efficient searching for multidimensional data made simple," in 7th Annual European Symposium on Algorithms (ESA'99), Prague, Czech Republic, July 1999, Lecture Notes in Computer Science, vol. 1643, Springer-Verlag: Berlin, pp. 339–353.
19. Simple Object Access Protocol (SOAP) 1.1, W3C, http://www.w3.org/TR/SOAP.
20. M. Talamo, F. Arcieri, G. Conia, and E. Nardelli, "SICC: An exchange system for cadastral information," in 6th Int. Symp. on Large Spatial Databases (SSD'99), Hong Kong, China, July 1999, Lecture Notes in Computer Science, vol. 1651, Springer-Verlag: Berlin, pp. 360–364.
21. Web Service Description Language (WSDL) 1.1, W3C, http://www.w3.org/TR/wsdl.
22. G. Weikum (ed.), Bulletin of the Technical Committee on Data Engineering, Special Issue on Infrastructures for Advanced E-Services, vol. 24, no. 1, 2001.
23. J. Yang and M.P. Papazaglou, "Interoperation support for electronic business," Communications of the ACM, vol. 43, no. 6, pp. 39–47, 2000.