Collaboration and the Knowledge Economy: Issues, Applications, Case Studies
Paul Cunningham and Miriam Cunningham (Eds)
IOS Press, 2008 Amsterdam
ISBN 978–1–58603–924-0

Common Capabilities for Trust & Security in Service Oriented Infrastructures

David BROSSARD[1], Theo DIMITRAKOS[1], Maurizio COLOMBO[2]
[1]*British Telecommunications plc, Adastral Park, Martlesham Heath, IP5 3RE, UK*
[2]*IIT-CNR, Pisa, Italy*
Contact author. Tel:+44 1473 60614, Email: david.brossard@bt.com

Abstract: Trust and security are of paramount importance to the successful implementation of Service Oriented Infrastructures based on Web Services and Grid technology. Security failures in a networked economy are often the result of exploiting the fuzzy boundaries of independently robust, partial security solutions. In contrast to the current state of the art, that is mainly based of results coming from the efforts of the e-Science community, such as VOMS [4], GSI [14], CAS [14] or GridShib [5], the solutions proposed in this paper are closely related to the needs and requirements of the business world. Following closely the current and future needs of European businesses has been a critical factor in the requirements elicited and the design choices made by the Business Experiments in GRID (BEinGRID) project [1]. The results presented in this paper stem out of Trust & Security activities of the EU FP6 BEinGRID project, and related work contributed by the EU FP7 project GridTrust [4].

1 Introduction

BEinGRID is the largest European research initiative looking into business applications of Service Oriented Infrastructures using SOA Web Services and Grid computing. The BEinGRID initiative has 96 partners including European enterprises of all sizes and is conducting 25 real-world Business Experiments (BE) across most market sectors. Each of these Business Experiments is assessing the relevance of some business solution that takes advantage of Grid computing in a specific market sector.

The BEs are supported by teams of technical consultants and business analysts, with whom they interact closely in order to elicit common requirements; prioritise them in terms of popularity, reusability, innovation potential and business impact; identify common capabilities; and finally validate the reference implementations of these common capabilities on the most relevant Grid and Web Services platforms. BEinGRID is also building a public knowledge repository [2] where case studies, market analyses, technical analyses and software components are going to be provided.

The objective of the paper is to identify the common capabilities of establishing trust and security in service oriented architecture and Grid computing system, so that more applications can be developed based on this platform, more modern services can be provided by service providers and cutting-edge technology can be easily implemented.

The following are some major research challenges identified through the analysis of 25 Business Experiments in several market sectors, conducted by the BEinGRID project:

- *Managing identities and federations in dynamic business collaborations*: How to manage the life-cycle of circles of Trust? How to enhance the structure of a circle of trust? How to coordinate a network of identity brokers in order to support the life-cycle of a Virtual Organisation? How do you contextualise identity issuance, how do you manage virtual identities and claims that are specific vary between virtual communities?

How do you manage revocation of claims in a large-scale distributed system? How do you delegate the authority to issue credentials on one’s behalf within given contexts?

- *Security autonomics in large scale, network-centric distributed systems*: How to detect or inform about contextual changes, and how to adapt in response to contextual changes in a large-scale distributed system based on local knowledge? How to interpret and consistently enforce VO-wide policies? How to adapt the way you manage, interpret and security policies in a dynamic environment where resources are shared across multiple, potentially unrelated, administrative domains.
- *Distributed access management in large*-scale decentralised systems: how to manage, reason with and enforce access policies in large-scale, network-centric distributed systems? How to share policy information across administrative domains? How to confirm that obligations are met? How to keep managing the confidentiality of your data and access to your applications once hosted in another’s environment?
- *End-to-end security and Governance*: How to achieve end-to-end security of interactions with Grid resources? How to aggregate security services in a Grid? How to securely govern aggregated security services that are distributed over the network?

2 Overview of selected security capabilities

In this section we summarise some of the security capabilities designed and developed by the BEinGRID and other relevant projects such as GridTrust [3] and TrustCoM [4]. These capabilities cover several important functionalities such as offering connectivity to external identity and security attribute providers; supporting the full management life-cycle of federating of trust realms; supporting distributed access control including the delegation of administrative authority in multi-administrative environments; offering real-time monitoring of policy enforcement; supporting real-time policy adaptation in response to events. These security capabilities can be offered as managed security services or as reusable components integrated into a larger security-enabling infrastructure. The following sections offer an overview of these capabilities. For more information please refer to www.gridipedia.eu

2.1 *B2B collaboration management*

This is a bundle of services that support the full life-cycle of defining, establishing, amending and dissolving collaborations that bring together a circle of trust (federation) of business partners in order to execute some B2B choreography. This capability consists of three distinct services that bundle the functionality as required by the different actors involved in the management of a B2B collaboration infrastructure:

- *Host:* provides common capability inventories, policy registries and notaries. Facilitates sharing the state of the B2B collaboration context.
- *Initiator*: manages the creation of B2B collaboration contexts. It implements a process composing all services required for managing the lifecycle of a B2B collaboration context.
- *Participant*: manages the participation in a B2B collaboration context. It manages the state of the member in each B2B collaboration context and coordinates interactions with the rest of the B2B collaboration management infrastructure.

The lifecycle of the B2B collaboration is divided in four phases:

- *Identification*, which includes the definition of the B2B collaboration and the associated B2B policies and agreements.
- *Formation*, which includes the discovery, invitation and selection of B2B collaboration partners and the initiation of a circle of trust among them.

- *Operation*, which includes initiating the creation of the service instances that are exposed by the B2B collaboration partners, monitoring partner-to-partner interactions and adapting the partner selection or the assignment of resources to the B2B collaboration.
- *Dissolution*, which includes cancelling the B2B collaboration context, the policies and agreements associated with it, and the service instances are destroyed and/or become inaccessible.

Please see [8] in these proceedings for more information on policy-based management of the Virtual Organisation life-cycle. Previous attempts to architect services that capture certain aspects of this functionality include work by BT and SAP Research in the TrustCoM consortium in [4], [9].

2.2 *B2B Federation Management services (FMS) and Security Token Services (STS)*

The STS acts as an identity broker for each enterprise, and manages the correlation of identities and security attributes within a security domain with commonly understood credentials across domains. It allows:

1) managing a local participant's perspective of a circle of trust, and
2) adapting local authentication mechanism, token scheme, identity token transformation scheme based on contextual information, secure remote management.

The STS pattern originates research by BT and Microsoft (EMIC) in the TrustCoM project [4], [9],[10]. A more elaborate presentation of this capability is also described in [12] and in paper [8] in these proceedings.

2.3 Distributed Access Control (AuthZ Policy Decision Point - PDP)

This capability provides authorisation services that allow the necessary decision making for distributed enforcement of access policies by multiple administrators, ensuring regulatory compliance, accountability and security audits. This is achieved by extending current access control models with (1) validity conditions for each authorisation policy, (2) policy issuance whereby administrators have to digitally sign the policies they produce, and (3) administrative delegation policies that allow a trusted administrator to define who can issue policies about what actions on which subset of the administered resources.

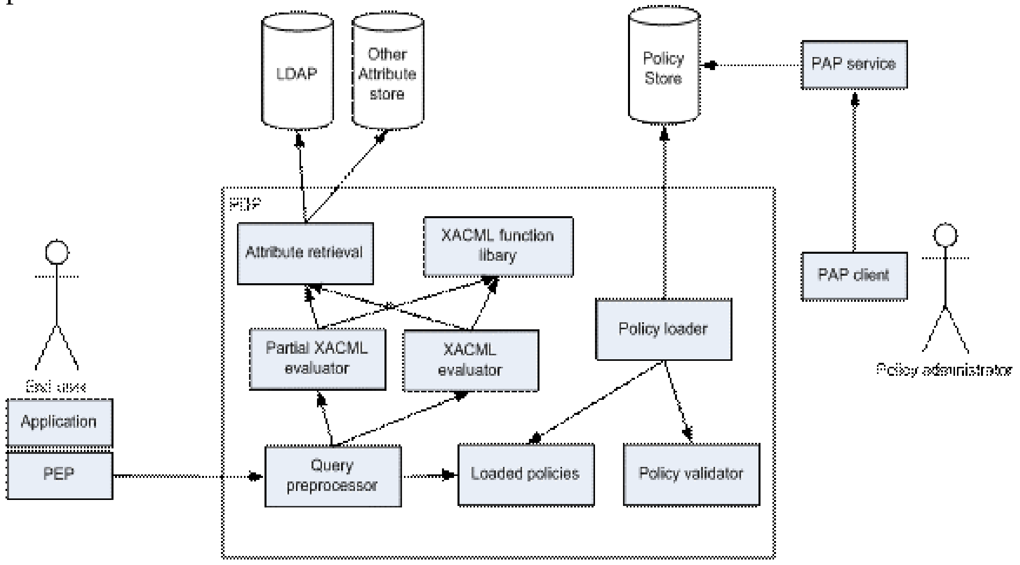

Figure 1: High-level overview of the Policy Decision Point architecture

Key functions include:

- *Policy-based access control*: applicable policies are stored on the system and are analyzed by the PDP. The PDP makes its decision and returns the decision. The policies can all be expressed in standardised access control languages such as XACML 2.0 / 3.0
- *Constrained Administrative Delegation:* the delegation mechanism is used to support decentralized administration of access policies. It allows an authority (delegator) to delegate all or part of its authority to another user (delegate) without any need to involve the central IT-administration. The specific authorization can then be delegated further by the user, in full or constrained to a subset of the original authorization. In this delegation model, the delegation rights are separated from the access rights. These are instead referred to as *administrative control policies* (see the XACML 3.0 draft). These policies can be targeted in the same way as access control policies. Access control and administrative policies work together.
- *Obligation*: when centralizing the security architecture with the XACML model, an obligation is a directive from the PDP to the Policy Enforcement Point (PEP) on what must be carried out before or after an access is granted. If the PEP is unable to comply with the directive, the granted access will not be realized. The augmentation of obligations eliminates the gap between requirements and policy enforcement previously described.
- *Segregation of policy stores*: by means of PDP instantiation and creation of separate stores, it is possible to have instances of the PDP service that each act as a single standalone PDP unaffected by the policies pushed to the other PDP instances. This is particularly important in a virtualized / contextualized environment where different virtual organizations may require their own PDPs.

A more detailed description of this capability is provided in [12]. The design and implementation of capability is based on research by BT, Axiomatics and SICS in the TrustCoM and BEinGRID projects [7], [6], [4], [9].

2.4 *Message-based Security Policy Enforcement Engine (PEP)*

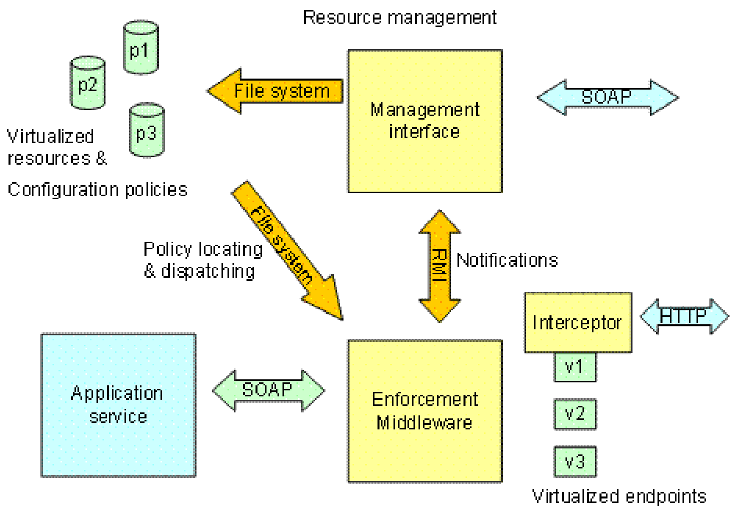

Figure 2: High-level overview of the Policy Enforcement Point architecture

This is a *secure message processing system* that is a fusion of (1) an application service firewall / gateway that protects interactions to XML applications and Web Services, (2) a proxy that intercepts, inspects, authorises and transforms content on outgoing requests to external services, (3) a message bus that enforces content- and context- aware message processing policies and (4) a light-weight core of a service bus that integrates all other SOI security capabilities. The PEP provides message-level security enforcement based on B2B agreement as opposed to the application logic.

The key functions of this capability include:

- *Policy-based message processing:* Processing actions are defined by means of security policy assertions that are executed by message processing engine as well as more complex actions such as remote calls to STS, PDP for token issuance, validation or access control request, etc… The PEP can also perform transformation actions on the content of intercepted or generated messages (including encryption / decryption of elements) as well as content-based routing and message structure or content validation.
- *XML threat protection*: In a more comprehensive variant, the PEP can also handle basic XML threat protection by validation messages against their relevant schemas, checking the format of the request, SOAP operations and their input parameters, and the simple XML threats such as node depth and so on.
- *Context-based security enforcement*: the PEP intercepts a SOAP message, analyzes the SOAP headers (typically the address headers) to locate a suitable enforcement policy which in turn determines which security operations the PEP will enforce.
- *Service contextualization*: the PEP allows exposing an enterprise's internal services externally. This is part of the service virtualization. As a result when incoming messages come in, the PEP not only processes it and applies security, it also routes it to the relevant web service in the back-end.
- *Management*: The PEP management service is based on WSDM (WS-Distributed Management). When a protected application service is virtualized and exposed, a new instance is of a PEP management resource created in order to manage the policies to the protected service. This resource instance can only manage the exposure of the associated protected application service and the corresponding security enforcement policies. This brings a clear separation of concerns regarding the administration of Web service exposures. Furthermore different implementations of PEP can be clustered without affecting the protected service – to – management resource association.

A more detailed presentation of this capability is provided in [12]. See also [15] for an elaboration of an earlier version of this pattern produced within BT's UK research labs.

2.5 Usage Control Service (UCS)

In an environment in which computational services execute unknown applications from different users, the resources must be protected from abuse by the applications through usage monitoring at all levels. At the computational level, such a monitor controls the actions performed by the applications executed on the associated computational service. At service level, it monitors the invocations to services executed on the grid node. At VO level it monitors the invocations to a group of associated services in the VO.

The Usage Control Service (UCS) is a common capability that is typically deployed on the service provider site. It takes as input the VO-level policies, the local resource policies and a user profile and produces the equivalent policy state machines. Based on this input, the UCS service generates policy state machines that are used by the resource-level monitors for policy evaluation.

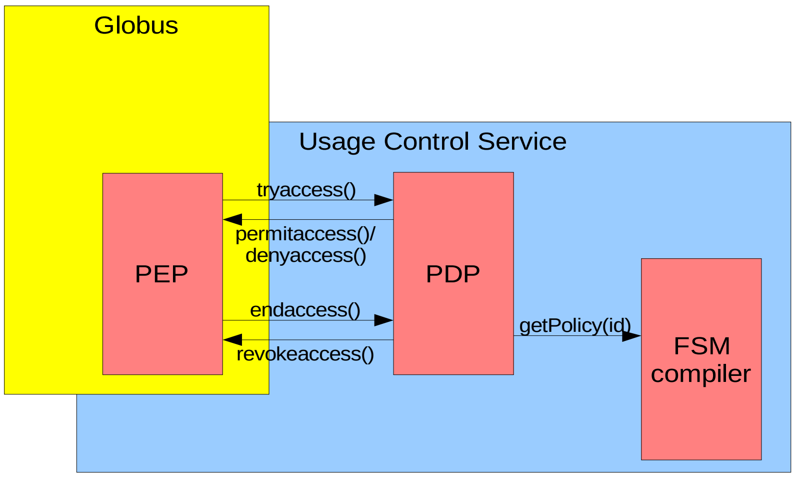

Figure 3: High-level overview of the Usage Control Services deployment architecture

The following components are needed in a UCS implementation:

1. *Policy Enforcement Point (PEP)*: is the component integrated within the middleware platform upon which the UCS is deployed (e.g. the Globus Toolkit). Its main task is the interception of the security relevant actions performed by the execution of the associated security policy; for instance it asks the PDP for a decision and subsequently implements those decisions.
2. *Policy Decision Point (PDP)*: is invoked by the PEP, it evaluates the policy and decides whether the action is allowed. While actions are in progress it interrupts the execution when the right granted to the user has been revoked.
3. *FMS compiler*: takes as input a Global VO policy and a Local policy and produces the appropriate *PDP* policy to be enforced on this Grid Node.

An implementation of this capability is currently being produced by IIT-CNR within the scope of the GridTrust project [3].

2.6 Security Autonomics

Adaptation Service (ECA): implementing novel technology that allows reconfiguring the security services in response to security or QoS events in order to optimise performance and to assure compliance with agreements and enterprise policies. This is achieved by correlating and processing events from managed infrastructure services (e.g. the capabilities above), then applying Event-Condition-Action policies that in turn triggers reconfiguration or other life-cycle management actions on managed infrastructure services under its control. An overview of the pattern for this capability is presented in [12]. It is based on research by BT and Imperial College in TrustCoM [9] and subsequent work [16].

Security Observer: implementing an assessment and reporting functionality that observes security changes or violations and generates events that can be in turn be acted upon by other capabilities such as the ones implementing the *Security Autonomics* layer summarised above. The importance of this function stems from the fact that due to the uncertainty and dynamics of Grid infrastructures local security changes may have a global impact (or an impact in certain other localities) but are not necessary observable (or understood) in the localities affected. A publisher-subscriber-based mechanism can be used to disseminate security change events. This can be coupled with a *Complex Event Processing* functionality which will enable the correlation of relevant events in order to achieve a better understanding of their aggregation. Overall the aim of such a system is, on the one hand, to understand the emerging global impact of various local security changes, and on the other hand, to inform those system entities that are interested in such security changes of some type. An initial version of the security observer has been developed by a team at CETIC in the context of BEinGRID and it is being validated within the scope of BE03 [17], a business experiment on Visualisation and Virtual Reality.

2.7 Virtualisation & life-cycle management

This is a governance layer managing the life-cycle of a secure exposure (aka “virtualisation”) of enterprise resources. Service virtualisation is achieved by exposing a service at a given “virtual” endpoint, associating a reference to this endpoint with a profile of a Service Oriented Infrastructure configuration and structure of policy templates for each infrastructure services in the profile. Life-cycle management is achieved by a governance layer that orchestrates the service exposure process, the life-cycle of instances of the infrastructure profile, the life-cycle of the configuration of each infrastructure service in the profile and the life-cycle of each policy for a configuration state of an infrastructure service in the infrastructure profile. Although this component was not part of the initial functionality envisaged for BEinGRID, it arose as a necessary integration and management layer through the contribution of one of the BEinGRID experiments (BE9 – Virtual Hosting Environment for online gaming [13]). Please see [8] in these proceedings for more information on this capability.

3 A Case Study

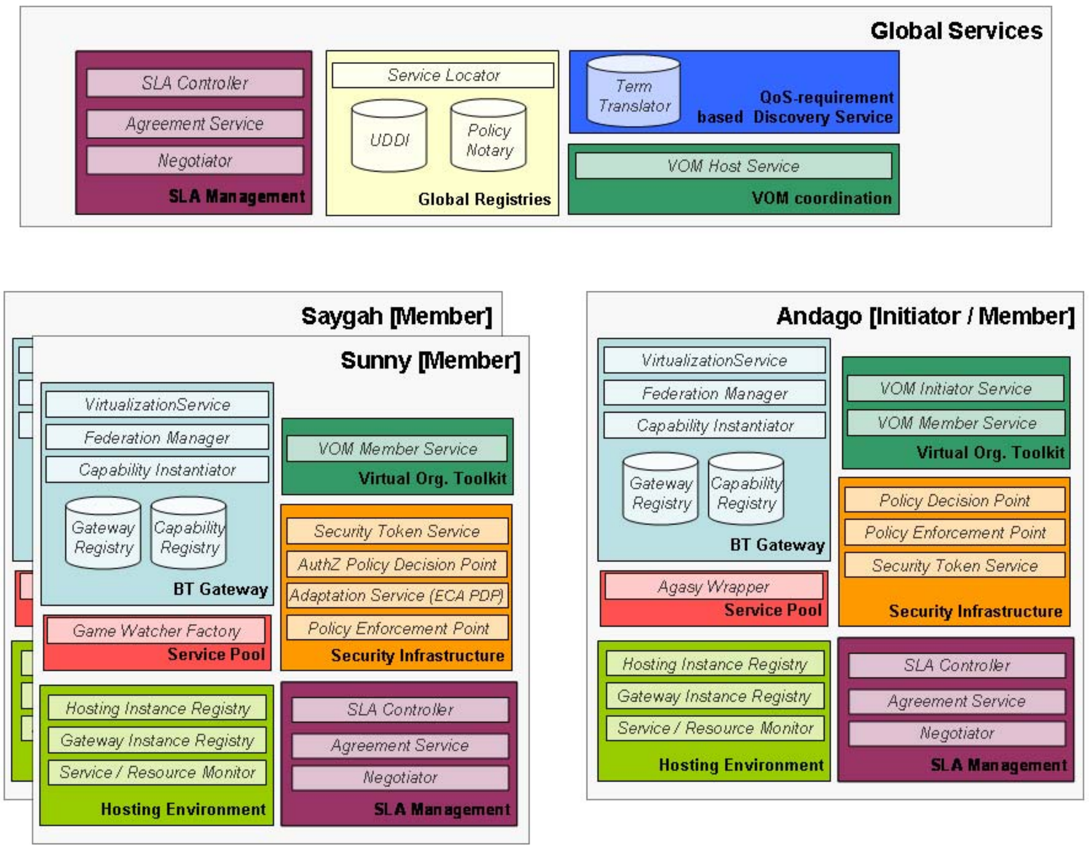

Figure 4: Overview of the VHE infrastructure deployed in [13]

Some of the common capabilities described above have been demonstrated in a Business Experiment of the BEinGRID project [1] within the scope of a Virtual Hosting Environment (VHE) for online gaming [13]. In this scenario a distributed online game application runs over a Virtual Hosting Environment (VHE). The latter is a Service Oriented Infrastructure composed of the following subsystems
(i) Global services for B2B collaboration management service (VOM),

For each business partner:

(ii) A B2B gateway integrating and governing the following groups of infrastructure services
  (1) VOM participant services
  (2) A Security Infrastructure
  (3) A SLA Management infrastructure

(iii)the hosting environment, and

(iv) a pool of application services, i.e. the online gaming platform, game management services (Game Watcher) and managed game titles.

In a typical scenario, a number of host providers offer hosting resources to the Application Providers for deploying and running their applications, which are then “virtualized” with the use of middleware services for managing non-functional aspects of the application, and are transparently exposed to the end user via a single VHE.

The aim of this business experiment is to improve flexibility, dynamism and performance of the game application exposure and execution. Current gaming platforms, in fact, are very static in nature, with pre-selected dedicated game servers. As such, the platforms experience extreme peaks and lows in demand, due to the period of the day or week, and ongoing gaming activity. This causes very low utilization of dedicated gaming servers, and therefore high cost of initial investment and maintenance.

The approach taken through the use of VHE is to make available infrastructure services for security, community management and virtualisation that can be used by various service providers, allowing them to link with each other. This is achieved via a generic “B2B gateway” component (see Figure 4).

In this scenario, the game application provider deploys its gaming application onto two different execution environments (gaming servers), owned by different host providers. The game platform provider, who wants to offer the game to an end user, discovers gaming servers and creates business relationships with them, and also with a separate service provider who offers a system for community management (of gaming clans, tournaments, advanced statistics). Through use of the VHE, these various services are offered transparently to an end user, including the game platform provider’s ability to perform the load balancing and server selection based on the defined SLAs

The VHE developed in this business experiment consists of a network of B2B service gateways integrated with common capabilities for B2B trust federation, identity management, access control, SLA management, accounting and monitoring, as well as application service and resource virtualisation. The B2B gateway functionality is complemented by a federated messaging bus and community management services that facilitate the establishment of B2B collaborations

4 Conclusions and related work

Security is a key challenge because Grid adopters must trust the global infrastructure and this cannot be achieved without proper security by design. This is especially difficult because of intrinsic characteristics of the Grid such as openness, heterogeneity, geographical distribution and dynamicity. Consequently, Grid security has been and is currently being investigated by a number of research groups and projects such as AssessGrid (risk-based approach), GridTrust (design of next-generation security framework), BEinGRID (integrated project involving 25 real-word industrial deployment of Grid Technology), and XtreemOS (Linux-based Operating System to Support Virtual Organizations). Each project is looking at establishing robust security patterns and leveraging existing middleware architecture and software to provide trust, security and privacy. BEinGRID is different from the other related initiatives in that it coordinates a large number of business experiments across various critical market sectors. These drive the elicitation and prioritisation of common requirements, the identification of common capabilities and the validation of reference implementations of these common capabilities on the most relevant Grid and Web Services platforms. The trust and security capabilities presented in this paper have resulted from such a process within the BEinGRID project [11],[12], augmented with selected results from associated targeted research projects such as GridTrust [3]. It is the intention of the authors to make the implementations of the common capabilities described in this paper available through the Gridipedia repository [2].

The trust and security capabilities presented in this paper take the form of a managed (and potentially) network-hosted service that enables an enterprise to achieve the following benefits:

- To virtualize its applications, employee accounts, computing / information resources; this can be achieved by plugging the governance layer into the VO Cluster's application virtualization component
- To govern such virtualised entities, including defining trust relationships enacted by the STS, security and access policies, identity schemes, etc, and to enforce them.
- To adapt the observable behaviour of virtualized entities and the use of infrastructure services in response to contextual changes. Examples include updating the security, privilege provisioning, or access control policy in response to changes of business process activity, changes of membership in a B2B collaboration, etc.
- To securely expose such assets to an open network through the PEP, with the option to apply different security policies in different collaboration contexts while, ensuring process and information separation between services transacting in different collaborations.
- To maintain the management of its participation in B2B collaborations. This includes managing the life-cycle of its participation in B2B collaborations and contributing to the joint management of the B2B collaboration with the other participants.

Through the study, we learnt the following lessons:

Lesson 1 – Security needs to be designed in the Business Experiments: security is a non-functional requirement and as such is often forgotten. So we have to emphasise that security solution and architecture needs to be well-designed and implemented in order to make full use of grid technology.

Lesson 2 – interoperability is essential since several specifications & different implementations are used in the distributed systems communications. For example, some BE has a mixture of Java, .Net, a wide range of web services specifications, and certificates usage. In particular when it comes to web services, because Java and .NET have separate implementations for – say WS-Trust, or WSRF, it was important to thoroughly test and address these issues before final integration stage.

Lesson 3 – Compatibility: the trust and security common capability (PEP-PDP-STS) need to be developed as a modular with a pluggable, extensible architecture to accommodate new security components operating with different standards.

The Best Practices can be summarised as follows from our research:

Best Practice 1 – Use software as a service – the SaaS pattern: Business experiments should pick up the solid security solution from a panel of ready-made components and use them directly in their architecture. In order to do so, security component needs to be developed and easily adopted as a service by an SOA-oriented architecture or a Grid-based architecture.

Best Practice 2 – Make full use of existing security components.

Best Practice 3 – Decouple business logic from security: this avoids poor security patterns, technology lock-ins, incompatibilities, and non-extensible systems. Security should be seen

as an add-on layer that can be configured and executed independently of the business logic. With new web service frameworks for instance, such as WSE or WCF, the security configuration has been offloaded to an external layer.
Best Practice 4 – Plan for an extensible architecture: A business experiment's world might grow and new components might be brought in. It is necessary to introduce extensibility by leveraging grid technology to develop pluggable exchangeable components.
Best Practice 5 – Interoperability & impact: When adopting security solutions, business experiments should spend more to consider interoperability issues. Also, as a general rule, the introduction of the additional security layer or components should not downgrade the system performance too much.

Acknowledgments

The paper is the result of work from several FP6 and FP7 European projects including TrustCoM, BEinGRID, and GridTrust. In addition, several components from FP5 European project GRASP were used in the demonstration mentioned in section 3.

References

[1] BEinGRID project site: www.beingrid.eu
[2] Gridiptedia repository site: www.gridipedia.eu
[3] GridTrust project site: www.gridtrust.eu
[4] TrustCoM project site: www.eu-trustcom.com
[5] Seitz L., et al (2007): A Classification of Delegation Schemes for Attribute Authority. In Dimitrakos et al. Formal Aspects in Security and Trust 2006, LNCS, Springer.
[6] J. Alqatawna, E. Rissanen, B. Sadighi: Overriding of Access Control in XACML. POLICY 2007: 87-95
[7] Seitz L., et al (2007): A Classification of Delegation Schemes for Attribute Authority. In Dimitrakos et al. Formal Aspects in Security and Trust 2006, LNCS, Springer.
[8] Angelo Gaeta, F. Orciuoli, N. Capuano, D. Brossard, T. Dimitrakos. A Service Oriented Architecture to support the federation lifecycle management in a secure B2B environment. In proceedings of e2008.
[9] Dimitrakos, Theo. TrustCoM Scientific and Technological Roadmap. Restricted TrustCoM deliverable available upon request. Contact: theo.dimitrakos@bt.com
[10] Christian Geuer-Pollmann. "How to Make a Federation Manageable". In Proc. of "Communications and Multimedia Security" 9th IFIP TC-6 TC-11 International Conference, CMS 2005.
[11] Theo Dimitrakos, et al. "Common Technical Requirements for Grids and Service Oriented Infrastructures". BEinGRID report. Restricted – ontact theo.dimitrakos@bt.com
[12] Dimitrakos, Theo, et al. "Common Capabilities and Design Patterns for Grids and Service Oriented Infrastructures". BEinGRID report. Restricted – please contact theo.dimitrakos@bt.com
[13] BEinGRID Business Experiment 9: http://www.beingrid.eu/index.php?id=be9
[14] GSI: Globus Security Infrastructure. See http://www.globus.org/toolkit/docs/4.0/security/
[15] Andreas Maierhofer, Theodosis Dimitrakos, Leonid Titkov, David Brossard: Extendable and Adaptive Message-Level Security Enforcement Framework. ICNS 2006: 72
[16] K. P. Twidle, E. Lupu: Ponder2 - Policy-Based Self Managed Cells. AIMS 2007: 230
[17] BEinGRID Business Experiment 3: http://www.beingrid.eu/index.php?id=be3