

# Secure Enterprise SOA

## Known and New Security Challenges

Markus Schumacher, Dominik Witte

*SOA looks like a promising paradigm that could simplify the IT of future business. If you don't consider security, however, you will probably fail by implementing your SOA roadmap. In this article we outline 10 (known and new) security challenges) and like to encourage to think more about Secure Enterprise SOA.*

### Introduction

SOA looks like a promising approach for maximizing the IT support of business processes, which are subject of constant change. Experts say that SOA help companies to adopt more quickly and cost-effectively to new business needs [1]. Whenever such a new paradigm is introduced, you should carefully consider the impact on today's business.

In its annually Hype Cycle, Gartner says that SOA needs another 5-10 years until SOA is commonly used [2,3]. Thus, the migration to SOA should be considered as a journey that requires careful planning and a sound roadmap. A plan requires thinking about known and new challenges that arise by implementing SOA concepts.

As security in an enterprise application context is our main area, we will provide insights about security questions that must be answered when going down the SOA road. We have discussed this with software architects, users, consultants and vendors at various occasions (e.g. CeBIT 2006 & 2007, NetWeaver Security Infoday 2007, SAP Developer Network, etc.) since early 2006.

Please see this article as a status quo report and as a call for further discussions in order to identify all SOA security challenges and to provide meaningful answers.

### Enterprise SOA, Securely

Enterprise SOA means standardized, reusable services that can be used as building blocks of business applications such as Customer Relationship Management, Supply Chain Management, Human Capital Management, etc. The term "enterprise" indicates a stronger focus on the business process (in contrast to underlying technol-

ogy). As a consequence you are going to find entities like "customer", "sales order", "invoice", etc. and corresponding services that process them. From a business point of view, SOA is first of all a management concept and second of all requires a system architecture [1]. From a technical point of view, SOA is the platform that is used to represent distributed business process in a modular and flexible way.

### 1.1 State-of-the-Art Application Security

Much security advice is publicly available today. A well-known source is the Open Web Application Security Project (OWASP) which "is dedicated to finding and fighting the causes of insecure software" [4]. While you can find excellent material there (and elsewhere), most security guidelines apply for Web applications, but are not necessarily applicable in the context of enterprise applications (that's why OWASP is going to start an Enterprise Application Security Project [5] – volunteers are highly welcome!). On the other hand, enterprise security guidelines today don't necessarily consider SOA issues too. In summary we identify a gap in terms of security for Enterprise SOA approaches.

### 1.2 Attacking SOA Solutions

In the following, we discuss a few possible attack targets in order to motivate why we see known and new security challenges. The figure on the following page shows a typical SOA representation: there are different layers of abstraction; each of them can potentially be used by an attacker for a security breach. We show a few, selected examples for each layer (by nature, our list is not complete).



Markus Schumacher

CEO of Virtual Forge GmbH.  
Topics: RO(S)I, Security Research, Business Application Security, Application Security Processes

E-Mail: markus.schumacher@virtualforge.de



Senior Consultant at SecurIntegration GmbH.

Topics: Security concepts and analyses, Service Oriented Architectures.

E-Mail: dominik.witte@securintegration.de

The easiest way to attack an application is to manipulate the client (see for example Phishing [6] or Cross Site Scripting [7] attacks). Therefore, the *presentation layer* can be used as a starting point for subsequent attacks that can bypass lower-level security mechanisms. Example: if an attacker can successfully steal the logon credentials of a user by sending a malicious script, the attacker can impersonate the user and execute further transactions with the user's privileges – authorization checks and other mechanisms are no help in such a case.

At the *control layer*, the attacker would try to get an advantage by manipulating processes and workflows. For example, he tries to change the order of process steps, to find alternative routes to a business object, or to identify loops (which lead to a Denial-of-Service).

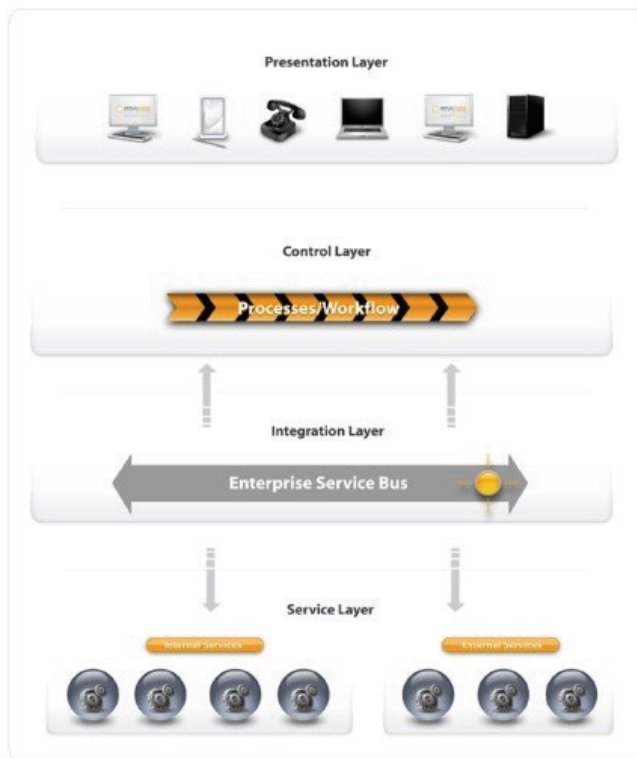
The enterprise service bus (ESB) at the *integration layer* is responsible for exchanging data. It's triggered by requests from the control layer and routes them to the corresponding services. The target of an attacker would be, for example, to introduce his own requests, to change requests, to delete requests or to send as many as he can in order to bring the ESB down.

At the service layer we find both internal and external services. At this layer, we are close at the design and implementation of the business functionality. That is, bugs and flaws of a service can introduce loopholes for attacks. In terms of external services, an attacker might try to hijack a service or to introduce his own service(s) that will do surprising things (e.g. parsing credit card data and forwarding it to another service that is controlled by the attacker).

Many more attack vectors can be imagined, in particular, in shaded areas where the vertical layer and horizontal company boundaries are fuzzy.

### 1.3 Why things go wrong

In the past decades it was (more or less) possible to distinguish between bad (the Internet) and good (internal systems and applications). That way, it was possible to enforce security at the border between known applications or – in most cases - network domains. Today, such boundaries vanish – it's no longer clear who's inside or outside – a trend that is significantly increased by the SOA paradigm. Gray McGraw summarized this a more general



trend nicely as the trinity of trouble [8,9]: exposure, complexity, and extensibility. We briefly drill each of them down in a SOA context:

1) Exposure: as discussed before, SOA-based applications can consist of internal and external services, they can be distributed all over the world. Thus, you have to expect that parts of them are exposed to the Internet and therefore to attacks (not to speak about internal attacks against internal services).

2) A word of the security expert Bruce Schneier says "Complexity is the worst enemy of security. Secure systems should be cut to the bone and made as simple as possible. There is no substitute for simplicity. Unfortunately, simplicity goes against everything our digital future stands for." [10]. Although SOA seems to be easy on the higher levels at a first sight, much complexity is hidden in the lower levels. In summary, software becomes more and more complex. The will be more software that runs somewhere and interacts with other many other objects. All in all this makes security even more challenging.

3) Extensibility: SOA-based applications are by definition flexible and extensible in a very dynamic way. Getting an idea about the security impact of the new functionality and its side-effects on the existing functionality will be very hard.

## Ten ESOA Security Challenges

In our joint Enterprise SOA Security Task Force (which is open for everyone who likes to contribute answers for ESOA related security questions), we have identified the following areas so far:

1. Authorizations
2. End-to-End Security
3. Interoperability
4. Secure Configuration
5. Quality of Service
6. Secure Development
7. Assurance
8. Regulatory Compliance
9. Auditing
10. Migration of Legacy Applications

Each of the ESOA security challenges will be briefly discussed in the following.

### 2.1 Authorizations

Authorizations are a function that is most severely affected in an Enterprise SOA world. Traditionally, authorizations are used to protect business functionality (that is processes and workflows) of an application server. Authorization checks are implemented during design/implementation time and are hard-coded in the respective application.

cations and modules. The application server is in charge of the enforcement.

In an Enterprise SOA scenario, however, there is no central application server anymore that can enforce authorization restrictions. Instead, business processes spread across a variety of servers, each one providing selected elements of the process – the enterprise services. This implies the definition and implementation of multiple authorization concepts, possibly for completely different systems. Therefore, it is necessary to form a consistent overall authorization concept for the business process to be protected: if a single system authorization is not set properly, the complete business process might be unusable and might introduce a vulnerability that allows an attacker to misuse the services. The same is true for the user management: Users that shall be able to use a business process need to have a user account in all systems and be granted adequate permissions. This can create tremendous administrative maintenance efforts.

In order to ensure maintainability, user and authorization management should be centralized and integrated in an overall *Identity Management* framework wherever possible.

## 2.2 End-to-End Security

Common and widespread security measures such as SSL encryption can be considered as insufficient in a SOA world: they only focus on the low-level transport layer, the messages and requests of a process are not individually protected.

This is especially true for cross-company processes, as you cannot enforce security measures in a business partner's system landscape. Thus, security measures must be independent from the transfer channel. They have to aim at the protection of the business data itself. Only that way, the data itself remains protected, even when passed from one service to another beyond company boundaries.

For web service messages, the OASIS standard *WS-Security* looks like a suitable technology [11]. It provides confidentiality and integrity protection for messages by means of encryption and digital signatures.

## 2.3 Interoperability

Interoperability is a crucial point in Enterprise SOA. Services on multiple systems,

probably being written in different programming languages and running on various application servers shall be able to communicate without high efforts. Thus, a clear definition of industry-wide accepted standards is indispensable for a SOA landscape to evolve. However, interoperability has been proven to be more difficult than anticipated initially. In addition, most SOA experiences have been made in rather homogeneous landscapes in rather closed business scenarios so far. For example, there are different implementations of the Enterprise Service Bus (such as provided by IBM, Microsoft, Oracle, SAP, or TIBCO). Without a standard, high integration efforts will remain.

## 2.4 Secure Configuration

Enterprise services are often accessible via public networks and are therefore exposed to increased threats (compared to purely company-internal applications). By deploying an enterprise service, parts of these applications can now be reached from potentially hostile environments. This needs to be encountered with a secure configuration of the service itself and its complete application framework, i.e. application servers, process engines, etc. This includes clear and complete documentation of all relevant configuration settings and security functions by the service provider or application vendor as well as an adequate training of the people in charge. A *secure by default* delivery, where only minimum functionality is activated can help to minimize loopholes from the beginning. As the SOA landscape evolves, *regular security audits* will be a key success factor for a continuous high security level.

## 2.5 Quality of Service

Whether you use or provide an enterprise service, in both cases you will need to consider the quality of the service. If you consume a service, you want the service to be available and to provide its business functionality in an acceptable time. Note that the least reliable service can dictate the availability of the overall business process.

For externally procured services, these factors can only be indirectly influenced via contractual service level agreements and a multi-vendor approach where multiple service providers with comparable services

are integrated and can be easily exchanged with each other.

Thus, the very same parameters - availability and performance – are distinctive factors in a competition for each service provider and need to be monitored. The service quality provided can become a factor that influences the service's pricing and demand.

Security as a quality should be part of such a view, too. However, we still lack of widely accepted and used *software security metrics*.

## 2.6 Secure Development

The development of SOA-enabled services is different in several aspects from traditional software development. As a service should be easily reusable in various possibly hostile contexts, a developer must make no assumptions about its environment – he simply does not know how the service is going to be used.

Wherever this is not possible, all assumptions need to be made explicit by documenting them. Only then a service user can consider to use the service (or not) in a responsible way.

With respect to security, a service developer should always use *existing security services* provided by the runtime environment instead of designing new ones (which will often result in “snake oil” solutions). As most developers are no security experts, only the use of existing and proven security functions ensures a proper level of security (see for example Security Patterns [12]).

Another important issue is that a developer must not introduce security bugs and flaws. If he does, his service becomes the weakest link that can easily break a chain of security measures. Developers should be in a position to know typical problems. Regarding “low hanging fruits” (e.g. avoiding Buffer Overflows, using appropriate Authorization checks, etc.) development guidelines and corresponding test cases are a key success factor. However, it is important to call software security experts for harder security issues (the attacker will not only look at the easy ones. You should expect attackers with a high skill profile in a business context).

## 2.7 Assurance

Having secured an individual service is mandatory but not sufficient for security a

complete business process. Overhasty trust in the security of a process can be dangerous. This is especially true for externally procured services and processes. As a SOA infrastructure can comprise many constantly changing service providers, few long-lasting and trustworthy business relationships exist. Instead, trust in a service and its provider needs to be provided by means of certifications and objective audits or reviews. A *security certification program* that describes and checks meaningful security requirements will be useful. Existing standards like the Common Criteria [13] are too complex and too inflexible for dynamic SOA scenarios.

## 2.8 Regulatory Compliance

Compliance means to adhere to legal requirements. SOX, as a result of severe financial manipulations (Enron, Worldcom, and others), is a prominent example for an emerging business challenge. If executives don't follow such regulations, they personally get into deep trouble.

IT and software systems need to support compliance demands and proactively monitor their abundance. For SOA landscapes, this is additionally complicated due to the distribution of business processes. Existing isolated compliance solutions need to be integrated in order to gain a holistic view.

Note, that insecure services have a compliance impact, too. If your services are insecure, someone can potentially manipulate transactions. Therefore, you simply don't know what happens. Therefore, you are no longer compliant.

## 2.9 Auditing

Strongly related with regulatory compliance and often a legal requirement is the traceability of transactions. Again, the distribution of a business process across multiple systems and services severely complicate the auditing process. Actors and their actions need to be traced throughout the whole process. Also, the auditing information that is generated on the individual participating systems must be consolidated

in the end. Therefore, we see auditing as an important SOA challenge, too.

## 2.10 Migration of Legacy Applications

SOA doesn't start from scratch. There's always a roadmap that starts with extending existing „legacy“ applications by creating some sort of a SOA shell around. The idea is to explore new possibilities and advantages – step by step.

On the other hand, the “SOA readiness” with respect to the previously mentioned challenges must be carefully considered. Existing applications might contain weaknesses that were irrelevant in their existing context. All of a sudden they can become critical as they might be exposed to the public.

Also, expert knowledge about the existing applications is often not available anymore. This makes it very difficult to give a sound statement about their security and to add possibly required SOA-relevant functionality.

## Conclusion

SOA will be a long journey that probably has already started. In an enterprise context, we can see obvious advantages in terms of a significantly IT support of the continuously changing business processes. It will be easier to map process steps to IT components, to react timely considering new business demands and to improve the integration of systems and application landscapes.

Too good to be true? Yes. Old security questions have to be asked again in this new context. The applications will be more open while staying secure (compare this with the security model of a medieval town and a metropolis like New York, London or Berlin). As a consequence, known security measures (like firewalls) will no longer be sufficient.

In addition, new security questions arise, in particular when processes cross the company domain and public services are involved. There will be many new demands and no-one can go back to a long lasting

experience and “well hung” best practises. Our concluding advices:

- 1) When you think about Enterprise SOA, you should think about security.
- 2) Don't forget the security lessons learned so far. Watch out for new pitfalls!
- 3) Security is not a product. Don't forget to ask your people in order to master the SOA security challenge successfully.

Our list of 10 security challenges does not claim to be comprehensive and complete. Therefore, we like to encourage you to get in touch in order to discuss our hypotheses and to add more thoughts. The more people think about Secure Enterprise SOA in advance, the higher the odds for a successful (=secure) business in future.

## Literature

- [1] [http://en.wikipedia.org/wiki/Service-oriented\\_architecture](http://en.wikipedia.org/wiki/Service-oriented_architecture)
- [2] Gartner, Emerging Technologies Hype Cycle, 2006 <http://www.gartner.com/>
- [3] TecChannel, Gartner beurteilt Reifegrad neuer IT-Techniken [http://www.tecchannel.de/test\\_technik/news/445402/index3.html](http://www.tecchannel.de/test_technik/news/445402/index3.html), 2006
- [4] Open Web Application Security Project (OWASP), <http://www.owasp.org>
- [5] OWASP Enterprise Application Security Project, [http://www.owasp.org/index.php/OWASP\\_Enterprise\\_Application\\_Security\\_Project\\_Roadmap](http://www.owasp.org/index.php/OWASP_Enterprise_Application_Security_Project_Roadmap)
- [6] Sven Türpe, Anke Baumann, Phishing-Schutz im Online-Banking, Fraunhofer-Institut SIT
- [7] Andreas Wiegstein et al. The Cross Site Scripting Threat, Virtual Forge Whitepaper, 2007
- [8] Gary McGraw, Software Security, Addison-Wesley, 2006
- [9] Greg Hoglund, Gray McGraw, Exploiting Online Games: Cheating Massively Distributed Systems, Addison Wesley, 2007
- [10] Bruce Schneier, Software Complexity and Security, Crypto-Gram, March 2000
- [11] OASIS Open. Web Services Security: SOAP Message Security 1.1 (WS-Security 2004).
- [12] M. Schumacher et al., Security Patterns, J. Wiley & Sons, 2005
- [13] Common Criteria, Version 2.3 [www.commoncriteriaportal.org/](http://www.commoncriteriaportal.org/)