The SEMPER Framework for Secure Electronic Commerce

Matthias Schunter, Michael Waidner, Dale Whinnett

Inhalt

Abstract

The goal of the ACTS Project SEMPER (Secure Electronic Marketplace for Europe) is to provide the first open and comprehensive framework for secure commerce over the Internet and other public information networks.

A prototype of the SEMPER Framework for Secure Electronic Commerce has been implemented in the Java programming language. It includes the payment systems SET, Chipper, Mandate and ecash™. The prototype uses a distinguished user-interface for trustworthy user in- and output which enables to use SEMPER on secure hardware.

This article describes the basic concepts of the SEMPER Framework for Secure Electronic Commerce as well as experiences gained in the field trials of the SEMPER software. In addition, we assess our achievements in comparison to more recent projects in electronic commerce.

1. Introduction

A wide range of businesses are rapidly moving to explore the huge potential of networked information systems, especially with the Internet-based WWW (World-Wide Web). Although the Internet has its roots in academia and is still dominated by free-of-charge information, dramatic changes are expected in the near future.

The goal of the 9-million ECU project *SEMPER* (Secure Electronic Marketplace for Europe) [5, 6] is to provide the first open and comprehensive solution for secure commerce over the Internet and other public information networks.

The members of the SEMPER consortium are *Commerzbank* (D), *Cryptomathic* (DK), *DigiCash* (NL), *EUROCOM EXPERTISE* (GR), *Europay International* (B), *FOGRA Forschungsgesellschaft Druck* (D), *GMD – German National Research Center for Information Technology* (D), *IBM* (CH, F), *INTRACOM* (GR), *KPN Research* (NL), *Otto-Versand* (D), *r^3 security engineering / Entrust Technologies* (CH), *CNET* (F), *SINTEF* (N), *Stichting Mathematisch Centrum / CWI* (NL), *Universities of Dortmund, Freiburg, and Saarbrücken* (D). Sponsoring partners are *Banksys* (B), *Banque Generale du Luxembourg* (LU), and *Telekurs* (CH). IBM Zurich Research Laboratory provides the technical leadership for the project.

1.1 Roles and Services in the Marketplace

As with a physical marketplace, the main purpose of an electronic marketplace is to bring potential *sellers* and *buyers* together:

- Sellers offer their goods and buyers order these goods; together this is a two-party negotiation, sometimes ending with an agreement.
- Sellers deliver their goods and buyers make payments; together this is a two-party (fair) exchange.

Buyers or sellers might be dissatisfied with what has happened so far, i.e., several exception handlers and dispute handlers which may involve an arbiter are necessary.

In all these actions, the parties have specific *security requirements,* namely integrity, confidentiality, and availability. Confidentiality includes anonymity, which is often a requirement for browsing catalogues or for low-value purchases. Examples of typical scenarios of electronic commerce are:

- Mail-order Retailing: A retailer accepts electronic orders and payments, based on digital or conventional catalogues, and delivers physical goods.
- On-line Purchase of Information and Subscriptions: Like mail-order retailing, but with digital, maybe copyright-protected goods that are delivered on-line.
- Electronic Mall: An organisation offers services for several service providers, ranging from directory services ("index") over content hosting to billing services.
- Contract Signing: Two or more parties exchange signed copies of the same statement.

Naturally, an open system for electronic commerce cannot be restricted to these scenarios. It should be easily configurable and extensible to a broad range of different scenarios.

1.2 What is New in SEMPER?

SEMPER is the first project that aims at the *complete* picture of secure electronic commerce, not just on specific pieces (like electronic payments), specific scenarios (like electronic on-line purchases) or specific products and protocols. (The relation of selected projects on electronic commerce to SEMPER is explained in more detail in Section 5. An overview on electronic commerce projects can be found at <http://www.semper.org/sirene/outsideworld/ecommerce.html>.)

SEMPER provides an *open framework* for electronic commerce. This includes a legal framework as described in [9] as well as a technical framework which is described in this article.

The technical framework enables the integration of any protocol and product providing the necessary services (see Section 3.6). Therefore, applications are not restricted to specific proprietary technology or specific protocols.

Another objective which distinguishes SEMPER from other projects is the concept of "multi-party security": SEMPER provides verifiable security and privacy to all concerned parties. Some consequences are:

- SEMPER users can verify their own security with only minimal trust into other parties. This is, e.g., done by providing evidence for all critical actions so that these actions can be disputed at an arbiter in case of faults.
- SEMPER developed an integrated anonymity management scheme extending the existing concepts for anonymous communication and credentials. This means, e.g., that the user can start anonymous business scenarios where all actions are guaranteed to be anonymous.

1.3 Overview

This article first describes the SEMPER model of electronic commerce which comprehends commerce as being a workflow of atomic transfers and fair exchanges of business items such as electronic goods. This is reflected by the SEMPER Framework for Secure Electronic Commerce described in the next section. The SEMPER Framework is structured in layers: The lower layers provide the busiess items, the transfers, and the fair exchanges. The higher layers provide generic workflows for the most common commerce scenarios together with means to configure them to the specific requirements of a particular user. Then, we describe our experiences with field-trials of our prototype. We conclude this article by a self-assessment which compares SEMPER with other more recent frameworks for electronic commerce.

2. Model for Electronic Commerce

The SEMPER Framework described in this paper is based on a generic model for two-party electronic commerce. This model describes the flow of control as well as actions and decisions for commerce services.

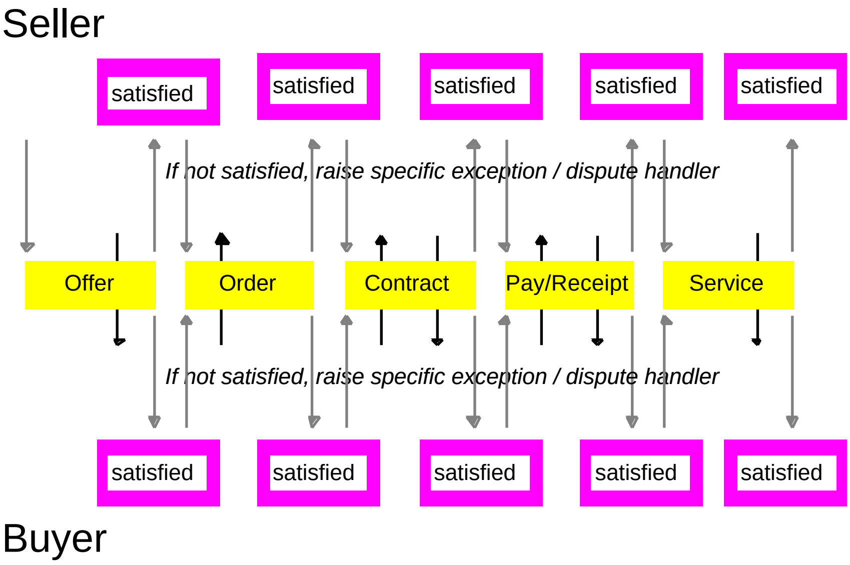


Figure 1Electronic Commerce is a Sequence of Transfers and Exchanges. Note that the protocol might enable other sequences as well, e.g., after "Contract" "Payment without Receipt" might also be enabled.

The main idea of the model is to describe business scenarios in terms of sequences of *transfers* and *fair exchanges* of business items with decisions based on the success of these actions (see Figure 1). This model is similar to the *dialogues* of interactive EDI.

2.1 Atomic Actions: Transfers and Exchanges

The main interactive actions between two players are *transfers* and *fair exchanges.* In a *transfer,* one party sends business items to one or more other parties. The sending and receiving parties can define certain security requirements, such as confidentiality, anonymity, or non-repudiation of origin. More generally, business items which can be transferred or exchanged include

- valuable information, such as a consultancy result, or program and video data,
- statements, such as signed documents or certificates, and
- money, such as credit-card, cash, or bank transfer payments.

A *fair exchange* is a simultaneous exchange of packages of business items typically among two parties. The parties have the *assurance* that their packages are sent if and only if the peer entity sends its package as expected. Either both packages are exchanged or none. If no fairness guarantee is required, we can model such an exchange by two transfers.

As an example of applying fair exchange imagine Bob has requested consultancy services from Alice, e.g., a piece of software, a translation, or a legal expertise. Alice wants to deliver to Bob a file containing the report. The file represents work of several person-months, so Alice wants a receipt if Bob receives the file. Bob, on the other hand, only wants to issue a receipt if he receives the file.

Figure 2 gives an overview of all possible exchanges of these primitive types. Transfers are included as exchanges of "something" for "nothing."

Transfer / Exchange *of* → *for* ↓	Money	Signature	Information
Nothing (i.e., Transfer)	Payment	Signature transfer etc.	Information transfer
Money	Fair money exchange	Fair payment with receipt	Fair purchase
Signature	*Same as* …	Fair contract signing	Certified mail
Information	*… in upper …*	*… right half*	Fair information exchange

Figure 2Transfers and exchanges of primitive types.

2.2 Electronic Commerce: Sequence of Exchanges

Using transfers and exchanges, a typical business scenario is modelled as a sequence of exchanges with user-interaction and local decisions between successive exchanges (see Figure 1).

In the course of an ongoing business, after each transfer or exchange, the parties are either

- satisfied, and thus willing to proceed with a certain number of other transfers or exchanges, or
- dissatisfied, in which case an exception or dispute is raised which might end up at a real court if all else fails.

This local decision depends on the success of the previous exchanges, the items received, and possibly user-input. After each round, a local decision as to whether and how to proceed is made. These sequences are similar to workflows of ordinary business transactions.

3. The SEMPER Framework

We now sketch the main services of the SEMPER Framework. After a short overview, we will sketch the layers bottom-up in more detail. For a complete description of SEMPER, we have to refer to the final documentation of the design [8] as well as the final report [9] containing a more fundamental view.

3.1 Overview

The SEMPER Framework (see Figure 3) is structured in layers. The lowest layer deals with low-level security primitives and other *supporting services*, whereas the highest layer deals with commerce issues only:

- The supporting services are the usual cryptographic services, communication, secure archiving of data (keys, non-repudiation tokens, audit trail), setting preferences, access control, and the trusted user interface “TINGUIN”. Furthermore, it provides secure communication services such as confidential, authenticated, or anonymous communication.
- The transfer layer provides services for transferring business items. This includes transfer-related security services such as non-repudiation of origin.
- The exchange layer supports fair exchange of business items.
- The commerce layer provides the local business sequences of our model which are locally executed by each player. Examples are sequences like “mail-order retailing,” “on-line purchase of information,” or “registration with service provider.” It is configurable by downloading new services or extending existing ones.

On top of these layers are so-called Business Applications. Business Applications are neither a layer nor part of SEMPER, but our name for any application that uses the SEMPER services. As Business Applications can be implemented outside

SEMPER, they are a priori untrusted and not allowed to perform security-critical actions without user authorisation.

Note that the security guaranteed by the layers gets larger and more abstract towards the top: The transfer layer only guarantees secure transmission whereas the commerce layer guarantees security of a whole commerce scenario.

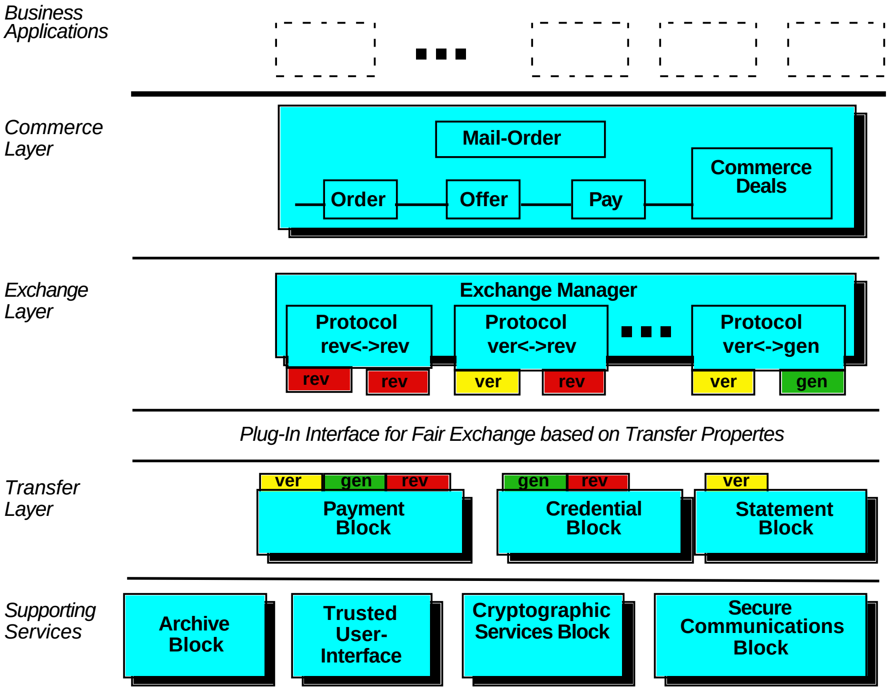


Figure 3The SEMPER Framework for Secure Electronic Commerce.

3.2 Supporting Services

Most *supporting services* as sketched above are not specific to electronic commerce. Except for a trusted user-interface "TINGUIN" (Trusted Interactive Graphical User-Interface), none of them was a primary objective of SEMPER.

The goal of the TINGUIN is to solve a common problem of Internet-based electronic commerce: In current electronic commerce sessions over the Internet the buyer typically interacts with the seller through his WWW-browser. As the HTML-pages presented in the browser are under the control of the Web-server (i.e., the seller), this exposes the buyer considerably. In particular, a message on such a page saying that a given offer from the seller has been signed correctly, cannot be trusted, as a crooked seller could simply put this on the HTML-page without validating the signed offer (in fact the seller may not have signed the offer at all). Another problem using the browser, is that any input supplied by the user may be read by the seller. This may lead to considerable problems if, for example, the buyer has to enter a PIN code or password in order to use an electronic purse or make a signature.

In our prototype, the TINGUIN is a secure and consistent user-interface in a fixed window which can be clearly distinguished from other (browser) windows. All normal service blocks of SEMPER interact with the TINGUIN block that provides this window, and they use some common look-and-feel elements that the TINGUIN Block provides. Thus, all security-critical in- and outputs of a user should be made via the TINGUIN. Together with the other SEMPER services, this enables the user to be sure that the outputs at the TINGUIN have been locally verified and to be warned that the inputs into the TINGUIN may have commercial consequences.

Naturally, this interface must be under control of the user's own device, and it must be difficult to fake the interface. The latter implies that the user must always be able to recognise the TINGUIN. Moreover, ergonomics has to be considered in the design of the TINGUIN even more carefully than usual, because any misunderstanding by the user may destroy security.

3.3 Transfer Service

The *transfer layer* provides services for packaging and trading business items. The basic items are electronic payments, credentials, and general statements which include digital signatures and data. These business items can be bundled into tree-like packages. The security attributes attached to each transfer determine the level of security which is required for the transfer or exchange of the transferred item.

Each type of business item is managed by a separate manager which provides unified services integrating existing implementations as described in Section 3.6.

Furthermore, the transfer services define interfaces of exchange-enabling properties used by the fair exchange protocols in Section 3.4.

3.4 Exchange Services

The *fair exchange services* developed by SEMPER are *optimistic* and *generic*. Optimistic means that a third party is only used in case of faults to restore fairness [3]. Generic means that the fair exchange protocols can be used to exchange arbitrary business items.

The reason why we developed *generic* fair exchange instead of using fair exchanges specific for each table cell in Figure 2 is that, e.g., a fair exchange protocol “payment for receipt” may work with one payment scheme but not with another. So instead of having at most nine different protocols, each new implementation of an item may require new fair exchange protocols. Furthermore, exchanging packages of items would require specific fair exchange protocols for any fixed combination of items to be exchanged. Thus, for a given number of *n* different kinds of electronic items this leads to n^2 different fair exchange protocols if one only wants to exchange one item for another. Furthermore, adding a new item to be exchanged (such as a new payment module) means adding another $n+1$ fair exchange protocols.

In order to achieve this independence of the actual items to be exchanged, we defined a minimal set of “exchange-enabling properties” which are required to be implemented by two transfers of business items in order to be usable in a fair exchange protocol:

> *External Verifiability*: The third party is enabled to check whether a transfer was successful or not. This can, e.g., be achieved by sending or re-sending the message via the third party.
>
> *Revocability*: The third party is able to undo a transfer (e.g., revoking a credit-card payment).
>
> *Generatability*: The third party is able to redo a transfer (e.g., signing a replacement receipt).

Based on transfers with these exchange-enabling properties of the two items to be exchanged, the so-called exchange manager negotiates with its peer which generic fair exchange protocol shall be used.

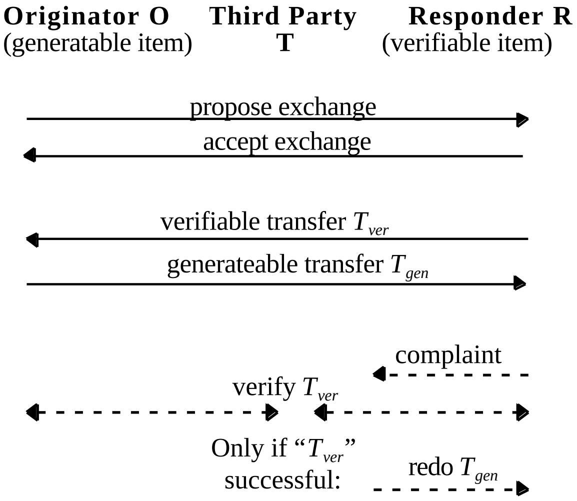


Figure 4: Fair Exchange of Generateable and Externally Verifiable Transfers.

An example of such a generic fair exchange protocol as used in SEMPER is depicted in Figure 4. The protocol can be used to exchange any generatable item for any externally verifiable item. It is similar to the protocol described in [3]: The basic idea is that the participants first agree on the exchange. If they agree, i.e., the descriptions of the expected items are fulfilled by the offered items, the responder transfers its item. If the item matches the expectation of the originator, the originator then sends its item as well. If the originator misbehaves and does not send its item, the responder complains at the third party which then produces an equivalent replacement for the item (this can be done since the second item was generatable) if and only if the first transfer was successful (this can be done using the external verifiability provided by the first item). A more detailed description can be found in [8, 9].

Following the same pattern, other protocols can be built which guarantee fairness if one of the items provides generatability and the other external verifiability or if both offer revocability.

3.5 Commerce Services

The *commerce layer* implements the flow of control of our model using the transfer and exchange service for interactions with the business partner, and the supporting services for user interaction and persistent storage. It also performs the trust management and access control necessary for downloading certified commerce services.

In order to provide overall security, the commerce layer sets up security contexts called “deals” which provide secure communication and signal certain commerce security attributes to all ongoing protocols. An example of a security service signalled by a context is anonymity which has to be guaranteed by all blocks participating in the deal.

Inside these deals, the commerce layer then runs the sequences of our model that directly implement protocols of business scenarios, e.g., how specific merchants or types of merchants handle customer registration and offering, ordering, payment, and delivery of goods. It implements the local flow of control, i.e., the enabled sequences of exchanges, of the electronic commerce model for each player separately. A set of client and server commerce services check each other. They are like an automatically verified electronic equivalent of the “terms of business” of the players. The commerce layer does not only offer entire commerce protocols, but also building blocks that may be of more general use, in particular services to manage and fill out standardised order forms.

Note that the commerce layer services are usually structured in a hierarchy: An offer/order or a payment/delivery building block can be used by a generic mail-order service as well as a service for selling database access. Thus, in principle, the commerce layer should provide transfer- and exchange-based work-flows implementing all common commerce scenarios.

Since one cannot fix the set of services in advance, the commerce layer includes services for secure downloading of services. This allows customers to participate in business scenarios they never encountered before. Since arbitrary terms of business may be implemented in a new commerce service, a downloaded service need not be secure at all. Therefore, downloading must be supplemented by evaluation and certification of downloadable services as well as proper access control.

3.6 Openness with Service Managers and Service Modules

Up to now, we have only described generic services of each building block without looking into the internal details of them. Now, we will describe the abstract concept guaranteeing openness of the implementation (see [1] for a detailed design based on this abstract concept).

Generic services are provided by so-called *service managers* and several *service modules* (see Figure 5). The union of a manager and its modules is called a *service block.*

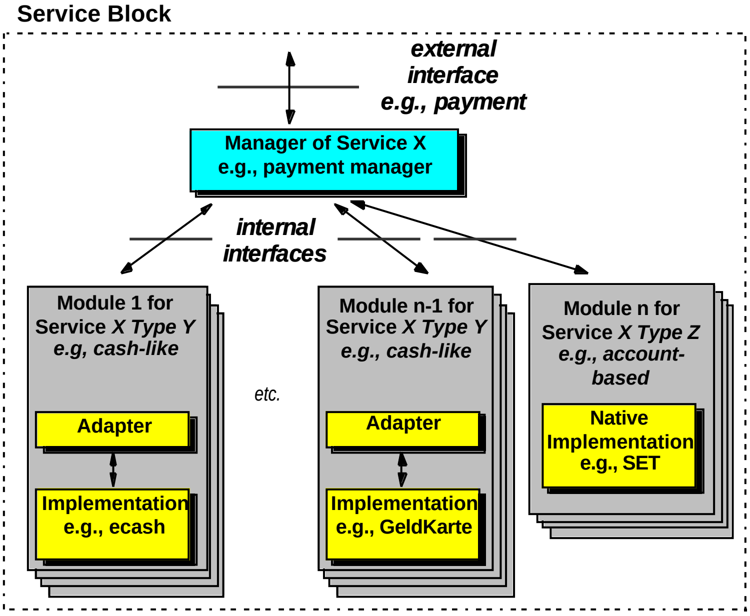

Service Block
external interface e.g., payment
Manager of Service X e.g., payment manager
internal interfaces
Module 1 for Service X Type Y e.g, cash-like
Adapter
Implementation e.g., ecash
etc.
Module n-1 for Service X Type Y e.g., cash-like
Adapter
Implementation e.g., GeldKarte
Module n for Service X Type Z e.g., account-based
Native Implementation e.g., SET

Figure 5: Open Services with Managers, Modules and Adapters[1].

The *service block* provides the generic, unified service, e.g., service = “payment”, which includes services for managing modules.

The generic service is based on a model of the service which should cover a broad range of protocols implementing this service, i.e., we have generic interfaces for a whole class of services.

For instance, the external interface of the payment manager is based on a *generic payment service* that will cover all kinds of *payment protocols* (or at most a small number of generic payment services, one for each payment model such as *account-based* or *cash-like*). Note that this does *not* mean that we will support only a few specific payment protocols, but that the interface definition is so general that *any* reasonable payment protocol can be accessed via that interface. If, e.g., company *XYZ* comes up with a new payment system *abc*, all they have to do in order to

[1] The last module is an example of a module written specifically for SEMPER and needs no adapter.

link *abc* into our architecture is to map the service interface of *abc* to the internal payment interface. This guarantees the desired openness of the marketplace.

A *service manager* provides a common interface to several modules together with methods for negotiation and selection of an appropriate module.

A *service module* corresponds to a protocol implementing the service, i.e., it more or less implements one entity of such a protocol. Its interface to the service manager is called an *internal interface*[2] of this service.

Having several modules per service allows different protocol implementations by different manufacturers. Service modules are said to be of the same type if their behaviour at the internal interface is the same. Examples of types of internal payment interfaces are “cash-like”, and “account-based”. Modules could be “SET”, and “e-cash” where “SET” implement the account-based model whereas “e-cash” implements the cash-like model.

As *SEMPER* wants to build on existing products as far as possible, we cannot assume that they all fit the same internal interfaces originally. Therefore, the interface of an existing implementation is enhanced by a *service adapter* so that the resulting module supports the required service.

The *SEMPER* architecture describes a fixed set of service managers. The set of service modules is not fixed, i.e., service modules can be dynamically attached to managers.

4. The SEMPER Trials

The trials are based on the *SEMPER* software prototype which implements the architecture and the basic security services. They are being conducted in various business contexts, both internal and external to the project. A map of the trial sites is depicted in Figure 6. Trial evaluation includes interviews of the trial users to measure the degree to which the software meets perceived security requirements, and to gain information about users’ levels of understanding and trust in security options.

4.1 Internal SEMPER Trials

In July 1997 the *SEMPER* prototype was tested by two service providers which are members of the project, Eurocom and Fogra. The Eurocom site, located in Athens (GR), offers distance learning services. Eurocom intends to use *SEMPER* to enable customers to browse their offering of courses, register and pay on-line and, subsequently, gain on-line access to the selected course presentation, notes, and examinations. Fogra, a research institute for the printing industry, located in Munich (D), offers its customers on-line ordering and delivery of documents and software.

[2] This is also often called a Service Provider Interface (SPI).

The ”Basic Trial” *SEMPER* software was used for the supervised trials which took place at Eurocom’s premises. The trials were integrated in a series of seminars for SME employees with the title *”Conducting Business over the Internet”*. After a brief presentation of the *SEMPER* architecture, the participants were able to run the *SEMPER* software and to make a purchase of a seminar from the Eurocom electronic store. Due to constraints of time, the *SEMPER* client software was pre-installed and user registration had also been performed prior to the trials. As a result, the participants experienced only the purse creation and purchasing procedures. 30 people participated in the Eurocom trials.

In general, the Greek trial participants didn’t think that the Internet is mature enough yet to perform business-to-business or business-to-customer transactions, not due to the technology itself, but due to its limited use in their customer base. The lack of a legal framework regarding the validity of electronic authentication was also viewed as an obstacle to performing important business transactions over the Internet. However, the general feeling was that these obstacles will be resolved and that electronic commerce has much to offer to small and medium-sized enterprises. The *SEMPER* software met most of the participant’s current business requirements for traditional commerce, apart from cheques as a form of payment. As a consequence, electronic cheques have been integrated into the advanced prototype.

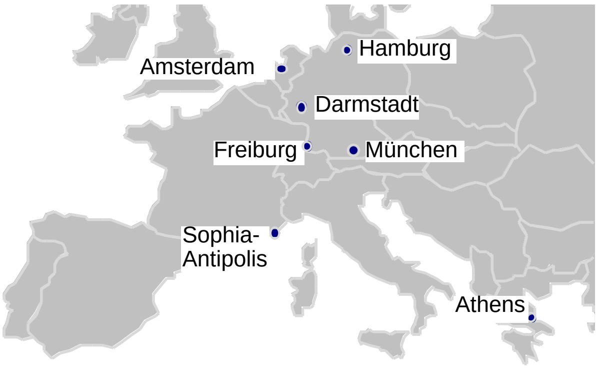


Figure 6: SEMPER trial sites.

Fogra demonstrated the *SEMPER* trial for three days in June 1997 at the IMPRINTA fair. Due to both software and access problems, trial of the scenario was only partially successful. Before releasing the *SEMPER* code for external testing, Fogra integrated a new version of the trusted user interface (*TINGUIN*), including status bars and other enhancements. Five persons from the Fogra customer base participated in ”unsupervised trials”.

The functionality and flexibility of the *SEMPER* architecture was greatly appreciated by the Eurocom and Fogra trial participants, but the state of the user interface was considered to be insufficiently developed for the ease of use to which non-specialists are accustomed. As a result, a new round of supervised trials, with participants selected on the basis of their networking experience, was conducted.

4.2 Supervised Basic Trial

In December 1997 the *SEMPER* software was installed at the Institute for Computer Science and Social Studies at the University of Freiburg, Germany and tested by 12 trial participants. Twenty hours of in-depth interview material was recorded and has been analysed as the basis of the trial report [7]. In order to obtain a more critical evaluation of the software, trial participants with extensive computing experience (and a minimum of 3 years Internet use) and a good awareness of security issues were selected. The participants subjected the prototype to particularly thorough testing, checking, for example, the software's response to incorrect input (random seed too short, incorrect password entry, attempting to obtain a second certificate from the CA, attempting to continue without inputting the requested information, rejecting offers, etc.) and were favourably impressed by its performance.

The Fogra business application and trial website, mentioned above, was used as the trial site. The trial bank was also run from the Fogra server. The Certification Authority (CA) was provided by the *SEMPER* CA at the GMD in Darmstadt, Germany. Each trial participant initialised the locally installed *SEMPER* software by completing the registration process and creating one or more purses. The registration procedure was based on the participant's personal data (name, organisation, city) which had been submitted to the certification authority prior to the trial in order to simulate off-line personal registration. The participant also entered a personal registration key which the CA had assigned to him/her.

Having completed the initialisation process, participants used the *SEMPER* software for their first ”semperised” experience of electronic commerce. They used the prototype to securely identify the Fogra website. They then browsed the Fogra website and selected a digital product (e.g., an abstract from the Fogra literature database). After filling out the order form on the Fogra website, they then requested that their local *SEMPER* software process this order securely. They obtained a digitally signed on-line offer from Fogra. They used their locally installed *SEMPER* software to send a digitally signed order to Fogra and used the purse function of the *SEMPER* prototype to make a (simulated) on-line payment. The abstract was delivered to the participant in the Netscape browser.

The trusted graphical user interface, *TINGUIN*, where all security relevant communications take place, is the visible and vital link between the user and the *SEMPER* software. As a result, it was subject to particular scrutiny. The credibility of the test for the participants was enhanced by the fact that it was possible for them to check the DOS (Java) window at all times during the test (and many did so). This ensured them that the test was actually *live*, i.e. that they were really exchanging certificates and business items with the CA and the bank and website servers in Munich.

4.3 SME Trials

Currently, external trials are being conducted by SMEs in Holland and France. They are also supported by the *SEMPER* basic security services and, in contrast to the previous trials, include real on-line payment and a test version of the SET purse. The OPL site (Oilfield Publications Limited; http://www.oilpubs.com/semper) offers books, maps, documents, and database access for the oil and gas industry. On-line payments using both credit cards and stored-value smartcards have been implemented. Two additional sites are located in Sophia Antipolis (F). ACRI (http://eurosud1.eurecom.fr/acrimall) provides access to a database of satellite images, processed and marked up with simulation. The second site is Acti-média (http://www.cyberlandpro.com), which sells CD-ROMs to the French speaking population world-wide. Simulated payments can be made by credit card or the test SET purse. These trials are currently concluded.

5. Self-Assessment

SEMPER is the only project aiming at securing electronic commerce as a whole. In addition, multi-party security including automated disputes has not been considered by any other project.

Our prototype implementation does not achieve product-level quality. Nevertheless, we were able to use it in our trials which helped us to validate and refine the SEMPER Framework.

Our model as well as our framework turned out to be a valuable tool for understanding and implementing electronic commerce. The design of some blocks in our framework (such as the payment block) is very detailed and extended the state of the art while some blocks may still gain from further refinement (e.g., the commerce and supporting services layer).

An indication for the quality of the basic concepts of our framework is that only minor changes to the SEMPER Framework were necessary during the course of the project. Furthermore, frameworks of other projects considering only parts of SEMPER integrate well into our framework:

- *Payment Projects:* Many projects aim at secure payments only [2]. During the course of the project, we integrated five fundamentally different payment schemes (eCash, iKP, SET, Mandate, and Chipper) without any problems. Therefore, we are convinced that we are able to integrate other payment schemes as well.
- *Java Electronic Commerce Framework [http://java.sun.com/commerce]:* This framework provides a more advanced design and implementation of our supporting services. It would be an ideal completion towards product-development based on the SEMPER Framework.
- *Open Buying on the Internet [http://www.openbuy.org]* considers business-to-business commerce only. It is based on EDI and could be used to extend the commerce layer.
- *Open Trading Protocol [http://www.otp.org]*, and *XML/EDI [http://www.xmledi .net]* define forms for business scenarios. They fit well into our concept of commerce-layer services: These forms can be used for interaction with the user and between commerce sequences on different machines.

- *CommerceNet's [http://www.commerce.net]* plans to define forms as well as processes for electronic commerce in their eCo Framework. Therefore, this would be another option for work-flows on the commerce-layer.

To conclude, we feel that a redesign of our prototype into a product based on the lessons learned and more recent implementations of some services should be worth the effort. Furthermore, we feel that other projects will gain by keeping our framework in mind even if they do not aim at the complete picture of electronic commerce.

6. Acknowledgements

This work was supported by the ACTS Project AC026, *SEMPER*. However, it represents the view of the authors only. The SEMPER deliverables can be obtained at the SEMPER homepage <www.semper.org>.

We would like to thank Birgit Pfitzmann for valuable comments which helped us to improve the paper. Furthermore, we would like to thank the anonymous reviewers.

References

[1] J. L. Abad-Peiro, N. Asokan, M. Steiner, M. Waidner: Designing a generic payment service; IBM System Journal 37/1 (1998) 72-88.

[2] N. Asokan, P. A. Janson, M. Steiner, M. Waidner: The State of the Art in Electronic Payment Systems; IEEE Computer 30/9 (1997) 28-35.

[3] N. Asokan, M. Schunter, M. Waidner: Optimistic Protocols for Fair Exchange, 4th ACM Conference on Computer and Communications Security, Zürich, April 1997, 6-17.

[4] M. Schunter, M. Waidner: Architecture and Design of a Secure Electronic Marketplace, Joint European Networking Conference (JENC8), Edinburgh, June 1997, 712.1-712.5.

[5] *SEMPER* Consortium: Basic Services: Architecture and Design; *SEMPER* Deliverable D03; Århus, October 1996.

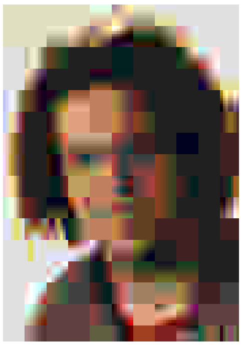
[6] *SEMPER* Consortium: Survey Findings, Trial Requirements, and Legal Franework -- Results from First Year of Project *SEMPER*; *SEMPER* Deliverable D05; Hamburg, December 1996.

[7] SEMPER Consortium: Evaluation of Phase II Trials; SEMPER Deliverable D09; Freiburg, July 1998.

[8] *SEMPER* Consortium: Architecture, Services and Protocols; *SEMPER* Deliverable D10; La Gaude, to be published in 1998.

[9] *SEMPER* Consortium: Final Public Report; *SEMPER* Deliverable D13; La Gaude, to be published by Springer-Verlag in 1999.

Autoreninformation



Matthias Schunter
Universität Saarbrücken
Institut für Informatik
Lehrstuhl für Kryptografie und Sicherheit
D-66123 Saarbrücken
Tel.: +49 (681) 302 5608
Email: schunter@acm.org

Dr. Michael Waidner
Information Technology Solutions
IBM Zurich Research Laboratory
Säumerstr. 4
CH-8803 Rüschlikon
Tel.: +41 (1) 724 82 20
Email: wmi@zurich.ibm.com

Dale Whinnett
Albert-Ludwigs-Universität Freiburg
Institut für Informatik und Gesellschaft
Abteilung Telematik
Friedrichstr. 50
D-79098 Freiburg
Tel.: +49 (761) 203 5700
Email: dalew@iig.uni-freiburg.de