SSONET – Diskussion der Ergebnisse[♣]

G. Wolf*, A. Pfitzmann**, A. Schill*, A. Westfeld**, G. Wicke**, J. Zöllner*

Technische Universität Dresden, 01062 Dresden
*Institut für Betriebssysteme, Datenbanken und Rechnernetze
**Institut für Theoretische Informatik
{g.wolf, pfitza, schill, westfeld, wicke, zoellner}@inf.tu-dresden.de

Zusammenfassung. Im Projekt SSONET (Sicherheit und Schutz in offenen Datennetzen) wurde eine Architektur für mehrseitige Sicherheit konzipiert, implementiert und an Beispielanwendungen validiert. Das vorliegende Papier stellt nach einer kurzen Beschreibung der Ziele des Projekts SSONET die erreichten Ergebnisse zusammen. Dabei werden alternative Konzeptions-, Design- und Implementierungsvarianten und ihre Auswirkungen erläutert und somit Entwurfsentscheidungen hinterfragt. Es werden ferner auch Probleme beschrieben, deren Lösungen nicht Ziel des Projektes waren, die aber interessant genug sind, um in weiterführenden Projekten bearbeitet zu werden.
Stichworte. mehrseitige Sicherheit, verteilte Kommunikation, Validierung

1 Einführung: Ziele und Annahmen

Ziel des Projektes SSONET (Sicherheit und Schutz in offenen Datennetzen) war es, eine Architektur für mehrseitige Sicherheit zu schaffen, die es Endbenutzern ermöglicht, ihre Ziele bei der Sicherung der Kommunikation mit verteilten Anwendungen zu formulieren und im Rahmen einer Verhandlung mit Kommunikationspartnern durchzusetzen. Das bedeutet einerseits, daß die Architektur eine Nutzerschnittstelle zur Verfügung stellen muß, mit der umzugehen sowohl Sicherheitsexperten als auch -laien verstehen oder erlernen können. Es bedeutet andererseits, daß Konzepte zum Abgleich verschiedener Nutzerinteressen auf der Ebene von Schutzzielen wie auch Sicherheitsmechanismen entwickelt und in die Architektur integriert werden müssen. Mit Hilfe solcher Konzepte muß selbst bei konfligierenden Schutzinteressen eine gemeinsame Kommunikationsgrundlage bzgl. Sicherheitseigenschaften gefunden werden können.
Weitere Ziele für SSONET waren Plattformunabhängigkeit, Modularität und Unabhängigkeit von Sicherheitsmechanismen-Implementierungen.

Um direkt an Lösungen der für dieses Projekt wesentlichen (oben genannten) Problemstellungen zu arbeiten, war es notwendig, manche Dinge vorauszusetzen oder aus den Betrachtungen auszuschließen. Es wurden folgende *Annahmen und Einschränkungen* gemacht:

1. Es wurde das Vorhandensein *lokal sicherer Endsysteme*, d.h. für ihren unmittelbaren Benutzer sicherer Endgeräte inklusive Betriebssystem, vorausgesetzt. Diese Voraussetzung zu erfüllen ist sehr aufwendig und sollte deshalb Inhalt eines separaten Projektes sein.
2. Die Betrachtungen und Entwicklungen des Projektes beschränkten sich auf Sicherheitseigenschaften und -mechanismen zur *Sicherung verteilter Kommuni-*

[♣] Diese Arbeit wurde finanziell unterstützt vom Bundesministerium für Bildung und Forschung (BMBF) und dem Bundesministerium für Wirtschaft und Technologie (BMWi).

kation. Es wurden also keine Arbeiten im Bereich Zugriffskontrolle, lokale Sicherung von Daten, etc. durchgeführt.

3. Es wird das Vorhandensein der notwendigen *Sicherheits-Infrastruktur* wie Schlüssel- und Zertifikatsserver vorausgesetzt. Außerdem wird davon ausgegangen, daß ein sogenannter SSONET-Server existiert, von dem Endbenutzer und Anwendungsentwickler Referenzen laden können wie zum Beispiel allgemeine Informationen über Sicherheitseigenschaften und -mechanismen, Expertenbewertungen für Sicherheitsmechanismen und Empfehlungen für anwendungsbezogene und anwendungsunabhängige Standardeinstellungen.

Im folgenden Kapitel 2 wird eine kurze Darstellung der im Projekt erarbeiteten Architektur gegeben, um dann in den nachfolgenden Kapiteln ausführlich auf die einzelnen Aspekte und getroffenen Designentscheidungen einzugehen.

2 Die SSONET-Sicherheitsarchitektur

2.1 Beschreibung der Struktur

Entsprechend der in Kapitel 1 beschriebenen Ziele besteht die Architektur aus folgenden wesentlichen Komponenten (vgl. Abbildung 1, ausführlicher in [PSWW2_99]):
Ein *Application Programming Interface (API)* ermöglicht die Einbindung von Sicherheitsmechanismen in auf der SSONET-Architektur aufsetzende Anwendungen.
Das *Security Management Interface (SMI)* bietet Endbenutzern die Möglichkeit zur Einstellung ihrer Schutzziele und präferierten Sicherheitsmechanismen für jede Teilaktion einer Anwendung. Mit diesen Teilaktionen wird jeweils eine Verbindung zu Kommunikationspartnern aufgebaut (sog. Kommunikationsaktion).
In der *Konfigurationskomponente* werden die Nutzereinstellungen gespeichert.

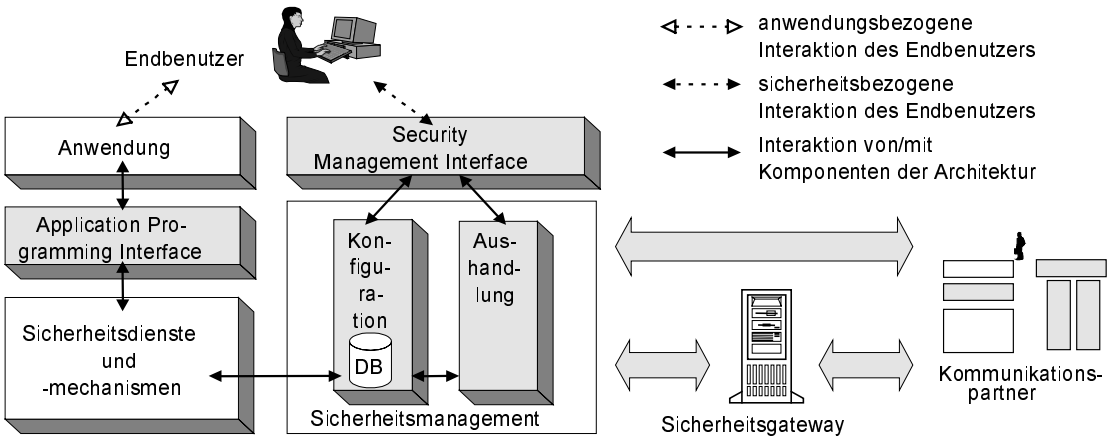

Abbildung 1: Aufbau der SSONET-Sicherheitsarchitektur

Die *Aushandlungskomponente* führt während des Verbindungsaufbaus zum Kommunikationspartner einen Abgleich der evtl. voneinander abweichenden Nutzerpräferenzen durch. Primärziel der Aushandlung ist die faire Ermittlung einer gemeinsamen Sicherheitsbasis für die Kommunikation der Teilnehmer. Sind die Schutzziele oder Mechanismenpräferenzen der Teilnehmer unvereinbar, so kann das Ergebnis der Aus-

handlung auch den bewußten Verzicht auf die Sicherung der Kommunikation oder gar auf die Kommunikation an sich bedeuten.
Einigen sich die Teilnehmer auf gemeinsame Schutzziele, besitzen aber keine gemeinsamen Sicherheitsmechanismen, so kann die Sicherheit der Kommunikation für manche Schutzziele über ein *Sicherheitsgateway* erreicht werden, das Umsetzungen zwischen den jeweils verwendeten Sicherheitsmechanismen vornimmt.
In SSONET sind *Schlüsselaustauschprotokolle* für symmetrische und asymmetrische Kryptoverfahren implementiert. Zugrundegelegt werden dabei Zertifikate nach X.509. Nach erfolgreicher Aushandlung der Kryptomechanismen für den Schutz der jeweiligen Anwendungsaktion werden die zugehörigen Schlüssel ausgetauscht und überprüft.
Das entwickelte System ist objektorientiert und plattformunabhängig durch Java™, was allerdings (noch) Einbußen in der Ausführungsgeschwindigkeit mit sich bringt. SSONET ist durch das API für beliebige Anwendungen einsetzbar und durch Kapselung der Mechanismen unabhängig von konkreten Mechanismen-Implementierungen. Es nutzt keine eigenimplementierten Sicherheitsmechanismen, sondern solche von existierenden Kryptobibliotheken wie der frei verfügbaren Cryptix 3.0.3 [Cryptix].

2.2 Beschreibung anhand von Merkmalen

In [WPSW_97] wurden einige Sicherheitsarchitekturen wie BirliX [HäKK_92], Kryptomanager [BaBl_96], Microsoft CryptoAPI [Wiew_96], SEMPER [Waid_96], DCE [Schi_97], CORBA [OMG_97], CISS [MPSC_93] sowie PLASMA [Kran_96] anhand der gegebenen Rahmenbedingungen, der angebotenen Funktionalität und Spezifika der Implementierung bewertet. Für einen direkten Vergleich zur SSONET-Architektur werden in den folgenden Tabellen 1 bis 3 die Ausprägungen in SSONET beschrieben.

Rahmenbedingungen	
Merkmale	**Ausprägung in SSONET**
Validierbarkeit: Wird eine Bewertung der Architektur hinsichtlich ihrer Sicherheitseigenschaften ermöglicht? Kann sich der Endbenutzer davon überzeugen, d.h. liegen Spezifikationen bzw. Quelltexte vor?	Validierung anhand von *Quelltexten* ist möglich, allerdings wird kein Konzept für die Verteilung der Quelltexte umgesetzt, z.B. Ladbarkeit von WWW-Server o.ä. Ggf. kann die Architektur neu übersetzt werden. Spezifikationen und Protokollabläufe liegen in Form von Projektberichten bzw. dem Lastenheft vor.
Sicherheitspolitik: Setzt die Architektur eine fest vorgegebene Sicherheitspolitik um oder ist sie politikneutral, d.h. erlaubt sie verschiedene Sicherheitspolitiken? Ist die Architektur offen für neue Schutzziele der Nutzer? (Dies hat z.B. Auswirkungen darauf, ob Instanzen gleichrangig bzw. gleich stark sein können oder nicht.)	SSONET verhält sich *politikneutral* und setzt somit das Konzept der mehrseitigen Sicherheit um, die allen Endbenutzern die Formulierung einer Politik entsprechend ihrer eigenen Schutzziele erlaubt. Die Eingabe von Politiken für Nutzergruppen (z.B. firmenintern) ist möglich. SSONET ist offen für Änderungen der Schutzinteressen der Nutzer. „Neue“ Schutzziele können vom Endbenutzer allein nicht integriert werden (bedarf der Implementierung).

Überprüfbare Rechteentstehung: Wenn Nutzer Rechte nicht nur verliehen bekommen, sondern auch selbst erzeugen und weitergeben können, ist solch ein Transfer dann überprüfbar und zum Ursprung zurückführbar? Welche Instanzen müssen kooperieren, um eine Zurückführung zu ermöglichen (z.B. keine, alle)?	*Autorisierung* wird in SSONET (bisher) *nicht betrachtet*. Erste Ansätze zur Rechteentstehung sind die integrierten fremd- oder selbsterzeugten Zertifikate.
Voraussetzung vertrauenswürdiger Instanzen: Muß eine bestimmte Infrastruktur vorhanden sein? Werden Instanzen zur Schlüsselzertifizierung, -verteilung oder -generierung vorausgesetzt?	Vorausgesetzt werden (nicht implementiert sind): – *SSONET-Server* zum Nachladen von Konfigurationen, Mechanismen und Mechanismenbewertungen, – *Zertifikat-Server* zur Zertifizierung und Verteilung von Schlüsseln.
Zertifizierung: Ist die Sicherheit der Architektur bezüglich bestimmter Kriterien zertifiziert worden?	Nein.
Standardisierung: Ist das Konzept der Architektur ein (internationaler) Standard?	Nein.
Referenzimplementierung: Existiert eine beispielhafte Implementierung der Architektur?	Ja, als *Prototyp*.
Marktverfügbarkeit: Handelt es sich - sofern eine Implementierung existiert - um ein frei verfügbares (public domain) oder ein kommerzielles Produkt?	Für Forschungszwecke frei *verfügbar*.

Tabelle 1: SSONET-Rahmenbedingungen anhand der Kriterien aus [WPSW_97]

Funktionalität	
Merkmale	**Ausprägung in SSONET**
Unterstützung spezieller Hardware: Kann durch die Architektur der Einsatz spezieller Hardware unterstützt werden?	Momentan ist *keine* Hardwareunterstützung implementiert. Spezielle Hardware wie z.B. ein Hardwaremodul für Sicherheitsmechanismen kann unterstützt werden.
Anonymität von Instanzen: Unterstützt die Architektur Anonymitätskonzepte; verhält sie sich diesbezüglich möglicherweise neutral?	Die *Benutzerschnittstelle* ist zur Integration von Anonymitätskonzepten vorbereitet; in der prototypischen Implementierung sind sie nicht umgesetzt. Es ist möglich, Zertifikate auf Pseudonyme auszustellen.
Sicherheitsmechanismen: Bietet die Architektur Sicherheitsmechanismen für symmetrische und asymmetrische Verschlüsselung, symmetrische und asymmetrische Authentisierung, Schlüsselaustausch, etc.?	Die Architektur bietet: – symm. und asymm. Verschlüsselung, – symm. und asymm. Authentisierung, digitale Signatur, – Schlüsselaustausch und -zertifizierung, – keine Zugriffskontrolle (bzw. Autorisierung)
Anpaßbarkeit: Ist die angebotene Funktionalität für eine Feinabstimmung auf spezielle Bedürfnisse verschiedener Nutzer geeignet?	Eine Anpaßbarkeit ist zum Teil durch Konfigurationsmöglichkeiten gegeben, wie etwa durch eine *private Konfigurationsdatei* des Endbenutzers. Weiterhin können Endbenutzer je nach Wissensstand Einstellungen zu Schutzzielen über die auszuwählenden Mechanismen bis hin zu den Details von Mechanismen vornehmen.

Tabelle 2: SSONET-Funktionalität anhand der Kriterien aus [WPSW_97]

Implementierung	
Merkmale	**Ausprägung in SSONET**
Schnittstellenspezifikation: Sind die Export- bzw. Importschnittstellen der Architektur erweiterbar? Herrscht z.B. Offenheit gegenüber neuen Kryptoverfahren (Import)? Werden zur Diensterbringung ausreichende Exportschnittstellen angeboten?	Schnittstelle zu Kryptoverfahren und API zu Anwendungen sind *erweiterbar*. Offenheit gegenüber neuen Kryptoverfahren wird durch *Adapterklassen*, ausreichende Exportschnittstellen durch sichere *Streams* erreicht.
Erweiterbarkeit: Können Module mit neuer Funktionalität aufgenommen werden?	Für die Aufnahme neuer *Module oder Klassen* sind keine softwaretechnischen Änderungen an der Architektur nötig.
Anforderungen an Hardware: Welche technischen Voraussetzungen, die über gängige Standardkonfigurationen hinausgehen, müssen erfüllt sein (z.B. manipulationssichere Hardware, Chipkartenleser)?	Ein *lokal sicheres Endsystem* wird jeweils als gegeben vorausgesetzt.
Verwendung von Standards: Sind standardisierte Kryptoverfahren und Datenaustauschformate in der Implementierung enthalten?	Die Architektur nutzt X.509-Zertifikate und standardisierte Kryptoverfahren wie DES und DSA.
Verwendung zertifizierter Bausteine: Wird in der untersuchten Architektur zertifizierte Hard-/ Software eingesetzt?	Nein, aber prinzipiell möglich.

Tabelle 3: SSONET-Implementierungsspezifika anhand der Kriterien aus [WPSW_97]

In [PiRi_94, S. 143f.] werden zur Einordnung von Architekturen die folgenden Kriterien genannt:

- **generisch ↔ spezifisch**: Generische Architekturen sind für alle informationstechnischen Systeme gleich gut anwendbar, bleiben dadurch aber abstrakt und oberflächlich. Spezifische Architekturen sind für ganz bestimmte Systeme maßgeschneidert.
- **top-down ↔ bottom-up**: Bei einem top-down-Ansatz werden die benötigten Sicherheitsfunktionen und -komponenten schrittweise aus den Sicherheitsanforderungen abgeleitet. Ein bottom-up-Ansatz beginnt mit der Untersuchung und Entwicklung von kryptographischen Algorithmen und Sicherheitsmechanismen, um ihren Einsatz in verschiedenen Systemen zu unterstützen.
- **konstruktiv ↔ analytisch**: Eine konstruktive Sicherheitsarchitektur beschreibt eine Methodik, um sichere Systeme zu entwerfen. Eine rein analytische Architektur kann beim Testen und bei der Evaluation der Sicherheit verschiedener Systeme unmittelbar behilflich sein.
- **voll integriert ↔ unabhängig**: Eine voll integrierte Sicherheitsarchitektur bildet einen nahezu untrennbaren Bestandteil der Systemarchitektur und wird gemeinsam mit dieser entwickelt. Eine unabhängige Sicherheitsarchitektur kann aus Bausteinen bestehen, die weitgehend beliebig integriert werden können.

Anhand dieser Kriterien kann SSONET folgendermaßen eingeordnet werden:
SSONET ist eine *generische* Sicherheitsarchitektur, allerdings spezifisch für die Sicherung von Kommunikationsverbindungen konzipiert. In SSONET wurde ein *top-down*-Ansatz verfolgt. Es wurde ausgehend von den Benutzerbedürfnissen eine Syste-

matik für die gezielte Auswahl der Mechanismen erarbeitet. Kryptographische Algorithmen wurden hingegen nicht entwickelt. SSONET beschreibt eine Vorgehensweise, um potentiell gegensätzliche Schutzinteressen zu sammeln und zur Laufzeit mittels Verhandlung einen Ausgleich herbeizuführen. Die Architektur ist in diesem Sinne als *konstruktiv* zu bezeichnen. SSONET ist insofern *unabhängig* vom Kommunikationssystem, da es eher neben einem Transportdienst steht, als daß es ins Netz integriert ist. SSONET ist außerdem nicht in Betriebssysteme integriert, ja nicht einmal auf ein spezielles zugeschnitten, aber in seiner Funktion und Sicherheit natürlich von der Sicherheit des verwendeten Betriebssystems abhängig (vgl. Annahme 1 in Kapitel 1).

In den folgenden Kapiteln wird eine selbstkritische Zusammenfassung der wesentlichen Konzeptionsentscheidungen und ihrer Auswirkungen im Projekt SSONET gegeben. Dabei wird insbesondere auf die Anwendungsanbindung sowie die Konfigurierung und Aushandlung von Schutzinteressen eingegangen.

3 Anwendungsanbindung

Im Projekt SSONET wurden zwei verschiedene Varianten für die Integration der SSONET-Funktionalität in Anwendungen implementiert: einerseits die direkte Integration über das API und andererseits die anwendungsunabhängige Protokollsicherung.

Anwendungsanbindung mittels API

Für die Anwendungsanbindung mittels API stellt die SSONET-Architektur Schnittstellen zur Verfügung, die der Anwendungsentwickler folgendermaßen nutzt:
Die durch die Architektur bereitgestellten Nutzerschnittstellen zur Eingabe anwendungsbezogener Schutzinteressen müssen mittels der Klasse *smi.AppConf.ApplicationConfiguration* in die Anwendung eingebunden werden. Dabei ist zu beachten, daß dieser Aufruf für jede Kommunikationsaktion der Anwendung auszuführen ist.
Der Anwendungsentwickler muß außerdem auf der Client-Seite der Anwendung eine Instanz von *SSONETClientConnection*, auf Server-Seite eine Instanz von *SSONETServerConnection* erzeugen. Zur Ausführungszeit verbindet sich die Server-Anwendung mit der Architektur und wartet auf die Anfrage eines Clients. Während der Instantiierung der SSONETClientConnection und dem Verbindungsaufbau erfolgt die Aushandlung mit dem Server. Für jede Klasse von Kommunikationsaktionen wird eine gesonderte Verbindung mit impliziter Aushandlung der Schutzziele und Mechanismen aufgebaut. Das Ergebnis der Aushandlung sind Datenströme mit zwischengeschalteten, konfigurierten Mechanismen. Die Mechanismen bleiben vor dem Anwendungsentwickler verborgen, sind jedoch für eine Statusermittlung abfragbar.

Anwendungsunabhängige Protokollsicherung

Die anwendungsunabhängige Protokollsicherung ist eine Möglichkeit, bei der auf SSONET aufsetzende Anwendungen im Gegensatz zur Anwendungsanbindung mittels API nicht modifiziert werden müssen. Das bedeutet, daß die Kommunikationssicherung transparent zu bestehenden Systemen hinzugefügt wird. Dies wird realisiert, indem die Anwendung ein virtuelles Gerät anspricht, für das die Konfigurierung

vorgenommen wird. Prinzip des Verfahrens ist, daß sowohl auf Server- als auch Client-Seite Adapter (vergleichbar mit Proxies) installiert werden, die die Sicherung der Kommunikation übernehmen (siehe Abbildung 2). Die eigentlichen, *unmodifizierten Server- bzw. Clientprogramme* verbinden sich statt direkt mit dem Gegenüber nur mit den Adapterprogrammen auf ihrem eigenen System. Dabei wird die Kommunikation mit der entfernten Gegenstelle wie auch im Beispiel oben mit SSONET-Streams abgewickelt, in die die lokal empfangenen Daten (z.B. Kommandostrings oder TCP-Pakete) eingebettet werden. Diese Teilkomponente ist an das jeweils zu unterstützende Protokoll anzupassen.
Für die anwendungsunabhängige Protokollsicherung müssen demnach *Adapter* implementiert werden, die an einem bestimmten Port warten und eingehende Daten über einen anderen Port ausgeben. Im Unterschied zur Anwendungsanbindung mittels API bleibt aber offen, wer dies implementiert. Einerseits kann es durch den Entwickler der Anwendung geschehen (der also zur Sicherung SSONET benutzen will), andererseits können auch SSONET-Entwickler für verschiedene Protokolle Adapter zur Verfügung stellen.

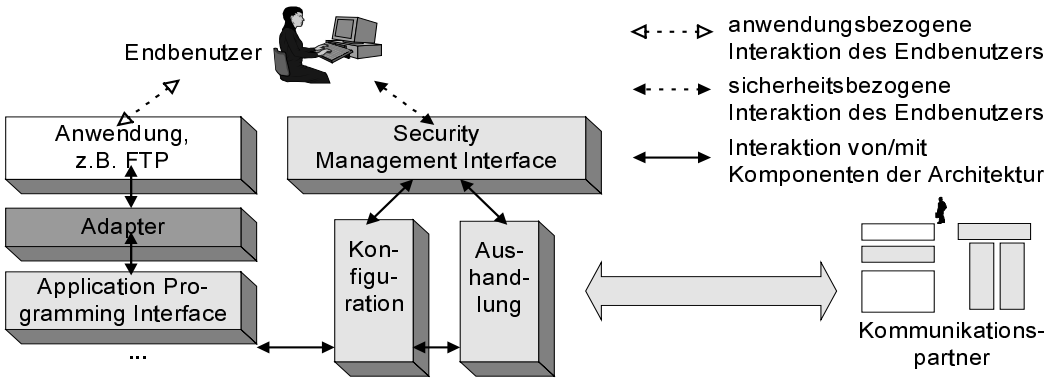

Abbildung 2: Der Adapter mit unveränderter Anwendungsschnittstelle

Folgender Unterschied ergibt sich für den Endbenutzer: Die Einstellungen für die Kommunikationssicherheit sind statt in der eigentlichen Anwendung in der Anwendungskonfiguration des Adapters vorzunehmen. Da der Adapter keine Unterscheidung zwischen den verschiedenen Zuständen des Systems macht, sondern nur die Protokollpakete umsetzt, existiert zwangsläufig nur *eine* Aktion, für die Schutzinteressen formuliert werden müssen und können, nämlich das Weiterleiten der Pakete an den entfernten Rechner.
Im Projekt wurde die anwendungsunabhängige Protokollsicherung am Beispiel von FTP realisiert. Die Architektur könnte auch weitere Protokolle wie SMTP, TELNET, etc. auf diese Weise unterstützen.

Zusammenfassung: Zur Anbindung von Anwendungen an SSONET (oder umgekehrt) bestehen zwei verschiedene Möglichkeiten. Für beide Varianten muß zusätzliche Implementierungsarbeit geleistet werden. Der wesentliche Unterschied besteht darin, daß die Anwendungsanbindung mittels API nur durch den Anwendungsentwickler ausgeführt werden kann. Außerdem sind die entwickelten Anwendungen

„SSONET-aware“, d.h., sie wurden für den Einsatz mit der SSONET-Architektur entwickelt oder zumindest erweitert und sind auch nur mit dieser verwendbar. Sie sind also von SSONET oder kompatiblen Architekturen abhängig.

Anwendungen	Verwendbarkeit	Aktionsbezug
SSONET-aware, z.B. Anbindung mittels API	nur angepaßte Anwendungen	aktionsbezogene Konfigurierung und Aushandlung
SSONET-unaware, z.B. anwendungsunabhängige Protokollsicherung	beliebige Anwendungen	kein Aktionsbezug bei Konfigurierung und Aushandlung, evtl. sogar Verlust des Anwendungsbezugs bei Verdeckung der Ports

Tabelle 4: SSONET-Awareness und ihre Auswirkungen

Bei der anwendungsunabhängigen Protokollsicherung müssen Adapter implementiert werden, die durch alternatives Anbieten bzw. Maskierung der Protokollschnittstelle den Anwendungen die Nutzung von SSONET ermöglichen bzw. sogar erzwingen. Es ist also keine Modifikation der Anwendung notwendig. Dies bedeutet nicht nur, daß die Anwendung „SSONET-unaware“ ist, sondern sogar, daß sie von SSONET unabhängig verwendbar bleibt. In Tabelle 4 werden die Ergebnisse zusammengefaßt.

4 Schutzziele, Anwendungsprotokolle und Mechanismen

Bei der Nutzung der SSONET-Architektur haben Endbenutzer die Möglichkeit, ihre eigenen Schutzinteressen zu formulieren. Konzipiert und implementiert sind in SSONET die Konfigurierungsschnittstellen und Aushandlungsmechanismen für:

- Schutzziele (Vertraulichkeit, Anonymität, Integrität, Zurechenbarkeit),
- Sicherheitsmechanismen (jeweils einer oder mehrere zur Umsetzung eines Schutzzieles),
- Mechanismendetails entsprechend den Mechanismen (wie etwa Schlüssellängen, Rundenzahlen oder Betriebsmodi).

Flexible Nutzung von Sicherheitsmechanismen

Die SSONET-Architektur bietet Flexibilität in bezug auf die integrierten Sicherheitsmechanismen. Der hauptsächliche Vorteil dieser Flexibilität besteht in der schnellen Anpaßbarkeit der zur Verfügung stehenden Mechanismen an aktuelle Entwicklungen, sei es die Existenz eines neuen Sicherheitsmechanismus oder der Ausschluß existierender Mechanismen aufgrund von Informationen über Schwächen oder Sicherheitslücken.

Der Nachteil anderer Lösungen ohne flexible Anbindung von Sicherheitsmechanismen liegt in der Notwendigkeit, die Schnittstellen oder die die Mechanismen benutzende Anwendung modifizieren zu müssen, und für den Endbenutzer darin, daß eine neue Programmversion (oder ein Patch) installiert werden müßte.

Die in SSONET erreichte Flexibilität bringt insofern mehr Arbeit mit sich, daß der Endbenutzer sich über aktuelle Entwicklungen (eben Verbesserungen oder entdeckte Sicherheitslücken in Algorithmen) informieren muß – und zwar nicht nur bzgl.

weniger fest vorgegebener Mechanismen, sondern bzgl. aller möglichen, zumindest aller von ihm ausgewählten. Dieses Problem könnte aber gelöst werden, indem sich die Architektur des Endbenutzers regelmäßig (z.B. einmal pro Woche) auf einem SSONET-Server einloggt und den Stand der aktuellen Entwicklungen lädt, und auch dementsprechend angepaßte Ratings, Regeln für Plausibilitätstests und neue Default-Werte für die Grundkonfiguration (siehe Kapitel 5). Das Laden der neuen Informationen kann mit den beim Endbenutzer verfügbaren Mechanismen gesichert werden.

Nutzung digitaler Signaturen

Digitale Signaturen leisten Sicherheit auf *mindestens zwei Ebenen*. Einerseits können sie die Zurechenbarkeit von Dateninhalten zu Personen und damit die Verantwortlichkeit der Personen für diese Dateninhalte sichern. Andererseits können digitale Signaturen Datenkontexte auf Verbindungsebene sichern. Da die SSONET-Sicherheitsarchitektur eine generisch benutzbare Schnittstelle zur Sicherung der Kommunikation auf Verbindungsebene anbietet, werden digitale Signaturen hier für das beweisbare Senden von aus Sicht von SSONET opaken Objekten verwendet. Diese signierten Objekte können auf Wunsch des Empfängers z.B. sequentiell in einer Datei abgespeichert werden. Der Anwender muß festlegen, wie lange digital signierte Daten aufbewahrt werden.

Eine weitere Konsequenz der Ansiedlung der Architektur auf Verbindungsebene ist, daß kein Konzept dafür geschaffen wurde, wie Anwendungen *signierte Dateninhalte anzeigen*. Empfänger von Nachrichten bekommen von der Architektur eine Fehlermeldung angezeigt, wenn der Signaturtest auf Verbindungsebene nicht erfolgreich war. Die Architektur kann nicht interpretieren, was signiert wurde, denn sie behandelt Daten unabhängig von der Anwendungssemantik. Nur die Anwendung selbst kann zu signierende Daten in ihrer Bedeutung darstellen. Ob die Darstellung authentisch ist, ob der Benutzer also genau das sieht, was er signiert, hängt von der Sicherheit der gesamten benutzten Hard- und Software ab. Indem signierte Nachrichten an der Oberfläche angezeigt werden, erhöht sich abhängig davon, ob der Nutzer noch Eingriffsmöglichkeiten hat, nicht die Sicherheit, in jedem Fall steigt aber das Bewußtsein beim Nutzer, mit Sicherheitsmechanismen umzugehen.

Um die Zurechenbarkeit von Dateninhalten zu ihren Sendern und die authentische Anzeige der zu signierenden bzw. signierten Dateninhalte zu gewährleisten, müßte SSONET eine Schnittstelle auf Anwendungsebene anbieten. Dies würde zu einer Erweiterung der Architektur auf mehrere Ebenen führen und hätte möglicherweise die Einschränkung der Wiederverwendbarkeit der Architektur zur Folge.

Protokolle zur Erreichung eines Schutzzieles

Bewußt ausgeblendet wurde in SSONET die *Konfigurierung und Aushandlung von Protokollen* bzw. der Zusammensetzung von *Protokollschritten* zur Erreichung eines Schutzzieles (wie zum Beispiel blinde Signaturen). Damit werden Sicherheitsprotokolle als fest vorgegeben betrachtet, und ihre Struktur ist nicht aushandelbar. Diese Vereinfachung reduziert die Komplexität der zu lösenden Problemstellung erheblich,

weil dadurch einfachere Modularität des Systems und eine einfachere Konzeption von Konfigurierung und Aushandlung möglich wurden.
Die Konfigurierung zusammengesetzter Protokolle würde einen hohen Wissensstand vom Endbenutzer fordern, und zwar nicht nur über einzelnen Sicherheitsmechanismen, sondern auch über das Zusammenwirken kryptographischer Verfahren und Algorithmen sowie organisatorisch-rechtlicher Rahmenbedingungen. Außerdem würde es von Entwicklern einer Sicherheitsarchitektur die Bereitstellung einer entsprechenden Schnittstelle, die alle Facetten eines zusammengesetzten Protokolls beachtet, erfordern. Wir schätzen es außerdem als sehr aufwendig ein, aus Teilmodulen zusammensetzbare Sicherheitsprotokolle konfigurierbar zu gestalten. Unsere These ist: Es gibt wenige auf diese Weise noch handhabbare Protokolle, und diese haben so viele spezifische Eigenschaften, daß sie vielleicht auch gleich als einzelne, „proprietäre“ Systeme umgesetzt werden könnten. Das heißt, der Wunsch nach einer generischen Architektur erscheint an dieser Stelle illusorisch.
Umgekehrt gilt: Wenn es gelänge, eine Strukturierung vorhandener Teilprotokolle vorzunehmen und eine Nutzerschnittstelle mit entsprechend zutreffenden Auswahlen an Protokollschritten pro Schutzziel zu erstellen, ergäbe dies eine essentielle Erweiterung des SSONET-Horizontes und brächte einen großen Fortschritt für die Aufbereitung und das Erlernen von Wechselwirkungen im Sicherheitsbereich mit sich.

5 Konfigurierung von Schutzzielen und Mechanismen

Das Security Management Interface (SMI, vgl. Abbildung 1) bietet dem Nutzer Oberflächen zur Eingabe und Modifikation seiner Schutzinteressen an. Dazu dienen sowohl die Fenster der *Grundkonfiguration* als auch der *Anwendungskonfiguration* (Abbildung 3). Die zuvor (von der Architektur bzw. dem Anwendungsentwickler) festgelegten *Anwendungsforderungen* stellen das Mindestmaß an notwendiger Sicherheitsfunktionalität für eine Anwendung (bzw. deren Aktionen) dar. Sind das lokale System und die Anwendung konfiguriert und startet der Endbenutzer die Kommunikation, so wird eine möglichst automatisch ablaufende Aushandlungsphase angestoßen, im Laufe derer sich die Kommunikationspartner auf eine Kommunikationsgrundlage (*Verbindungskonfiguration*) einigen.
In den folgenden Abschnitten wird auf besondere Aspekte der in SSONET implementierten Konfigurierung eingegangen. Für ausführlichere Beschreibungen und Screenshots des Ist-Zustandes der Konfigurationskomponente siehe zum Beispiel [PSWW_98], [WWWZ1_98] oder [PSWW_99].

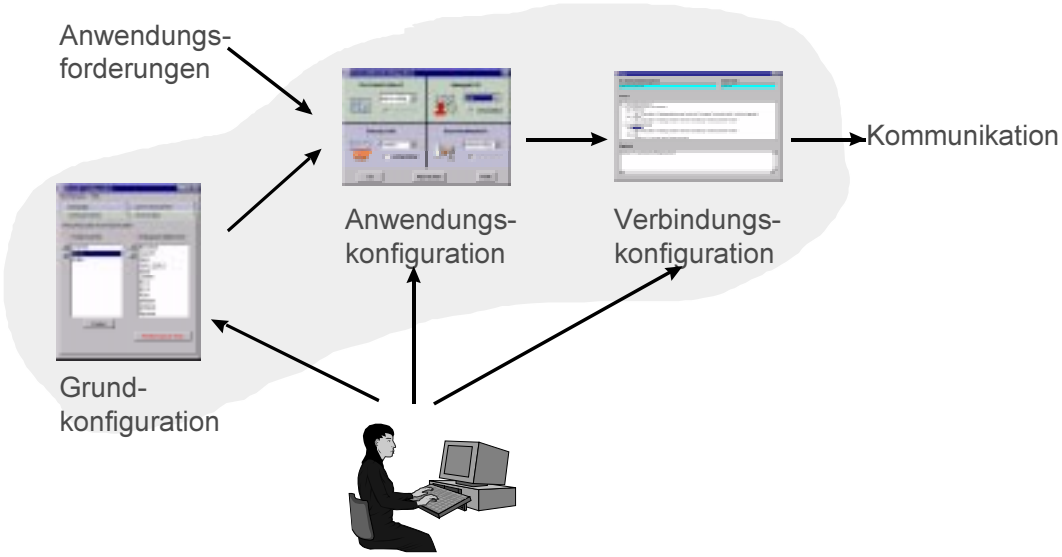

Abbildung 3: Wechselwirkung zwischen Anwendungsforderungen, Grund- und Anwendungskonfiguration [nach WWWZ1_98]

Der Anwendungsentwickler

Wie oben kurz erwähnt, soll der Anwendungsentwickler in der Lage sein, Anwendungsforderungen an die Sicherheit für eine Anwendung bzw. ihre Aktionen zu formulieren. Das erleichtert dem Endbenutzer die Konfigurierung, da aus Sicht der Anwendung unsinnige Einstellungen bereits aus der Nutzerschnittstelle ausgeblendet werden können. Insgesamt stehen in SSONET für die vier Schutzziele Vertraulichkeit, Anonymität, Integrität und Zurechenbarkeit fünf Abstufungen (unbedingt, möglichst, egal, wenn nötig, keinesfalls) zur Auswahl. Der Anwendungsentwickler hat die Möglichkeit, die Gewichtungen für Schutzziele durch die Angabe einer oberen und unteren Grenze dieses Raumes oder durch die Angabe eines Default-Wertes einzuschränken. Der eingeschränkte Auswahlraum ist eine nicht zurücknehmbare Vorschrift (mandatory security, siehe auch Kapitel 8); der Default-Wert ist ein durch den Endbenutzer überschreibbarer Vorschlag.

In SSONET nicht implementiert wurden Anwendungsforderungen für Mechanismen oder mechanismenbezogene Aspekte wie z.B. Performance. Dies wäre aber zumindest für manche Anwendungen mit Echtzeitanforderungen wie zum Beispiel Videokonferenzen eine durchaus lohnenswerte Erweiterung (siehe auch Abschnitt: Grund- und Anwendungskonfiguration).

Die Endbenutzer: Laien und Sicherheitsexperten

Eine Frage, die uns bei Präsentationen der SSONET-Konfigurierungskonzepte immer wieder gestellt wurde und beschäftigt hat, war: Nutzen Laien die Konfigurierungsmöglichkeiten überhaupt? Sind sie tatsächlich dazu in der Lage, benutzerdefinierte Einstellungen vorzunehmen, und entwickeln sie ein Interesse dafür?

Das Konzept der nutzerseitigen Konfigurierung zur Formulierung der Schutzziele setzt natürlich voraus, daß Endbenutzer mehr für die Sicherheitsproblematik sensibilisiert werden und daß das Sicherheitsbewußtsein wächst. Beschäftigt sich ein Mensch mit

neuer Funktionalität, muß er *dazulernen*[1]. SSONET bietet zwei grundsätzliche Formen der Unterstützung für Laien: einerseits Informationen über Sicherheitsmechanismen (wie zum Beispiel ein Performancetest, Mechanismenrating, Online-Hilfen), andererseits Standardkonfigurationen für Nutzer, die nicht selbst Entscheidungen treffen können oder wollen (Standards für Grundkonfiguration, Anwendungskonfiguration, Plausibilitätstest).
Eine Methode zur Unterstützung von Nutzergruppen mit unterschiedlichem Fachwissen ist die Einführung von sogenannten *User levels*. Beispielsweise wird bei der Nikon F50 Kamerabedienung ein Nutzermodus *simple* und *advanced* angewendet, wobei der simple-Modus Teile der Funktionalität der Kamera vor dem Nutzer verdeckt. Auch in [Oppe_94] werden zwei Nutzerstufen diskutiert und empfohlen. In SSONET ist das Problem der Unterstützung von Endbenutzern mit unterschiedlichem Fachwissen bewußt behandelt worden. Im Unterschied zu den zwei genannten Beispielen ist die Anzahl der Niveaustufen, die zwischen Laien und Expertenwissen unterscheiden, hier höher und hat fließende Übergänge. Für Laien wird z.B. von Mechanismendetails auf Mechanismen und danach auf Schutzziele abstrahiert, d.h. daß der Laiennutzer sich beispielsweise lediglich mit Schutzzielen beschäftigen kann, um sein System zu konfigurieren. Eine Designentscheidung in SSONET war, daß Konfigurierungsdetails, für die (tieferes) Expertenwissen notwendig ist, so für Laiennutzer zwar ausgeblendet werden, aber trotzdem zugänglich bleiben. Diese Funktionalität muß aber explizit als für Experten gedacht gekennzeichnet werden. Auf diese Weise kann ein Laiennutzer bei Kenntnisgewinn allmählich erweiterte Funktionalität benutzen und darüber selbst entscheiden.
Durch das Konzept der mehrseitigen Sicherheit wird eine Verringerung des Machtungleichgewichts möglich. Der Weg für Laien dorthin ist beschwerlich; um echten Einfluß nehmen zu können, müssen sie viel dazulernen. Sie müssen Schutzinteressen entwickeln und ein Mittel zur Durchsetzung ihrer Interessen in die Hand bekommen, das sie nach und nach kennenlernen und dessen Funktionalität sie verstehen und nutzen können. Dabei werden sie von Systemen wie der SSONET-Architektur unterstützt. Es ist klar, daß Nutzer nicht immer alle ihnen gebotenen Eingriffsmöglichkeiten nutzen werden. Aber mit SSONET wurde ein System geschaffen, das ihnen die Möglichkeit gibt, sich mit der Thematik „Sicherheit“ zu beschäftigen und dabei Wissen zu erwerben und Selbstbestimmung zu erlangen[2].

Grund- und Anwendungskonfiguration
Zwei Vereinfachungen, die in SSONET vorgenommen wurden, beziehen sich auf die Konfigurierung von Schutzzielen und Mechanismen. So können Endbenutzer in der Grundkonfiguration ihre Präferenzen zu Sicherheitsmechanismen global festlegen.

[1] Auch im Umgang mit anderen technischen Geräten muß man durch Erfahrung lernen.
[2] Mehr Selbstbestimmung bedeutet nicht zwangsläufig höhere Sicherheit. Falls Nutzer überfordert sind und trotzdem von sinnvollen Standardeinstellungen abweichen, kann das eine Abschwächung ihrer Sicherheit bedeuten. Hinweise über Wirkungen von vorgenommenen Änderungen müssen interaktiv gegeben werden.

Diese *Grundkonfiguration erben alle Anwendungen gleichermaßen*, d.h. es ist nicht möglich, für einzelne Anwendungen die Liste der Sicherheitsmechanismen zu modifizieren, wohl aber, die Wahl der Schutzziele anwendungsspezifisch vorzunehmen. Im ersten Abschnitt dieses Kapitels wurde schon erwähnt, daß für manche Anwendungen eine Auswahl von Sicherheitsmechanismen mit anderen Eigenschaften sinnvoll sein kann. Dies könnte zum Beispiel realisiert werden, indem bei der Anwendungskonfiguration nicht nur Einstellungen zu den Schutzzielen, sondern auch anwendungs- und/ oder aktionsbezogen Sicherheitsmechanismen ausgewählt werden können. Die Implementierung dieses Konzeptes wäre nicht sehr aufwendig. Es müßten allerdings Vererbungsstrategien formuliert werden, aus denen auch die Endbenutzer auswählen können. Ein Vorschlag zur Lösung wäre: Wenn die Mechanismenpräferenzen in der Grundkonfiguration geändert werden, erfolgt für jede konfigurierte Anwendung eine Abfrage, ob sie die neuen Einstellungen erben soll. Dann muß anhand einer Liste aller betroffenen Aktionen vom Endbenutzer entschieden werden, ob sie die neuen Einstellungen erben sollen.
Die zweite Vereinfachung (zumindest aus gesamtkonzeptioneller Sicht) besteht darin, daß Endbenutzer ihre Schutzziele nur aktionsbezogen innerhalb der jeweiligen Anwendungen formulieren. D.h. es gibt keine Möglichkeit, eine globale Grundeinstellung der Schutzziele vorzunehmen, aus der die einzelnen Anwendungen erben können. Im Rahmen des Projektes bleibt offen, ob es sinnvoll wäre, eine für alle Anwendungen gültige Definition der Schutzziele zu integrieren (globale Schutzziele in der Grundkonfiguration). Um den Endbenutzer trotzdem von zuviel Konfigurierungsaufwand zu entlasten, stehen die Konzepte Anwendungsforderungen (siehe oben) und Schutz- bzw. Aktionsklassen [Wolf_98] zur Verfügung.
Die Begründung für diese Vereinfachungen war die Reduzierung der Komplexität für den Endbenutzer. Natürlich geht mit dieser reduzierten Komplexität ein Verlust der Flexibilität (und auch der gebotenen Funktionalität) einher. Das alternative Konzept wäre mit Java implementierbar (Mehrfachvererbung durch Interfaces). Der Konzeptions- und Implementierungsaufwand für die Grundfunktionalität wird auf einen Personen-Monat geschätzt.

6 Aushandlung

Mit Hilfe der in SSONET implementierten Aushandlung wird auf Grundlage der in der Konfigurierung ausgedrückten Schutzinteressen der Teilnehmer eine gemeinsame Basis von Schutzzielen und Sicherheitsmechanismen ermittelt.
Aus den jeweiligen Nutzereinstellungen werden nach zuvor definierten Regeln „ausrechenbare“ Ergebnisse ermittelt. In den folgenden Abschnitten werden einige interessante Ergebnisse der Arbeit an Aushandlungskonzepten diskutiert; die grundsätzliche Vorgehensweise der Aushandlung in SSONET kann u.a. in [PSWW_98], [PSWW1_99] und [PSWW2_99] nachgelesen werden.

Schutz und Datensparsamkeit durch komplexe Aushandlungsprotokolle
Eine Methode des Datenschutzes ist Datensparsamkeit. Dazu gehört, daß eigene personenbezogene Daten nicht unnötigerweise preisgegeben werden. Für die Aushand-

lung müssen Kommunikationspartnern die eigenen Schutzinteressen übermittelt werden. In SSONET wurde diskutiert, inwiefern hierbei datensparsam vorgegangen werden kann. *Gegenüber Kommunikationspartnern* kann man im Aushandlungsprozeß versuchen, Informationen über die eigenen Interessen zurückzuhalten, um nicht alle Interessen zu Beginn aufzudecken. Allerdings kann bei Vertragsverhandlungen nicht „vermieden“ werden, eigene Interessen aktiv zu vertreten und zu offenbaren. Deshalb wird durch das teilweise Aufdecken der Schutzinteressen über den ganzen Aushandlungsprozeß gesehen nur eine Verzögerung der Preisgabe der Informationen erreicht. Dies führt also nicht zu mehr Datensparsamkeit, sondern allenfalls zu komplizierteren Protokollen und verlängerten Aushandlungszeiten.
Die Aushandlung in SSONET wurde so konzipiert, daß ein gewöhnlicher Nutzer während des automatischen Teils der Aushandlung keine Informationen über die Einstellungen des Partners erhält, da der Abgleich der Einstellungen im System erfolgt[3]. Nur im Konfliktfall kann er ableiten, wo der Partner vollkommen gegensätzliche Wünsche hat. Die Ergebnisse der Aushandlungsschritte sind im Statusfenster der Architektur anzeigbar.
Schutz der Aushandlung *gegenüber Dritten* (z.B. durch ein in allen SSONET-Installationen vorhandenes Basisset an kryptographischen Mechanismen) ist sinnvoll, da sonst Dritte die Aushandlung unbemerkt mitlesen oder sogar verfälschen können. Deshalb werden die Ergebnisse der Aushandlung zum Abschluß u.a. digital signiert ausgetauscht. Eine ausführliche Beschreibung des Protokolls zum Schutz der Aushandlung findet sich in [PSWW2_99].

Nutzerbezogener Umgang mit der Aushandlung
Ein Ziel in SSONET war, die Verhandlungen möglichst automatisch ablaufen zu lassen, um den Teilnehmern wenig interaktive Entscheidungen abzuverlangen. Dies wurde durch Kombination der folgenden Mittel erreicht: Einerseits wurde der Auswahlraum und die Strategie des Abgleichs so gestaltet, daß keine unnötigen Konfliktfälle auftreten. Andererseits besteht eine *Wechselwirkung zwischen vorab geleisteter Konfigurierungsarbeit und automatischer Aushandlung*: Je mehr Parameter der Aushandlung bereits vorher konfiguriert wurden, desto weniger muß der Endbenutzer mit interaktiven Abfragen während der Aushandlung belästigt werden. Es ist also sinnvoll, den Teilnehmern im Vorfeld Konfigurierungsarbeit für eventuell eintretende Sonderfälle abzuverlangen. Manche Aspekte wie zum Beispiel partnerbezogene Entscheidungen sind nicht einfach vorab entscheidbar. In solchen Fällen läßt sich eine Interaktion zur Aushandlungszeit nicht vermeiden (es sei denn, man kann Wünsche bezüglich aller potentiellen Kommunikationspartner vorher abfragen). In SSONET werden partnerbezogene Entscheidungen auf Schutzzielebene ermöglicht, indem der Endbenutzer für eine Aktion konfigurieren kann, ob ein Schutzziel partnerspezifisch verhandelbar ist. Aufgrund dieser Voreinstellung wird er dann im Konfliktfall während

[3] Ein versierter Nutzer kann natürlich Wege finden, um sich die systeminternen Daten anzusehen.

der Aushandlung gefragt, ob er mit diesem Partner von seiner Einstellung abweichen möchte.
Eine weitere interessante Fragestellung ist, ob die *Wiederverwendung* einmal ausgehandelter Verbindungskonfigurationen möglich ist und sogar Effizienzverbesserungen mit sich bringt. Hier ist also die Geltungsdauer von Aushandlungsergebnissen von Interesse. In SSONET werden Aushandlungsergebnisse (Verbindungskonfigurationen) nicht abgespeichert, sondern direkt nach der ihnen entsprechenden Sicherung der Kommunikation, d.h. beim Schließen des Streams, „vergessen“. Es wäre denkbar, daß bei Verbindungsaufbau zuerst getestet wird, ob für diese Aktion mit diesem Kommunikationspartner bereits eine erfolgreiche Aushandlung durchgeführt wurde. Ist dies der Fall, bleibt zu klären, ob deren Ergebnis aus Sicht beider Teilnehmer noch gültig ist. D.h., daß Aushandlungsergebnisse maximal so lange gespeichert werden müßten, bis einer der Teilnehmer sie ändern will, und minimal so lange, wie es sich beide Menschen merken. Offen und Gegenstand weiterer Arbeiten ist, ob nachfolgende Aushandlungen auf der Basis von gespeicherten Verbindungskonfigurationen optimiert werden können. Organisiert werden könnte die Abspeicherung der Verbindungskonfigurationen am besten in einer Art Adreßbuch, wobei wieder zwei Varianten realisierbar wären: entweder ein primär anwendungsbezogenes und sekundär personenbezogenes Adreßregister oder genau umgekehrt. Entscheidend ist hier die anwendbare Konkretisierung, d.h. ob anwendungs- oder partnerbezogen mehr Ausnahmen gemacht werden.

Verallgemeinerung der Aushandlung auf mehr als zwei Teilnehmer
Die SSONET-Aushandlung bietet Lösungen für die Verhandlung der Schutzinteressen zweier Kommunikationspartner. Diese Einschränkung war zunächst auch realistisch, da viele Kommunikationsschritte in Anwendungen genau zwei Teilnehmer betreffen. Durch die so reduzierte Komplexität konnte konzentrierte Arbeit an den grundlegenden Aushandlungsmechanismen geleistet werden. Für ein weiterführendes Projekt, das auch Lösungen für komplexere Anwendungen z.B. mit Multicast-Mechanismen erarbeitet, muß es aber ein Ziel sein, Verhandlungsprotokolle wie die aus SSONET auf mehr als zwei Teilnehmer zu verallgemeinern und die Auswirkungen auf organisatorische Rahmenbedingungen zu analysieren. Weiterhin sind auch die Wechselwirkungen *einer tri- oder multilateralen Aushandlung* mit der Konfigurierung (Kapitel 5) und den Sicherheitsgateways (Kapitel 7) zu diskutieren.

7 Sicherheitsgateways

SSONET-Sicherheitsgateways sind Instanzen, die in der Lage sind, zwischen verschiedenen Verfahren und Mechanismen zur Kommunikationssicherung zu konvertieren. Diese Gateways lösen keine Unstimmigkeiten bezüglich der Schutzziele der Teilnehmer, sondern erweitern die Möglichkeiten zur technischen Umsetzung der Schutzziele. Das bedeutet, daß mit Hilfe eines Sicherheitsgateways eine gesicherte Kommunikation ermöglicht werden kann, obwohl beide Teilnehmer disjunkte Sicherheitsmechanismen zur Verfügung haben.

Intermediäre und lokale Sicherheitsgateways

SSONET-Sicherheitsgateways können (mindestens) in zwei Lokationsvarianten realisiert werden: einerseits als *intermediäres* Gateway, das zwischen den Teilnehmern positioniert ist und während der Kommunikation Umsetzungen zwischen den verschiedenen Mechanismen vornimmt; andererseits als *lokales* Gateway, das vom Sender oder Empfänger unilateral zu Hilfe genommen wird, um zu sendende oder empfangene Nachrichten sicher konvertieren zu lassen. Beide Gatewayvarianten können sowohl vom Absender als auch vom Empfänger angewendet bzw. in ihrem Vertrauensbereich plaziert werden. In Tabelle 5 wird eine Übersicht über Vor- und Nachteile gegeben.

Intermediäres Sicherheitsgateway	Lokales Sicherheitsgateway
Niedrigerer Nachrichtenübertragungsaufwand: Die Nachricht muß nicht extra zum Gateway und zurück transportiert werden. Dies kann insbesondere bei zeitkritischen Anwendungen (z.B. Videokonferenz) relevant sein.	*Selbstbestimmung*: Teilnehmer können Gateways einschalten, ohne mit dem anderen darüber zu verhandeln. Aus der Sicht des einen Teilnehmers wird das gewünschte Verfahren vom anderen direkt unterstützt.
Geringere Flexibilität: Intermediäre Gateways sind weniger flexibel einsetzbar: Bereits vor Beginn der Kommunikation muß klar sein, daß ein Gateway benutzt wird.	*Transparente Einbindung*: Es sind keine Änderungen in der Kommunikation zwischen Sender und Empfänger notwendig.
Modifikation der Verbindung: Mindestens die Kommunikationsadressen, meistens auch die Datenformate unterscheiden sich von denen einer Verbindung ohne Sicherheitsgateway.	*Höhere Sicherheit*: Der Empfänger erhält bei Einsatz eines lokalen Gateways die Originalnachricht, solange nicht ein man-in-the-middle-attack in Zusammenarbeit mit dem Gateway erfolgt. Ein intermediäres Gateway kann unerkannt modifizieren.
Bekannte Technik: Intermediäre Gateways lassen sich (bei reduzierter Selbständigkeit des Endsystems) wie ein Proxy realisieren, wobei man auf bekannte Verfahren zurückgreifen kann und diesen proxy-ähnlichen Dienst z.B. in ein Firewallsystem integrieren könnte.	*Vereinfachte Abrechnung*: Die möglicherweise notwendige Abrechnung vereinfacht sich, weil der Dienst des Gateways eindeutig einem Teilnehmer zurechenbar ist.
	Flexiblerer Einsatz: Das lokale Gateway ist flexibler einsetzbar. Die Teilnehmer müssen das Gateway nicht immer benutzen, sondern können es je nach Anwendungsfall einschalten (z.B. kann möglicherweise der Test bei digitalen Signaturen unter eher irrelevanten Nachrichten entfallen oder auf später verschoben werden).

Tabelle 5: Jeweilige Vor- bzw. Nachteile der Gateway-Lokationsvarianten

Folgender Unterschied zwischen den Lokationsvarianten wird deutlich: Intermediäre Sicherheitsgateways vermitteln bei Vorhandensein disjunkter Sicherheitsmechanismen bei den Teilnehmern. Das leistet auch das lokale Sicherheitsgateway; allerdings kann es darüber hinaus auch Unterstützung bieten, wenn ein Teilnehmer *keine* Mechanismen für ein Schutzziel besitzt. Dies trifft für alle Schutzziele zu.

Aus Architektursicht werden über die (sowohl intermediären als auch lokalen) SSONET-Sicherheitsgateways realisierte Schutzziele in der Konfigurierung als weiterer auswählbarer Mechanismus implementiert.

Umsetzbare Schutzziele

Bei der Diskussion der durch ein Gateway umsetzbaren Schutzziele (bzw. Sicherheitsmechanismen) wurden zunächst nur diejenigen betrachtet, bei deren Einsatz das

Gateway kontrollierbar bleibt. Also sind SSONET-Sicherheitsgateways zunächst auf Integrität und Zurechenbarkeit beschränkt.
Wird der Vertrauensbereich der Endbenutzer auf das Sicherheitsgateway erweitert, können auch Mechanismen für Schutzziele wie Vertraulichkeit und Anonymität umgesetzt werden (vgl. Tabelle 6). Hiermit weichen wir die Annahme 1 aus Kapitel 1, daß ein sicheres Endsystem vorausgesetzt wird, das erste Mal etwas auf. Wir gehen nicht mehr zwingend davon aus, daß ein Benutzer der SSONET-Architektur Mechanismen zur Umsetzung seiner Schutzinteressen auf seinem *eigenen* System vorhanden haben muß, sondern daß ein sicheres Endsystem gemeinsam mit lokal und extern vorhandenen Sicherheitsmechanismen Sicherheit in verteilten Systemen ermöglicht. Eine Auswirkung dieser Abschwächung ist, daß der Vertrauensbereich des Endbenutzers erweitert werden muß (siehe Tabelle 6).

Schutzziel: Transformation	Wirkung auf den Vertrauensbereich
Integrität: Symmetrische Authentikation ➔ Symmetrische Authentikation	Der Vertrauensbereich muß auf das Gateway erweitert werden. Betrug ist feststellbar, wenn das Gateway nicht zu jedem Zeitpunkt alle Kommunikationswege zwischen den Partnern kontrollieren kann.
Integrität und Zurechenbarkeit: Dig. Signatur ➔ Dig. Signatur	Dem Gateway muß nicht vertraut werden, da jeder Mißbrauch Dritten beweisbar ist.
Vertraulichkeit: Konzelation ➔ Konzelation	Der Vertrauensbereich muß auf das Gateway erweitert werden, denn das Gateway erhält Kenntnis der geheimen Nachricht; unerlaubte Weiterverbreitung durch das Gateway ist möglich.

Tabelle 6: Eine Auswahl umsetzbarer Schutzziele und nötiger Vertrauensbereiche

Durch die in SSONET integrierten Sicherheitsgateways kommen Diskussionen über Schnittstellen zu rechtlichen und organisatorischen Rahmenbedingungen ins Spiel. Es wird somit auf Mechanismenebene mehr als nur der kryptographische Mechanismus betrachtet.

8 Grenzen der Selbstbestimmung bezüglich Sicherheit

Eigen- und fremdbestimmte Sicherheitspolitiken
Die SSONET-Architektur wurde unter dem Blickwinkel der mehrseitigen Sicherheit – also einer *eigenbestimmten Sicherheitspolitik (discretionary policy)* – entwickelt. Menschen sind in ihrem Handeln jedoch nicht immer völlig autonom, sondern in Organisationen eingebunden. Deshalb wird in diesem Abschnitt diskutiert, was SSONET in Anwendungsbereichen *fremdbestimmter Sicherheitspolitiken (mandatory policies)* leisten kann. Es gibt in SSONET bereits manche Aspekte für die hierarchische Festlegung fremdbestimmter Sicherheitspolitiken: Der Anwendungsentwickler kann beispielsweise die für den Endbenutzer auswählbaren Präferenzstufen für Schutzziele einschränken (siehe Kapitel 5). Eine rechtliche Regelung, die für eine Anwendung einen bestimmten Signaturmechanismus vorschreibt, wäre mit SSONET derzeit allerdings nicht durchsetzbar. Fremdbestimmte Sicherheitspolitiken sind mit SSONET also formulierbar, deren Einhaltung ist aber nicht erzwingbar. Dazu müßte der SSONET-

Quellcode so modifiziert werden, daß eine Rolle (z.B. der Anwendungsentwickler oder Vorgesetzte) Vorgaben machen kann, die der Endbenutzer (z.B. ein Mitarbeiter) nicht mehr modifizieren kann. Das unterstellt natürlich gleichzeitig, daß der Mitarbeiter auch die SSONET-Quellen nicht modifizieren kann. Diese Forderung wäre realisierbar mit einem authentischen Boot-Prozeß, in dem digitale Signaturen Hard- und Software authentisieren. Unter der Annahme, daß das möglich wäre, wären fremdbestimmte Sicherheitspolitiken mit SSONET durchsetzbar. Mit den Konzepten für mehrseitige Sicherheit kann man also auch *mandatory security* ausdrücken, aber nicht unbedingt durchsetzen.

Kommunikation in und zwischen Organisationen

Die Kommunikation *zwischen autonomen Organisationen* oder Organisationseinheiten mit eigenen Sicherheitspolitiken (z.B. Profitcenter) ist hinsichtlich der Nutzung der SSONET-Architektur weitestgehend mit der durch Individuen vergleichbar. Das heißt also, daß in diesem Fall die SSONET-Konzepte für Konfigurierung und Aushandlung nahezu unverändert nutzbar sind.

Natürlich wird die von SSONET gebotene Funktionalität nur beschränkt genutzt, wenn ein oder mehrere Teilnehmer eine fremdbestimmte Politik vertreten (müssen). Das bedeutet dann, daß diejenigen nur die durch die Politik vorgegebenen Spielräume nutzen können; im Extremfall steht z.B. nur ein Mechanismus für Verschlüsselung zur Verfügung. Eine Aushandlung und sichere Kommunikation auf dieser Basis ist immer noch möglich. Je weniger Spielraum die fremdbestimmten Sicherheitspolitiken jedoch lassen, desto höher ist die Gefahr, daß zwischen den Teilnehmern keine gemeinsame Basis zur Kommunikationssicherung gefunden werden kann.

Ist die entwickelte Architektur auch für *firmeninterne Bereiche* sinnvoll anwendbar? Oben wurde schon gezeigt, wie die SSONET-Architektur in großen Organisationen mit dezentralem Management oder für Organisationen mit mehreren Profitcentern genutzt werden kann. Bieten sich auch Möglichkeiten zur Anwendung *innerhalb von Hierarchien* – also die Aushandlung zwischen Zielen von Vorgesetzten (und damit der Firmenpolitik) und den Zielen einzelner? Hier ist es wichtig, die verschiedenen Rollen zu identifizieren, in denen Entscheidungen getroffen und ausgeführt werden. Es müssen organisatorische Fragen geklärt werden wie: Muß ein Mitarbeiter mit seinem Vorgesetzten aushandeln? In welchen Anwendungsbereichen kann das Ergebnis der Aushandlung mehr bedeuten als die Durchsetzung der Vorschriften des Vorgesetzten? Muß man sich Aushandlungen mit anderen Mitarbeitern vom Vorgesetzten (vor- oder nachher) genehmigen lassen? Berücksichtigt die Aushandlung die Regeln des Betriebsverfassungsgesetzes?

Zusammenfassend läßt sich feststellen, daß das Konzept der Konfigurierung und Aushandlung zumindest über Firmengrenzen, je nach Organisation auch über Abteilungen oder kleinere Bereiche mit verschiedenen Sicherheitspolitiken anwendbar ist, aber nur eingeschränkt zwischen Einzelnutzern innerhalb einer Organisationseinheit. Sicherheitsgateways (siehe Kapitel 7) können, gekoppelt mit Firewalls, als Teil einer fremdbestimmten Sicherheitspolitik (mandatory policy) fungieren, wenn sie sich nicht

beliebig den Wünschen der Endbenutzer anpassen und die Firewalls Kommunikation, die die Gateways umgeht, nicht zulassen.

9 Zusammenfassung und Ausblick

SSONET ist ein Projekt, in dem *eine* Möglichkeit zur Konzeption und Implementierung einer Architektur für mehrseitige Sicherheit umgesetzt wurde. Es wurden neue Konzepte entwickelt, wobei zur Reduzierung der Komplexität teilweise vereinfachte Varianten implementiert wurden. Insgesamt sehen wir SSONET als einen Schritt auf dem Weg zur Entwicklung mehrseitig sicherer Anwendungen und zur Sensibilisierung der Endbenutzer für die Problematik von IT-Sicherheit, indem Spielräume und Grenzen aufgezeigt wurden.

Als wichtigstes Ergebnis des Projektes SSONET wurde gezeigt, daß mehrseitige Sicherheit mit verhältnismäßig geringem Mehraufwand für Anwendungsentwickler und Endbenutzer umsetzbar ist. Es wurde eine *einfache* Möglichkeit geschaffen, sichere Verbindungen aufzubauen, die plattformunabhängig und unabhängig von konkreten Mechanismen-Implementierungen ist.

Aufsetzend auf den erreichten Ergebnissen könnten u.a. zwei Dinge zur Vervollkommnung des Projektes beitragen. Wichtig wäre, einen Feldversuch mit Anwendungsentwicklern und Endbenutzern durchzuführen, um die in begrenztem Umfang u.a. mit Psychologen diskutierten Konzepte und Implementierungen in der Praxis zu testen. Ein zweiter wichtiger Schritt für Systeme wie SSONET ist die Integration in Anwendungen, die am Markt etabliert sind. Dadurch ist es möglich, Resonanz zu erhalten und Benutzer für Sicherheitsprobleme und vorhandene Lösungen zu sensibilisieren.

Sicherlich konnten im Rahmen des vorliegenden Papiers nur manche Teilprobleme beschrieben werden. Wir hoffen trotzdem, daß zukünftige Projekte mit ähnlichen Zielsetzungen von den beschriebenen Ergebnissen profitieren können.

Literatur

BaBl_96 T. Baldin, G. Bleumer: CryptoManager++: an object oriented software library for cryptographic mechanisms. IFIP/Sec '96, Chapman & Hall, London 1996, 489-491

Cryptix http://www.cryptix.org/

HäKK_92 H. Härtig, O. Kowalski, W. Kühnhauser: The BirliX Security Architecture. In: Journal of Computer Security, 2(1). 5-21, 1993

Kran_96 A. Krannig: PLASMA - Platform for Secure Multimedia Applications. In: Proc. Communications and Multimedia Security II, Essen, 1996

MPSC_93 S. Muftic, A. Patel, P. Sanders, R. Colon u.a.: Security Architecture for Open Distributed Systems. Wiley, Chichester 1993

OMG_97 OMG: Security Service Specification. In: CORBAservices: Common Object Services Specification. Chapter 15, November 1997

Oppe_94 R. Oppermann: Individualisierung von Benutzungsschnittstellen. In: Edmund Eberleh, Horst Oberquelle, Reinhard Oppermann (Hrsg.): Einführung in die Software-Ergonomie. 2. Auflage, Walter de Gruyter, 1994

PiRi_94 J-M. Piveteau, H. P. Rieß: Eine generische Sicherheitsarchitektur für Telekommunikationsnetze. In: Proc. der Fachtagung Sicherheit in Informationssystemen, Schweiz, 1994

PSWW_98 A. Pfitzmann, A. Schill, A. Westfeld, G. Wicke, G. Wolf, J. Zöllner: A Java-based distributed platform for multilateral security. Proc. of TREC '98, LNCS 1402, Springer-Verlag, pp. 52-64

PSWW1_99 A. Pfitzmann, A. Schill, A. Westfeld, G. Wicke, G. Wolf, J. Zöllner: Flexible mehrseitige Sicherheit für verteilte Anwendungen. In: R. Steinmetz (Hrsg.): Kommunikation in Verteilten Systemen, 11. ITG/GI-Fachtagung, Darmstadt, 1999, Springer-Verlag, 130-143

PSWW2_99 A. Pfitzmann, A. Schill, A. Westfeld, G. Wicke, G. Wolf, J. Zöllner: Systemunterstützung für mehrseitige Sicherheit in offenen Datennetzen. Erscheint in: Informatik, Forschung und Entwicklung, 14/2 1999, Springer-Verlag, Berlin

Schi_97 A. Schill: DCE - Das OSF Distributed Computing Environment: Grundlagen und Anwendung, 2., erweiterte Auflage, Springer-Verlag, 1997

SSONET http://www.mephisto.inf.tu-dresden.de/RESEARCH/ssonet/ssonet.html

Waid_96 M. Waidner: Development of a Secure Electronic Marketplace for Europe. In: E. Bertino, H. Kurth, G. Martella, E. Monolivo: „Computer Security - ESORICS 96“, Springer-Verlag, Berlin 1996, 1-14

Wiew_96 E. Wiewall: Secure Your Applications with the Microsoft CryptoAPI. In: Microsoft Developer Network News, 3/4 1996, Microsoft Press

Wolf_98 G. Wolf: Generische, attributierte Aktionsklassen für mehrseitig sichere, verteilte Anwendungen. In: Proc.Workshop Sicherheit und Electronic Commerce, Oktober ’98, Essen, Vieweg-Verlag

WPSW_97 G. Wolf, A. Pfitzmann, A. Schill, A. Westfeld, G. Wicke, J. Zöllner: Sicherheitsarchitekturen: Überblick und Optionen, In: Günter Müller, Andreas Pfitzmann: Mehrseitige Sicherheit in der Kommunikationstechnik – Verfahren, Komponenten, Integration. Addison Wesley Longman Verlag, 1997

WWWZ1_98 A. Westfeld, G. Wicke, G. Wolf, J. Zöllner: Generalisierung und Implementierung der Sicherheitsarchitektur. Zwischenbericht zum 30.04.1998. TU Dresden, April 1998.