Georg Borges
Jörg Schwenk
Herausgeber

Daten- und Identitätsschutz in Cloud Computing, E-Government und E-Commerce

Springer

Daten- und Identitätsschutz in Cloud Computing, E-Government und E-Commerce

Georg Borges • Jörg Schwenk
(Hrsg.)

Daten- und Identitätsschutz in Cloud Computing, E-Government und E-Commerce

Springer

Herausgeber
Georg Borges
Lehrstuhl für Bürgerliches Recht
Ruhr-Universität Bochum
Bochum
Deutschland

Jörg Schwenk
Lehrstuhl für Netz- und Datensicherheit
Ruhr-Universität Bochum
Bochum
Deutschland

ISBN 978-3-642-30101-8 ISBN 978-3-642-30102-5 (eBook)
DOI 10.1007/978-3-642-30102-5
Springer Heidelberg Dordrecht London New York

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Gedruckt auf säurefreiem Papier

Springer ist Teil der Fachverlagsgruppe Springer Science+Business Media (www.springer.com)

Vorwort

Der Schutz von Identitäten als technische und rechtliche Herausforderung hat mit der umfassenden Nutzung des Internet in nahezu allen Lebensbereichen eine neue Qualität erreicht.

Auch wenn der Missbrauch von Identitäten eine seit alters bekannte Form kriminellen oder jedenfalls unbefugten Verhaltens darstellt, hat sich mit der Nutzung des Internet eine neue Dimension der Bedrohung ergeben, die sich auch in der Bildung des neuen Begriffs des Identity Theft (Identitätsdiebstahl) widerspiegelt. Dies dürfte im Kern darauf beruhen, dass die Identität einer Person in der elektronischen Kommunikation oft auf einen Datensatz mit wenigen Elementen, etwa Name (Nutzername) und Kontonummer, beschränkt wird und entsprechend leicht missbraucht werden kann. Folglich werden zur Feststellung oder Überprüfung der Identität Authentisierungsverfahren eingesetzt, die aber, wie die Entwicklungen der letzten Jahre mit beeindruckender Deutlichkeit gezeigt haben, ihrerseits Angriffen ausgesetzt sind.

Ebenso erhält der Datenschutz in der Kommunikation per Internet eine neue Qualität, da die Speicherung und automatisierte Verarbeitung personenbezogener Daten in der elektronischen Kommunikation notwendig in weitaus höherem Maße anfallen als bei traditionellen Kommunikationsformen.

Neue technische und organisatorische Trends, wie Cloud Computing, und die Erschließung neuer Anwendungsfelder für die Kommunikation per Internet, wie etwa staatliche Verwaltung in Form des sogenannten E-Government, führen nicht zuletzt im Identitäts- und Datenschutz zu besonderen Herausforderungen. Diese können nicht disziplinintern bewältigt werden, da insbesondere technische und rechtliche Aspekte, aber auch ökonomische und soziologische Fragen ineinander greifen. Die notwendige Verknüpfung technischer, rechtlicher und ökonomischer Forschung ist bisher nicht ausreichend gelungen. Dies dürfte ein Grund dafür sein, dass bei der Einführung neuer Technologien, wie Cloud Computing, oder der Nutzung des Internet, etwa im E-Government, die bestehende oder jedenfalls empfundene Rechtsunsicherheit als wichtiges Hemmnis gesehen wird.

Das vorliegende Werk untersucht aktuelle Herausforderungen an Daten- und Identitätsschutz im Internet aus interdisziplinärer Perspektive. Die Beiträge beruhen auf dem Symposium „Identitäts- und Datenschutz zwischen Sicherheitsanforderungen und Sicherheitslücken“, das die Arbeitsgruppe Identitätsschutz im Internet (a-i3)

und das Bundesamt für Sicherheit in der Informationstechnik (BSI) im April 2011 in Bochum veranstaltet haben (Informationen dazu unter www.a-i3.org), sind aber vom Symposium unabhängig.

Im ersten Teil des Werkes werden zentrale technische und rechtliche Aspekte des Cloud Computing erörtert. Dabei liegt der Fokus auf dem Datenschutz, da in diesem Bereich derzeit die zentralen rechtlichen Probleme des Cloud Computing gesehen werden.

Im zweiten Teil werden technische und soziologische Grundlagen der Sicherheit der Kommunikation im Internet erörtert. Neben neuen Angriffen im E-Commerce wird die Bedeutung des Nutzers und dessen Verhaltens für die Sicherheit der Kommunikation per Internet im dritten Teil interdisziplinär untersucht.

Der dritte Teil behandelt, wiederum interdisziplinär, Sicherheit im E-Government. Hier werden Aspekte der Sicherheit der Identifizierung bei chipkartenbasierten Verfahren, denen im E-Government besondere Bedeutung beigemessen wird, untersucht und Grundlagen der rechtlichen Anforderungen an die Sicherheit im E-Government erörtert.

Georg Borges

Inhalt

Autorenverzeichnis

Matthias Armgardt Lehrstuhl für Bürgerliches Recht, Antike Rechtsgeschichte, Römisches Recht und Neuere Privatrechtsgeschichte, Universität Konstanz, Universitätsstraße 10, 78457 Konstanz, Deutschland
E-Mail: matthias.armgardt@uni-konstanz.de

Georg Borges Lehrstuhl für Bürgerliches Recht, deutsches und internationales Wirtschaftsrecht, insb. IT-Recht, Ruhr-Universität Bochum, Universitätsstraße 150, 44801 Bochum, Deutschland
E-Mail: georg.borges@rub.de

Kirstin Brennscheidt Lehrstuhl für Bürgerliches Recht, deutsches und internationales Wirtschaftsrecht, insb. IT-Recht, Ruhr-Universität Bochum, Universitätsstraße 150, 44801 Bochum, Deutschland
E-Mail: kirstin.brennscheidt@rub.de

Jens Eckhardt JUCONOMY Rechtsanwälte, Graf-Recke-Straße 82, 40627 Düsseldorf, Deutschland
E-Mail: eckhardt@juconomy.de

Marit Hansen Stv. Landesbeauftragte für Datenschutz Schleswig-Holstein, Unabhängiges Landeszentrum für Datenschutz Schleswig-Holstein, Holstenstraße 98, 24103 Kiel, Deutschland
E-Mail: marit.hansen@datenschutzzentrum.de

Moritz Horsch ecsec GmbH, Sudetenstraße 16, 96247 Michelau, Deutschland
E-Mail: moritz.horsch@ecsec.de

Stephan G. Humer Arbeitsbereich Internetsoziologie, Universität der Künste Berlin, Grunewaldstraße 2–5, 10823 Berlin, Deutschland
E-Mail: humer@udk-berlin.de

Detlef Hühnlein ecsec GmbH, Sudetenstraße 16, 96247 Michelau, Deutschland
E-Mail: detlef.huehnlein@ecsec.de

Florian Kohlar Lehrstuhl für Netz- und Datensicherheit, Ruhr Universität Bochum, Universitätsstraße 150, 44780 Bochum, Deutschland
E-Mail: florian.kohlar@rub.de

Johannes Schmölz ecsec GmbH, Sudetenstraße 16, 96247 Michelau, Deutschland
E-Mail: johannes.schmoelz@ecsec.de

Jochen Schneider SSW Schneider Schiffer Weihermüller, Beethovenstraße 6, 80336 München, Deutschland
E-Mail: jochen.schneider@ssw-muc.de

Jörg Schwenk Lehrstuhl für Netz- und Datensicherheit, Ruhr-Universität Bochum, Universitätsstraße 150, 44801 Bochum, Deutschland
E-Mail: joerg.schwenk@rub.de

Ralf Müller-Terpitz Lehrstuhl für Staats- und Verwaltungsrecht sowie Wirtschaftsverwaltungs-, Medien- und Informationsrecht Universität Passau, Innstraße 40, 94030 Passau, Deutschland
E-Mail: ralf.mueller-terpitz@uni-passau.de

Tobias Wich ecsec GmbH, Sudetenstraße 16, 96247 Michelau, Deutschland
E-Mail: tobias.wich@ecsec.de

Teil I
Sicherheit und Datenschutz beim Cloud Computing

Kapitel 1
Angriffe gegen Cloud Computing

Jörg Schwenk

1.1 Einleitung

Der Begriff „Cloud Computing“ wurde geprägt als eine zwar nicht scharfe, aber dennoch technisch fundierte Beschreibung verschiedener neuer Outsourcing-Technologien. Mittlerweile ist daraus ein reiner Marketingbegriff geworden, der auf alle möglichen Arten von Webanwendungen, insbesondere auf Web Storage, angewandt wird. Es ist daher zunächst eine Begriffsklärung erforderlich.

Cloud Computing wurde vom amerikanischen NIST definiert als ein Modell, um einfach und on demand über ein Netzwerk Zugriff auf einen Pool von konfigurierbaren Computing-Ressourcen (z. B. Netzwerke, (virtuelle) Server, Speicherplatz, Anwendungen und Dienste) zu gewähren. Diese Ressourcen sollen schnell bereitgestellt und freigegeben werden, mit minimalem Managementaufwand und minimaler Interaktion mit dem Cloud Provider.[1]

Diese etwas schwammige Definition dient zunächst einmal zur Abgrenzung gegenüber einem älteren Begriff: Outsourcing ist im Gegensatz zu Cloud Computing ein langwieriger Prozess, der mit hohem Managementaufwand und in intensiver Interaktion mit dem Outsourcing-Provider durchgeführt wird.

Für Cloud Computing werden traditionell die drei Bereiche Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) und Software-as-a-Service (SaaS) unterschieden. Typische IaaS-Angebote sind Elastic Cloud Computing von Amazon Web Services (AWS EC2, Public Cloud)[2] oder Eucalyptus-basierte Angebote (Private Cloud).[3] Microsoft hat mit seiner AZURE-Plattform[4] ein wichtiges PaaS-Angebot

[1] Grance und Mell 2010.

[2] http://aws.amazon.com/de/.

[3] Eucalyptus, http://open.eucalyptus.com/.

[4] http://www.microsoft.com/germany/business/cloudservices/.

J. Schwenk (✉)
Lehrstuhl für Netz- und Datensicherheit, Ruhr-Universität Bochum, Universitätsstraße 150, 44801 Bochum, Deutschland
E-Mail: joerg.schwenk@rub.de

G. Borges, J. Schwenk (Hrsg.), *Daten- und Identitätsschutz in Cloud Computing, E-Government und E-Commerce,*
DOI 10.1007/978-3-642-30102-5_1, © Springer-Verlag Berlin Heidelberg 2012

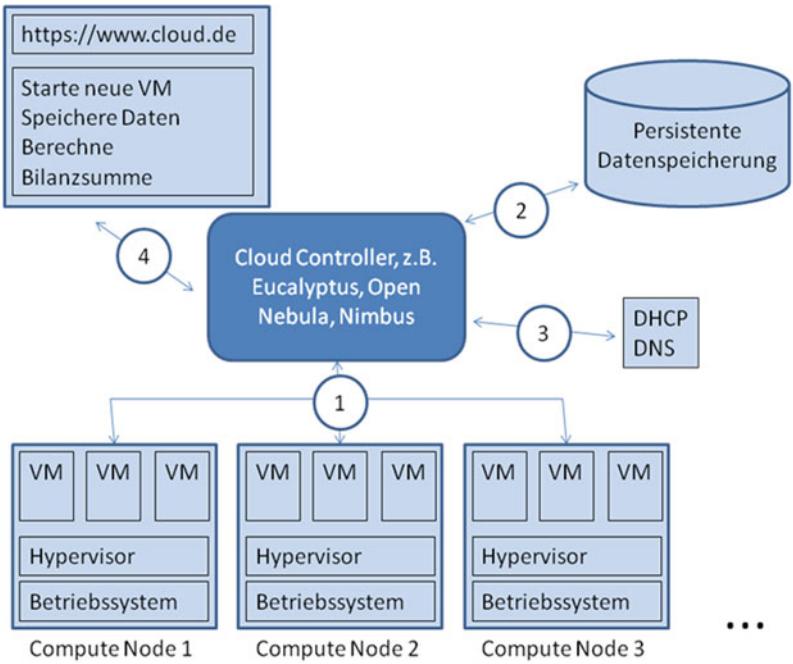

Abb. 1.1 Komponenten einer IaaS Cloud Computing-Infrastruktur. Auf die nummerierten Schnittstellen wird im Text Bezug genommen

geschaffen. Bei SaaS verschwimmen die Grenzen hin zu fortgeschrittenen Web 2.0-Anwendungen: Google Docs oder Salesforce könnte man in beide Kategorien einordnen.

Da Cloud Computing in seiner IaaS-Ausprägung am klarsten zu definieren ist, wollen wir im Folgenden einen Blick auf die diversen Komponenten einer solchen Anwendung werfen.

1.2 Komponenten einer Cloud-Umgebung

Abbildung 1.1 stellt die verschiedenen Komponenten einer IaaS-Anwendung dar. Wir lehnen uns in dieser Abbildung an Sempolinski und Thain[5] an; in diesem Artikel ist die Funktionsweise dreier wichtiger Open Source-Plattformen im Bereich IaaS (Eucalyptus, Nimbus, OpenNebula) beschrieben.

[5] Sempolinski und Thain 2010.

1.2.1 Cloud Controller

Im Zentrum eines IaaS-Systems steht der Cloud Controller, der die verschiedenen Komponenten automatisch konfiguriert und zusammenbindet.

1.2.2 Die Hypervisor-Schnittstelle (1)

Eine der Basistechnologien, ohne die Cloud Computing nicht denkbar wäre, ist die Virtualisierung von Hard- und Software. Auf den physikalisch vorhandenen Rechnern läuft ein Host-Betriebssystem und darin ein Hypervisor, der die Virtualisierung steuert. Über den Hypervisor können virtuelle Maschinen (VMs), deren Images vorher definiert wurden, on demand gestartet und gestoppt werden.

Diese Kerntechnologie von Cloud Computing wurde bereits intensiv auf Sicherheitslücken untersucht:

- Ristenpart et al. haben versucht, die Grenzen zwischen verschiedenen VMs zu überwinden. Sie konnten zeigen, dass es unter bestimmten Bedingungen für einen Angreifer möglich ist, eine eigene VM auf dem gleichen physikalischen Server zu starten, auf dem die VM des Opfers läuft. Anschließend können Seitenkanalinformationen der Opfer-VM gelesen werden.[6]
- Bugiel et al. konnten zeigen, dass viele in der Amazon Cloud abgelegte Serverimages fehlerhaft konfiguriert waren: Sie enthielten Passwörter und kryptographische Schlüssel. Dieser Fehler ist allerdings nicht dem Cloud-Anbieter, sondern allein den Cloud-Nutzern anzulasten.[7]

1.2.3 Persistente Datenspeicherung (2)

Da beim Stoppen einer virtuellen Maschine auch alle Daten gelöscht werden, ist es erforderlich, einen persistenten Speicherort für Daten bereitzustellen. Dieser persistente Datenspeicher muss automatisch in die VM eingebunden werden, wenn dies im Image konfiguriert ist.

1.2.4 Setup virtueller Netzwerke (3)

VMs müssen, wie reale Computer auch, miteinander kommunizieren können. Hierzu muss eine Netzwerkanbindung vorhanden sein.

---

[6] Ristenpart et al. 2009, S. 199.

[7] Bugiel et al. 2011, S. 389.

Da VMs ständig erzeugt und gelöscht werden, ist die Neukonfiguration des Netzwerks (Zuweisung von MAC-Adressen, Zuweisung von IP-Adressen, Konfiguration von Switches und Routern) ein hochdynamischer Vorgang, der automatisiert abgewickelt werden muss.

1.2.5 Das Kontrollinterface (4)

Die Kontrolle der Daten und Berechnungen in einem Cloud-System erfolgt über eine oder mehrere Kontrollschnittstellen. Fast immer wird eine Web 2.0-Schnittstelle angeboten, die über einen normalen Webbrowser bedient werden kann. Seltener findet man dedizierte Clientsoftware, die eine Automatisierung der Steuerung erlaubt und die mit SOAP- oder REST-Technologien arbeitet.

Das Kontrollinterface ist der Hauptangriffspunkt eines Cloud-Systems, da es zur Außenwelt hin offen ist. Wir werden auf diese Problematik in Abschn. 1.4 näher eingehen.

1.3 Implementierungen des Kontrollinterface

Bei unseren Untersuchungen haben wir drei Ausprägungen des Kontrollinterface gefunden:

- Fast alle Cloud-Dienste bieten eine browserbasierte Schnittstelle an. In dieser Ausprägung kommen nur gängige Web 2.0-Technologien zum Einsatz, zentrale Sicherheitskomponenten sind die Same Origin Policy, SSL/TLS, Passwörter und Session Cookies.
- Zur Automatisierung der Steuerung der Cloud-Instanzen können REST-basierte Techniken angewandt werden. Im Wesentlichen wird hierbei das HTTP-Protokoll mit neuen Headerzeilen verwendet, die auch Sicherheitsfunktionalität enthalten. Zur Erzeugung der HTTP-Request und zur Verarbeitung der HTTP-Responses ist kein Browser erforderlich: Dies grenzt REST-basierte Techniken von rein browserbasierten Schnittstellen ab.
- Das XML-basierte SOAP-Protokoll bietet bessere Strukturierungsmöglichkeiten als eine REST-basierte Schnittstelle und kann daher auch zur Automatisierung eingesetzt werden. Allerdings verfügen Web-Entwickler im Allgemeinen über gute HTTP-, aber nur geringe SOAP-Kenntnisse, weshalb hier keine klare Präferenz zu erkennen ist. Der große Vorteil von SOAP besteht in den durch die WS-Security-Standards festgelegten Sicherheitsmechanismen, die in vielen Softwarebibliotheken unterstützt werden.

Im Folgenden soll das Kontrollinterface, das unser Hauptforschungsgebiet darstellt, in diesen drei Ausprägungen näher dargestellt werden.

1.3.1 Browsertechnologien

Im Browser gibt es drei zentrale Sicherheitsparadigmen: Sandboxing, Informationsflusskontrolle (mittels der Same Origin Policy) und SSL/TLS.

1.3.1.1 Sandboxing

Durch verschiedene Sandboxing-Mechanismen für Java-Applets und geladenen Javascript-Code soll verhindert werden, dass über aus dem Internet geladene Codefragmente persönliche, auf dem PC des Internetnutzers gespeicherte Daten gelesen werden können.

Dieses Paradigma verliert zunehmend an Bedeutung, weil die wichtigen persönlichen Daten vermehrt im Browser gespeichert werden:

- Passwörter werden im Passwortmanager des Browsers gespeichert.
- Authentifizierungsdaten wie HTTP-Cookies oder SAML-Assertions werden im Document Object Model des Browsers dauerhaft oder vorübergehend gespeichert und können dort über browserinterne Angriffe ausgelesen werden.

1.3.1.2 Same Origin Policy

Die Same Origin Policy (SOP) ist von ihrem Ansatz her ein äußerst sinnvoller Versuch, eine Informationsflusskontrolle im Browser zu erzwingen. Die Grundidee war hierbei, nur solche aktiven (Javascript) und passiven Inhalte miteinander interagieren zu lassen, die von dem gleichen Webserver geladen wurden. Die SOP legt dabei folgende Parameter aus der URL zugrunde: das Protokoll (z. B. http oder ftp), den Domainnamen und die Portnummer (die entweder explizit angegeben sein kann oder sich aus dem Protokoll ergibt).

Abbildung 1.2 stellt an einem fiktiven Beispiel dar, wie die SOP funktioniert: Im Browser sind gleichzeitig eine Online-Banking-Seite und die Seite des Angreifers geladen. Der Angreifer hat in seine Seite ein Script eingebettet, das versucht, die Kontonummer für die aktuelle Transaktion zu verändern. Dies wird von der SOP unterbunden, weil zwar Protokoll (https) und Portnummer (443) gleich sind, die Domainnamen sich aber unterscheiden.

Dieses Konzept wurde aber sukzessive aufgeweicht:

- Für Inhalte von Servern, die zur gleichen Domain gehören, kann mittels Javascript eine Superdomain als Origin definiert werden (ftp.example.org, images.example.org und www.example.org können demnach dem gemeinsamen Origin example.org zugeordnet werden).[8]
- Für verschiedene im DOM des Browsers gespeicherte Objekte wurden unterschiedliche Cross-Domain-Kommunikationspolicies entwickelt. So ist zum

[8] http://code.google.com/p/browsersec/wiki/Main.

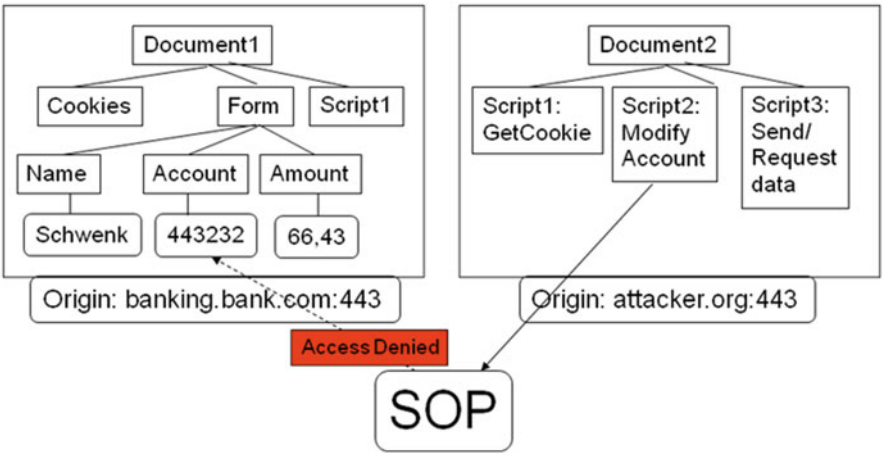

Abb. 1.2 Funktionsweise der Same Origin Policy

Beispiel die Cross-Domain-Übertragung von HTTP-Cookies verboten; die Cross-Domain-Übertragung von Daten, die in einem HTML-Formular gespeichert sind, ist aber erlaubt.

- Da in der Regel umfangreiche CSS- und Javascript-Bibliotheken in moderne Webseiten eingebunden werden, ist es vorteilhaft, diese nicht auf dem eigenen Webserver zu hosten, sondern direkt von anderen Servern (z. B. aus einem Content Delivery Network) zu laden. Die Domains dieser fremden Server werden dann in die Origin des Dokuments automatisch eingebunden, da die Hyperlinks auf diese Server in dem zuerst geladenen HTML-Dokument enthalten sind.
- Neue Entwicklungen im Bereich Web 2.0 machen eine strikte Anwendung der SOP unmöglich. So werden z. B. Mashups zunehmend beliebter; hierbei handelt es sich um Webseiten, die Inhalte von verschiedenen Webseiten kombinieren (z. B. die Einbettung von Google Maps in die Webanwendung eines Immobilienmaklers).

1.3.1.3 SSL/TLS

Das Transport Layer Security-Protokoll (der Name „Secure Socket Layer“ stammt von der Firma Netscape und wurde mit der Standardisierung durch die IETF mit Version 3.1 aufgegeben) kann dazu eingesetzt werden, die über eine TCP-Verbindung übertragenen Bytes zu verschlüsseln und ihre Integrität zu sichern.

Die Intelligenz des TLS-Protokolls steckt im TLS-Handshake, der in Abb. 1.3 dargestellt ist. Ein einfacher Drei-Wege-Schlüsselaustausch, meist auf Basis des RSA-Algorithmus (in der Abbildung dargestellt durch die Briefkästen), wird

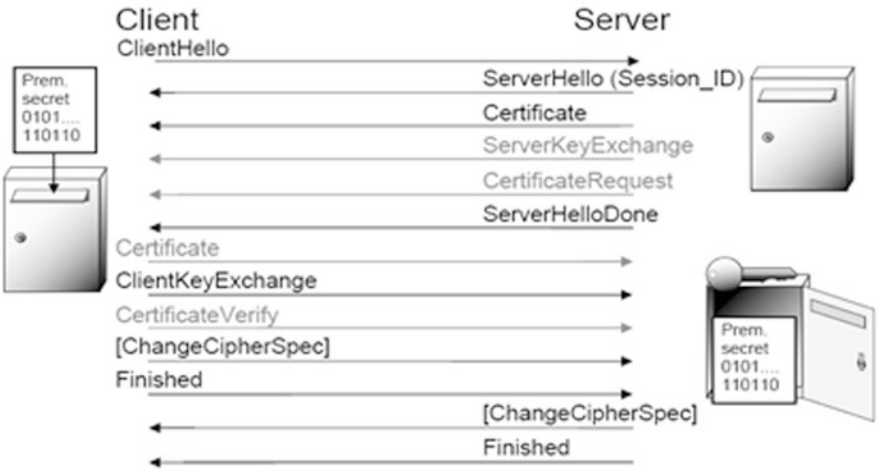

Abb. 1.3 Das TLS-Handshake-Protokoll

umrahmt von Protokollnachrichten, die ein automatisches Aushandeln der kryptographischen Algorithmen ermöglichen und das Handshake-Protokoll gegen alle bekannten Angriffe absichern.

In seiner am häufigsten eingesetzten Form bleibt der Browser für den Server vollständig anonym. In diesem Fall werden die grau dargestellten Nachrichten nicht benötigt.

1.3.1.4 Angriffe auf browserbasierte Schnittstellen

Die komplexe Interaktion zwischen dem Browser und den einzelnen Serverkomponenten kann durch eine Reihe spezialisierter Angriffe ausgenutzt werden:

- iFrame Injection[9]: Bei diesem Angriff wird ein fremder Inhalt in eine Webseite eingeschleust. Dies kann nicht-persistent erfolgen (z. B. durch Übergabe des HTML-Quelltextes als Suchparameter, der dann zusammen mit der Fehlermeldung vom Server an den Browser zurückgesandt und dort nicht als String, sondern als interpretiertes HTML dargestellt wird) oder persistent (indem der Angreifer sich Zugang zum Webserver verschafft und ein unsichtbares iFrame dort speichert). In einem solcherart injizierten iFrame sind oft Hyperlinks eingebunden, die Schadcode auf den Browser von der Seite des Angreifers nachladen.
- Cross Site Scripting (XSS)[10]: Analog zu iFrame Injection wird ein Javascript-Programm unberechtigt in eine fremde Webseite, die im Browser ausgeführt

[9] http://eisabainyo.net/weblog/2009/04/06/iframe-injection-attack/.

[10] http://www.owasp.org/index.php/Cross-site_Scripting_(XSS).

wird, eingeschleust. Dadurch erlangt das Programm die Berechtigung, auf alle im Browser vom fremden Webserver geladenen Inhalte lesend und schreibend zuzugreifen. Dies kann unter anderem dazu benutzt werden, Identitätsdaten zu stehlen.

- SQL Injection[11] ist ein Angriff auf Serverseite: Hier wird ausgenutzt, dass fast jede Webanwendung eine SQL-Datenbank einbindet. Durch geeignete Formatierung einer Eingabe in ein HTML-Formularfeld kann der Angreifer bei dieser Form des Angriffs sicherstellen, dass seine Eingabe von den Frontend-Servern als SQL-Befehl an die Datenbank weitergeleitet wird und dort den gewünschten Effekt erzielt.
- Cross Site Request Forgery[12]: Bei dieser Angriffsart wird ausgenutzt, dass die Authentifizierung eines Nutzers gegenüber einer Webanwendung oft sitzungsbasiert ist. Außerdem kann die Webanwendung oft nicht unterscheiden, ob eine HTTP-Anfrage unter Mithilfe des Nutzers erzeugt wurde, oder ob sie automatisch generiert oder vorberechnet ist. Im einfachsten Fall muss der Angreifer auf seiner eigenen Webseite nur einen Image-Link einbetten, der als src-Parameter einen Query-String für eine Webanwendung enthält, für die sich das Opfer bereits authentifiziert hat. Der Server der Webanwendung sieht nur die HTTP-Anfrage und beantwortet diese, da die sitzungsbasierten Identitätsdaten vom Browser mitgeschickt wurden. Durch den Einsatz von Javascript kann man diese Art von Angriffen ausweiten, bis hin zu vollautomatischen Angriffen auf Online-Banking[13].

1.3.2 REST-basierte Dienste

Die Abkürzung REST steht für Representational State Transfer.[14] Sie beschreibt eine Alternative zu SOAP und XML-RPC als Basis für Webservices.

REST-basierte Webanwendungen sind schon seit einigen Jahren erfolgreich im Einsatz. Sie basieren auf HTTP als Kommunikationsprotokoll, mit seinen beiden einfachen Methoden GET und POST. Auf Client-Seite werden HTML und Javascript eingesetzt, auf Serverseite Skriptsprachen (PHP), Java oder andere Entwicklungsumgebungen (.NET).

Das größte Problem bei diesem Paradigma besteht in der unstrukturierten Art der Datenübertragung vom Client zum Server: Sowohl GET als auch POST erlauben nur eine einfache, schwach strukturierte Übertragung von Daten in der Form „Name=Wert“, wobei „Name“ und „Wert“ jeweils nur Strings sein dürfen.

---

[11] http://en.wikipedia.org/wiki/SQL_injection.

[12] http://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF).

[13] https://freedom-to-tinker.com/blog/wzeller/popular-websites-vulnerable-cross-site-request-forgery-attacks/.

[14] http://en.wikipedia.org/wiki/Representational_State_Transfer.