Web-Security

www.sit.fraunhofer.de

Fraunhofer SIT

Entwickler | Nutzer | Administrator | Angreifer | Kontrollpunkt | Bereich

Fraunhofer-Institut für Sichere Informationstechnologie

SIT Darmstadt
Rheinstraße 75
64295 Darmstadt
Tel.: +49 6151 869-399
Fax: +49 6151 869-224

SIT St. Augustin
Schloss Birlinghoven
53754 Sankt Augustin
Tel.: +49 2241 14-3272
Fax: +49 2241 14-3007

SIT Berlin
Anna-Louisa-Karsch-Str. 2
10178 Berlin

www.sit.fraunhofer.de

Gegenmaßnahmen

Sichere Ablage von Benutzereingaben
Kritische Daten nur serverseitig verwalten, da alle Informationen auf Nutzerseite manipulierbar sind. Alle Parameter generell auf gültigen Wertebereich gegen eine Whitelist prüfen. Zum Beispiel mit regulären Ausdrücken wie (a-zA-Z0-9){1,30}, um die Erwartung von Parametern der Länge 1 bis 30 ohne Sonderzeichen zu prüfen.

Schutz vor bösartigem Inhalt
Alle aktiven Inhalte nur kontrolliert aktivieren. Mit JavaScript das Laden von Webanwendungsseiten innerhalb von iFrames verhindern und X-FRAME-OPTIONS-Header nutzen. Firefox-Nutzer können das kostenlose Plugin NoScript verwenden, das mit einem Schutz gegen Clickjacking wirbt.

Aufmerksame Nutzung / Nutzerunterstützung
- Benutzerschulung über die Gefahren und den sicheren Umgang mit Browsern.
- Authentisierung der Anwendung gegenüber dem Benutzer durch Benutzer-individuelles Wissen.

Authentizität der Interaktion sicherstellen
Aktionen in Web-Anwendungen dürfen nur bei Übermittlung eines Parameters starten, der dem Angreifer nicht bekannt sein kann und nicht mit wirtschaftlichem Aufwand durchprobierbar ist. Alle Funktionsaufrufe von Web-Anwendungen müssen so geschützt werden, um eine unautorisierte Fernsteuerung durch Angreifer zu verhindern.

Browsersicherheit
- Betriebssystem-, Browser- und Virenscanner-Updates regelmäßig einspielen.
- Browser-Techniken wie **DNS-Pinning** erschwere **DNS-Rebinding**-Angriffe. Das Umgehen dieses Schutzes wird als **Anti-DNS-Pinning** bezeichnet.
- Isolieren des Browsers für Umgebung mit hohem Schutzbedarf (z. B. Virtualisierung)

Kommunikationssicherheit
- HTTPS mit sicheren Wurzel-Zertifikaten für die gesamte Dienstnutzungszeit verwenden und SSL/TLS-Konfiguration am Server gegen BEAST Angriff absichern
- Nutzer über mögliche Sicherheitswarnungen aufklären und Verhalten für diesen Fall festlegen.

DNS absichern
- Browser mit DNS-Pinning verwenden
- In Intranet-Anwendungen den Host-Header prüfen
- Intranet durch Filtern der DNS-Antworten schützen (lokale IP-Adressen in Antworten von entfernten DNS-Servern zurückweisen)
- DNS-Infrastruktur und Clients gegen Manipulation durch Malware sichern und DNSSEC konfigurieren
- HTTPS mit sicheren Wurzel-Zertifikaten hilft, gefälschte DNS-Antworten für externe Server aufzudecken, wenn der Benutzer die Warnmeldungen seines Browsers richtig interpretieren kann.

Schutz der Sitzungen
- Schutz auf Netzwerkebene siehe Kommunikationssicherheit
- Sitzungsverwaltung auf Anwendungsebene muss das Raten der Sitzungsidentifikation verhindern.
- Sitzungsidentifikatoren immer nur serverseitig generieren. Sitzung mit dem serverseitig vergebenen Sitzungsidentifikator erst nach erfolgter Anmeldung des Nutzers verknüpfen.

Gefälschte Kommunikationspartner ausschließen
Komponenten müssen sich gegenseitig mithilfe kryptografischer Verfahren authentisieren. Zusätzlich Netzwerk gegen Missbrauch von Identitäten schützen.

Sicherer Umgang mit Passwörtern
- Bei der Installation Standardpasswörter immer in der Dokumentation recherchieren und ändern.
- Passwörter immer vom Benutzer während der Installation wählen lassen und keine fest vordefinierten vorgeben.

Dedizierte Sprachkenntnisse
Die Entwickler müssen die Fallstricke der Programmiersprachen kennen, um sichere Programme zu schreiben. Zudem können Tools helfen, kritischen Code zu identifizieren oder bei der Übersetzung abzusichern.

Sicherer Umgang mit Daten
Alle Daten vor der Verwendung in Web-Seiten konvertieren (siehe ***Metazeichen***). Auch manipulierte Multimedia-Daten und Dokumente sind potenziell Träger von aktivem Code (z. B. JavaScript, CSS, Applet, Flash, HTML, etc.). Daher vor der Verwendung in Web-Seiten konvertieren, um enthaltenen Code zu entfernen. Zudem Dateiablage unter anderer Domain (siehe Dateikombination **GIFAR**, Code in **EXIF-Metadaten** und ähnliche Angriffe).

Dateioperation absichern
Benutzereingaben in Web-Anwendungen sollten grundsätzlich nicht in Dateioperationen verwendet werden. Falls dies nicht möglich ist, sollte der Zugriff auf eine Liste von Optionen dienen (Case-Abfragen). Bei der Filterung von Pfaden besteht ansonsten immer die Gefahr, plattformspezifische Formatzeichen und unterschiedliche Kodierungsformen unberücksichtigt zu lassen.

Sichere Zeichenverarbeitung
Alle Eingabedaten als manipuliert annehmen und umfeldspezifisch restriktiv konvertieren (**Escaping**), da je nach Programmumfeld Zeichen eine Sonderrolle besitzen. Daten nicht mit String-Operationen zu Befehlen zusammensetzen, sondern bspw. in **„Prepared-Statements"** verwenden oder mit erwarteten Eingaben vergleichen, um zugeordnete Aktion auszulösen.

Sicherheit im Entwicklungsprozess
- Notwendige Rechte bei der Entwicklung bereits reduziert festlegen.
- Nach Abschluss der Entwicklung Reinigungsphase für Anwendung und Dokumentation der angenommenen Web-Server-Konfiguration vorsehen.
- Mit restriktiver Konfiguration der Web-Server-Umgebung starten.
- Zugriffsrechte für Web-Server-Dateisystem prüfen.

Datenbankabsicherung
- Die Daten für Abfragen sind nicht als Zeichenketten in SQL-Befehlen einzufügen, sondern als Parameter an vorbereitete Befehlsfolgen zu übergeben (sog. **Prepared-Statements**)
- Passwörter für Datenbanken nicht in den dynamischen Webseiten speichern.
- Für lesende Zugriffe auf Datenbanken eingeschränkte Nutzer anlegen und konsequent verwenden.

Angriffe

SQL Injection
Ändern von Datenbank-Abfragen durch Einschleusen von Datenbank-Befehlen innerhalb unzureichend geprüfter Eingabedaten. Diese werden innerhalb der Abfrage dann als Teil eines Datenbank-Kommandos und nicht mehr nur als reine Werte behandelt.

Unsichere Konfiguration
Aus der Entwicklungszeit verbliebene Reste unsicherer Konfiguration wie z. B. HTML-Kommentare mit sensiblen Informationen, unsichere Nutzung von Passwörtern in interpretierten Web-Seiten oder eine der Web-Anwendung nicht angepassten Interpreter-Konfiguration können Angriffe auf die Web-Anwendung ermöglichen.

Directory Traversal
Manipulation von Pfadangaben, über die durch Schwächen in der Web-Anwendung auf Dateien und Verzeichnisse des Servers zugegriffen werden kann, die dafür nicht vorgesehen waren. Dadurch kann auf sensible Daten innerhalb des Web-Servers zugegriffen werden.

Format String Angriff
Der Angreifer nutzt eine Schwäche bei der Behandlung von Daten der Web-Anwendung bei formatierten Ausgaben (z.B. mittels printf) aus. Er erhält unberechtigten lesenden und schreibenden Zugriff auf Speicherbereiche durch unzureichend geprüfte Daten mit Formatierungsanweisungen.

Metazeichen
Unsichere Behandlung von Zeichen, die eine Sonderrolle in der verwendeten Programmumgebung besitzen. Innerhalb einer Web-Anwendung können Daten verschiedene Ausführungsumgebungen durchlaufen. Bei fehlender Anpassung können sie dabei das jeweilige Anwendungsverhalten im Sinne des Angreifers verändern. Daraus resultieren beispielsweise auch **SQL-, Command-, XPath-, LDAP-Injection** und ähnliche Angriffe.

Filter Evasion
Umkodieren von Zeichenfolgen mit dem Ziel, die Filterung von Eingabedaten innerhalb der Web-Anwendung zu umgehen. Durch geschicktes Einfügen zusätzlicher Zeichen wie Null-, Tabulator- oder Zeilenwechsel-Zeichen können Schlüsselwörter versteckt werden, wenn die Zeichen auf anderen Ebenen (z.B. im Browser) ignoriert werden. Alternative Kodierungsformen und Zeichensätze können ebenso Filter umgehen.

Buffer Overflow
Provozierter Überlauf eines Pufferbereichs innerhalb von Client, Web-Server oder Web-Anwendung, um dadurch in andere Speicherbereiche der attackierten Komponente ausführbaren Code einzuschleusen. Der Angreifer übernimmt damit die Kontrolle über Browser, Web-Server oder Web-Anwendungen und kann ihr Verhalten zu seinen Gunsten ändern oder Restriktionen umgehen.

Cross Site Scripting – XSS
Die Web-Anwendung fügt Code des Angreifers (z.B. JavaScript) als vermeintlich harmlose Benutzereingabe in eine Antwortseite ein. Der Webbrowser des Opfers führt den Code aus, wodurch die Same-Origin-Policy umgangen wird und der Angreifer Zugriff auf die Web-Anwendung erhält. Der Code kann in der Anwendung gespeichert sein (**Stored-XSS**) oder durch einen Link ausgelöst werden (**Reflected-XSS**).

Standard Accounts
Vergessene oder dem Betreiber unbekannte Standard-Accounts mit hohen Rechten, die nach der Installation ein vorgegebenes Passwort besitzen. Angreifer, die solche Zugänge kennen, missbrauchen diese, um Zugriff auf die Web-Anwendung oder ihre Server-Umgebung zu erlangen.

HTTP Request Smuggling
Ausnutzen der unterschiedlichen Bearbeitung von HTTP-Anfragen in Application Firewall, Cache- und Web-Server. Ziel ist das Umgehen von Filter-Regeln und die Manipulation von Einträgen im Cache. Der Angreifer entgeht damit Restriktionen von Filtern und kann die Zuordnung von Adressen zu Inhalten ändern (**Cache Poisoning**).

DNS Spoofing
Fälschen der Zuordnung von Domainname zur IP-Adresse durch den Angreifer – entweder lokal im Client-Betriebssystem oder in der Hierarchie der zuständigen DNS-Server (**Cache Poisoning**). Eine vermeintlich vertrauenswürdige Adresse der Web-Anwendungen führt dann zum Server des Angreifers.

Session Fixation
Der Angreifer legt die Identifikation der Kommunikationsbeziehung für sein Opfer vor der Anmeldung fest. Meldet sich das Opfer nun an, kann der Angreifer die Sitzung mit der vorgegebenen ID übernehmen.

Sniffing
Daten werden während der Übermittlung von und zur Web-Anwendung mitgelesen aber nicht verändert. Passiver Angriff gegen unverschlüsselte oder schlecht gesicherte Protokolle.

Man-in-the-Middle (MITM)
Eindringen in eine Kommunikation zwischen Web-Anwendung und Web-Server zum Auslesen und Verändern der übertragenen Daten. Ein Angreifer nutzt fehlende oder verwundbare Authentisierung aus, um sich als jeweiliger Endpunkt auszugeben.

Session Hijacking
Angreifer übernimmt eine Kommunikationsbeziehung zwischen Client und Web-Server, nachdem sich beide Kommunikationspartner erfolgreich authentifiziert haben.

Spoofing
Ändern der Netzwerkidentität einer Angreiferkomponente durch Vortäuschen, Manipulieren oder Verschleiern, z.B. um Vertrauenswürdigkeit zu erlangen oder Filter zu umgehen.

DNS Rebinding
Umgehen der Same-Origin-Policy im Browser durch Alternieren von DNS-Antworten. Durch gezieltes Wechseln zwischen IP-Adresse von Angriffs- und Zielrechner erlangt der Angreifer über den Browser eines Opfers indirekten Zugriff auf Anwendungen des Zielrechners, da für den Browser abwechselnd beide IP-Adressen zum selben Host-Namen gehören. Dies hebt die Trennung zwischen Angreifer- und Zielrechner auf.

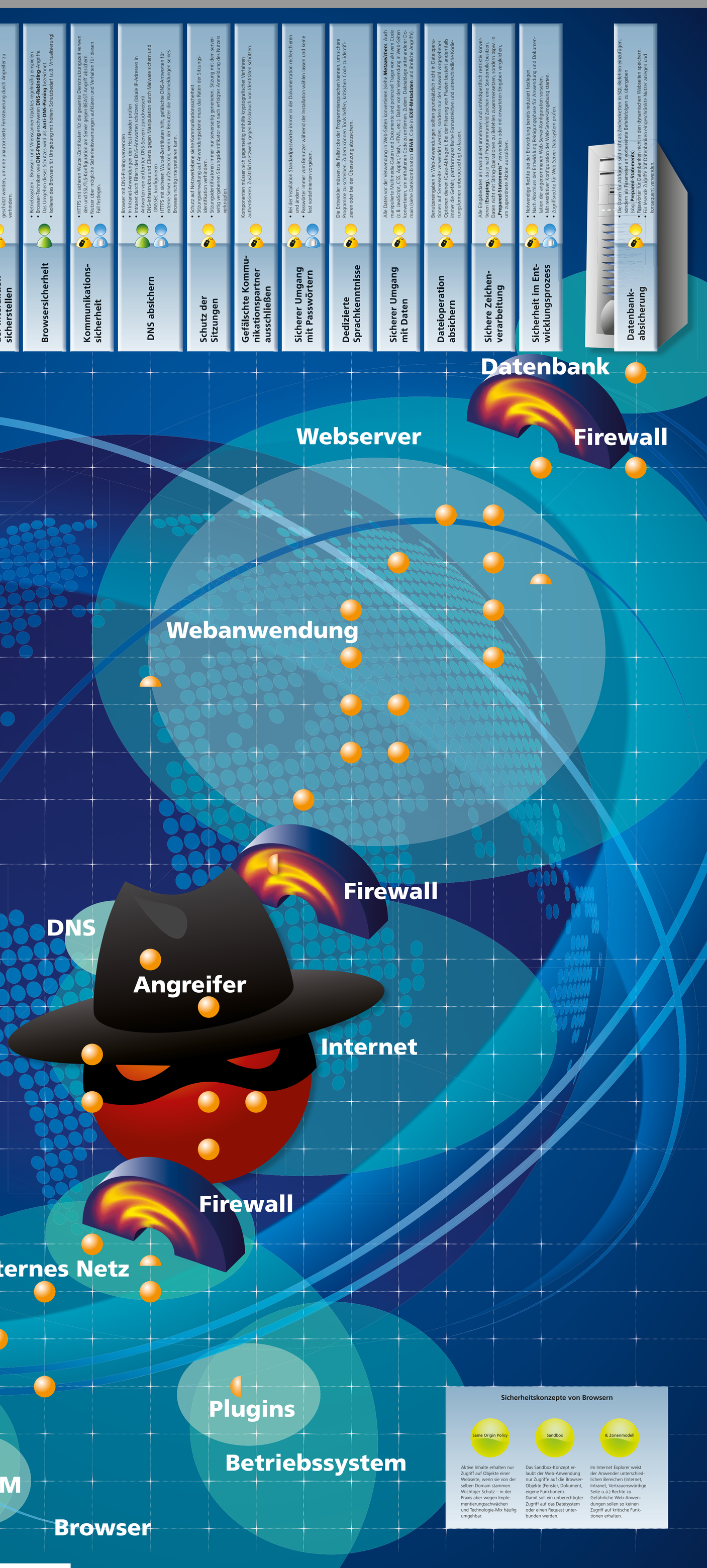
Cross-Site Request Forgery (XSRF) / Session Riding
Auslösen von Aktionen in geöffneten Web-Anwendungen durch die Nutzung manipulierter Web-Seiten. Der Angriff nutzt den Browser des Opfers als Sprungbrett, um dessen Rechte gegenüber der Web-Anwendung unbemerkt einzusetzen.

Malicious Content
Aktiver Angriffscode in Web-Seiten, der Schwächen im Browser des Opfers ausnutzen soll. Bei erfolgreichen Angriffen wird der Browser missbraucht, um weiteren Schadcode auf den Client-PC zu laden.

Phishing
Nachbilden von Seiten und Diensten durch Angreifer, um die Besucher zu verleiten, Passwörter und andere vertrauliche Informationen preiszugeben, im Glauben es handele sich um die vertrauenswürdige Instanz.

UI Redressing / Clickjacking
Täuschen des Benutzers durch manipulierte Web-Seiten, um ihn unbemerkt Aktionen in Web-Anwendungen auslösen zu lassen. Für den Benutzer unsichtbar, führt sein Klick auf die präparierte Schaltfläche zur Aktivierung einer Funktion, die mit seinen Rechten ausgeführt wird.

Hidden Field Tampering
Ändern von Daten, die als versteckte Parameter in ausgelieferten Web-Formularen für die weitere Verwendung in der Web-Anwendung zwischengespeichert wurden. Der Angreifer kann damit der Web-Anwendung gefälschte Daten unterschieben, wenn die Web-Anwendung diese Parameter nicht erneut prüft.



Sicherheitskonzepte von Browsern

Same Origin Policy
Aktive Inhalte erhalten nur Zugriff auf Objekte einer Webseite, wenn sie von der selben Domain stammen. Wichtiger Schutz – in der Praxis aber wegen Implementierungsschwächen und Technologie-Mix häufig umgehbar.

Sandbox
Das Sandbox-Konzept erlaubt der Web-Anwendung nur Zugriffe auf die Browser-Objekte (Fenster, Dokument, eigene Funktionen). Damit soll ein unberechtigter Zugriff auf das Dateisystem oder einen Request unterbunden werden.

IE Zonenmodell
Im Internet Explorer weist der Anwender unterschiedlichen Bereichen (Internet, Intranet, Vertrauenswürdige Seite u.ä.) Rechte zu. Gefährliche Web-Anwendungen sollen so keinen Zugriff auf kritische Funktionen erhalten.

Cloud-Studie zum Download

Die Sicherheit von Cloud-Speicherdiensten ist oft mangelhaft. Das ist das Ergebnis einer Studie des Fraunhofer-Instituts für Sichere Informationstechnologie, das verschiedene Anbieter getestet hat. Fazit: Keiner der getesteten Anbieter konnte die Sicherheitsanforderungen vollständig erfüllen, teilweise fehlte eine ordentliche Verschlüsselung. Neben technischen Mängeln fanden die Tester auch Schwächen in der Benutzerführung.

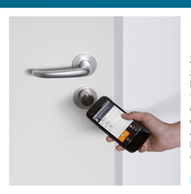
sit4.me/cloudstudy

BizzTrust

Mitarbeiter nutzen Smartphones und Tablet-PCs privat und geschäftlich. Fraunhofer SIT hat mit BizzTrust for Android eine Lösung entwickelt, die die Sicherheit von Unternehmensdiensten und -daten gewährleistet, ohne Funktionalität und Nutzer einzuschränken. BizzTrust trennt Geschäftliches und Privates und kennzeichnet farblich die unterschiedlichen Kontexte. Mit BizzTrust gewinnt das Fraunhofer SIT den „TeleTrusT Award 2012".

www.bizztrust.de

Key2Share



Zutrittsrechte zum Beispiel in Unternehmen zu verteilen kann mühsam sein. Das Fraunhofer SIT hat mit Key2Share eine Lösung für NFC-fähige Android-Smartphones entwickelt. Mitarbeitern von Unternehmen erteilt eine App Zutrittsrechte zu Büros und anderen Räumlichkeiten ihrer Firma. Diese können flexibel und zeitlich begrenzt auch Gästen übertragen werden.

http://sit4.me/key2share