Aktuelle Angriffstechniken auf Web-Applikationen

Andreas Kurtz

cirosec GmbH, Heilbronn

cirosec

Gliederung

- Schwachstellen-Überblick
- Präsentation aktueller Angriffstechniken
  - XPath-Injection
  - Cross-Site Request Forgery
- Neue Sicherheitsrisiken durch „Web 2.0“-Technologien wie AJAX

Gliederung

- Schwachstellen-Überblick
- Präsentation aktueller Angriffstechniken
  - XPath-Injection
  - Cross-Site Request Forgery
- Neue Sicherheitsrisiken durch „Web 2.0“-Technologien wie AJAX

cirosec

Bug-Parade

- Schwachstellensammlungen
  - OWASP: Top 10
  - SANS/MITRE: Top 25 der gefährlichsten Programmierfehler

(siehe iX 03/2009)

cirosec

OWASP Top 10

A1	Cross-Site Scripting
A2	Injection Flaws
A3	Malicious File Execution
A4	Insecure Direct Object Reference
A5	Cross Site Request Forgery (CSRF)
A6	Information Leakage & Improper Error Handling
A7	Broken Authentication & Session Management
A8	Insecure Cryptographic Storage
A9	Insecure Communications
A10	Failure to Restrict URL Access

cirosec

OWASP Top 10

A1	Cross-Site Scripting
A2	**Injection Flaws**
A3	Malicious File Execution
A4	Insecure Direct Object Reference
A5	**Cross Site Request Forgery (CSRF)**

A6	Information Leakage & Improper Error Handling
A7	Broken Authentication & Session Management
A8	Insecure Cryptographic Storage
A9	Insecure Communications
A10	Failure to Restrict URL Access

cirosec

Gliederung

- Schwachstellen-Überblick
- Präsentation aktueller Angriffstechniken
  - XPath-Injection
  - Cross-Site Request Forgery
- Neue Sicherheitsrisiken durch „Web 2.0“-Technologien wie AJAX

cirosec

Was ist XPath?

- XPath ist eine Abfragesprache, um gezielt Knoten aus einem XML-Dokument auszulesen
- W3C-Standard
- Gemeinsamkeiten zwischen SQL und XPath
- Viele Sprachen unterstützen XPath: Java, .NET, PHP, Python, Perl, Ruby, JavaScript, ...

Gefahren bei Verwendung von XPath

- Berechtigungsmodell einer Datenbank erlaubt Rechtevergabe auf Feldebene
  - Eingeschränkte Sichtweise eines Datenbankbenutzers bei SQL-Injection
- Wie erfolgt eine Zugriffsbeschränkung innerhalb von XML-Dokumenten?
  - XPath-Injection hat meist Vollzugriff auf das gesamte XML-Dokument zur Folge!

XPath-Syntax

```
<?xml version="1.0" encoding="UTF-8"?>
<users>
    <user id="1">
        <username>admin</username>
        <password>yoUd0nTg3tTh!s</password>
    </user>
    <user id="2">
        <username>krause</username>
        <password>krause123</password>
    </user>
</users>
```

cirosec

XPath-Syntax

Ausdruck	Ergebnis
/users	Auswahl des Wurzelelements **users**
/users/user	Auswahl der **user**-Elemente, die Kinder von **users** sind
//username	Auswahl aller **username**-Elemente unabhängig von ihrer Position
//user[@id='2']	Auswahl des **user**-Elements mit ID-Parameter 2

```
<?xml version="1.0" encoding="UTF-8"?>
<users>
    <user id="1">
        <username>admin</username>
        <password>yoUd0nTg3tTh!s</password>
    </user>
    <user id="2">
        <username>krause</username>
        <password>krause123</password>
    </user>
</users>
```

cirosec

XPath-Syntax

Ausdruck	Ergebnis
/users	Auswahl des Wurzelelements **users**
/users/user	Auswahl der **user**-Elemente, die Kinder von **users** sind
//username	Auswahl aller **username**-Elemente unabhängig von ihrer Position
//user[@id='2']	Auswahl des **user**-Elements mit ID-Parameter 2

```
<?xml version="1.0" encoding="UTF-8"?>
<users>
    <user id="1">
        <username>admin</username>
        <password>yoUd0nTg3tTh!s</password>
    </user>
    <user id="2">
        <username>krause</username>
        <password>krause123</password>
    </user>
</users>
```

cirosec

XPath-Syntax

Ausdruck	Ergebnis
/users	Auswahl des Wurzelelements **users**
/users/user	Auswahl der **user**-Elemente, die Kinder von **users** sind
//username	Auswahl aller **username**-Elemente unabhängig von ihrer Position
//user[@id='2']	Auswahl des **user**-Elements mit ID-Parameter 2

```
<?xml version="1.0" encoding="UTF-8"?>
<users>
    <user id="1">
        <username>admin</username>
        <password>yoUd0nTg3tTh!s</password>
    </user>
    <user id="2">
        <username>krause</username>
        <password>krause123</password>
    </user>
</users>
```

cirosec

XPath-Syntax

Ausdruck	Ergebnis
/users	Auswahl des Wurzelelements **users**
/users/user	Auswahl der **user**-Elemente, die Kinder von **users** sind
//username	Auswahl aller **username**-Elemente unabhängig von ihrer Position
//user[@id='2']	Auswahl des **user**-Elements mit ID-Parameter 2

```
<?xml version="1.0" encoding="UTF-8"?>
<users>
    <user id="1">
        <username>admin</username>
        <password>yoUd0nTg3tTh!s</password>
    </user>
    <user id="2">
        <username>krause</username>
        <password>krause123</password>
    </user>
</users>
```

cirosec

XPath-Syntax

Ausdruck	Ergebnis
/users	Auswahl des Wurzelelements **users**
/users/user	Auswahl der **user**-Elemente, die Kinder von **users** sind
//username	Auswahl aller **username**-Elemente unabhängig von ihrer Position
//user[@id='2']	Auswahl des **user**-Elements mit ID-Parameter 2

```
<?xml version="1.0" encoding="UTF-8"?>
<users>
    <user id="1">
        <username>admin</username>
        <password>yoUd0nTg3tTh!s</password>
    </user>
    <user id="2">
        <username>krause</username>
        <password>krause123</password>
    </user>
</users>
```

cirosec

Authentisierung über XPath

- Query liefert ID zum gefundenen User:

```
/users/user[username='user'
            and password='pass']/@id
```

- Mögliche XPath-Injection:
  - Username: `admin' or '1' = '1`

```
/users/user[username='admin' or '1' = '1 '
            and password='']/@id
```

cirosec

Demonstration

- XPath-Injection in der cirobank

cirosec

XPath-Injection: Gegenmaßnahmen

- Maßnahmen sind ähnlich zu denen bei SQL
- Eingabevalidierung (Whitelist): Überprüfen jedes Parameters beim ersten Kontakt mit der Applikation, z.B. a-z A-z 0-9
- Sonderzeichen kodieren/escapen (Umwandlung in HTML Entities)
  - Apostroph ‘ → '
- Parameterized XPath (vgl. Prepared Statements)

Gliederung

- Schwachstellen-Überblick
- Präsentation aktueller Angriffstechniken
  - XPath-Injection
  - Cross-Site Request Forgery
- Neue Sicherheitsrisiken durch „Web 2.0“-Technologien wie AJAX

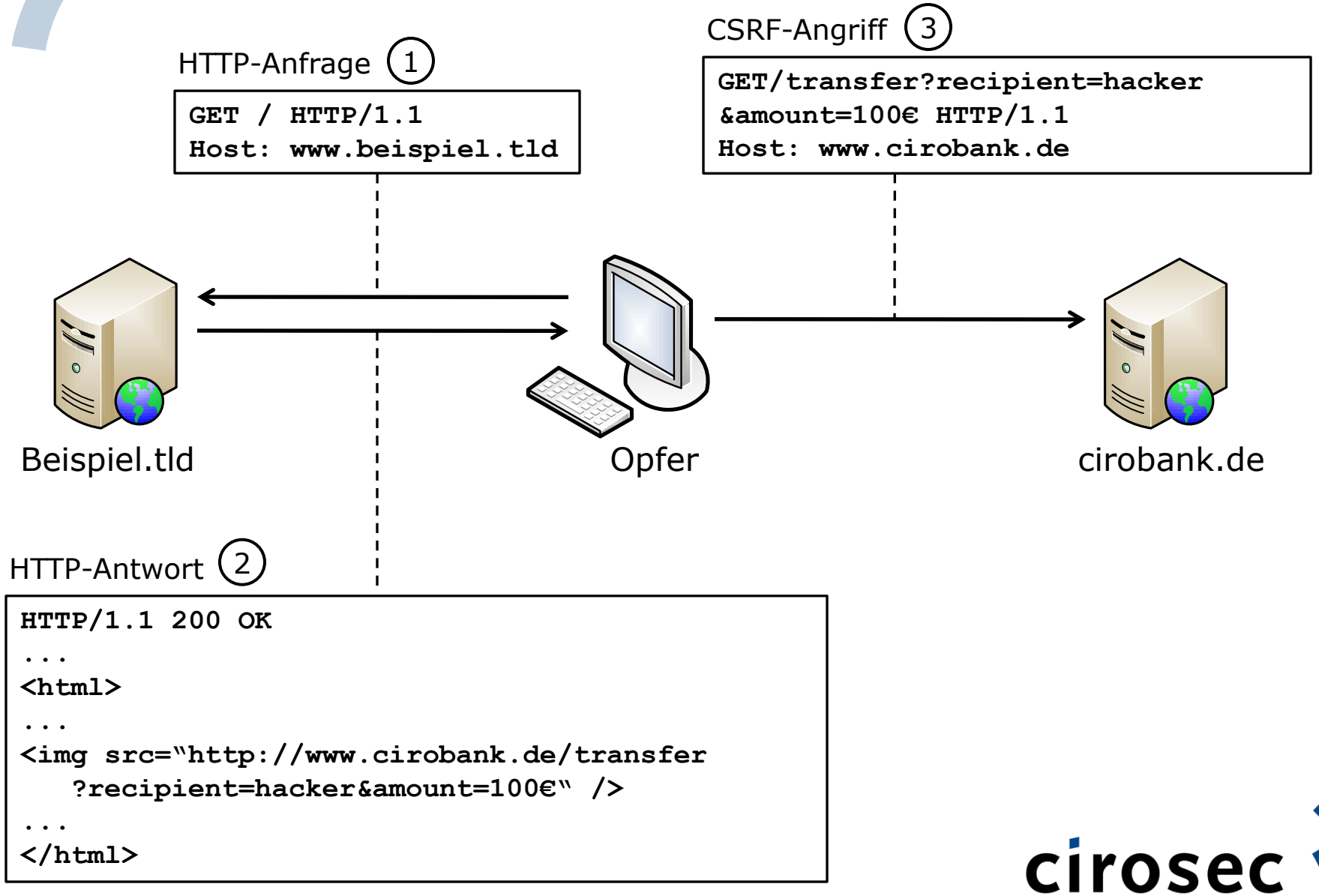
cirosec

Cross-Site Request Forgery (CSRF)

- Andere Begriffe
  - XSRF
  - Session Riding
  - URL Command Attack
- Unbemerktes und unerwünschtes Ausführen von Aktionen im Kontext eines angemeldeten Benutzers, z.B.
  - Passwortänderung
  - Einträge in Foren / Blogs
  - Kaufen / Verkaufen von Aktien
  - …

cirosec

CSRF



cirosec

Cross-Site Request Forgery (CSRF)

- Schwachstelle liegt in der Webanwendung
  - Keine erneute Verifikation der Identität beim Aufruf einer Aktion
  - Anfragen sind statisch und vorhersagbar
  - Es werden keine „Shared Session Secrets“ eingesetzt
- Die Applikation vertraut einem einmal angemeldeten Benutzer

CSRF Angriffsvektoren

- Verteilung über E-Mail-Nachrichten im HTML-Format
- Einbinden in andere bekannte Internetseiten
  - Gästebücher, Foren, etc.
  - Die Möglichkeit Bilder einzubinden reicht aus, um eine Angriffs-URL einzubetten
- Ausnutzung persistenter/nicht-persistenter Cross-Site-Scripting-Schwachstellen

Demonstration

- Cross-Site-Request Forgery (CSRF) innerhalb der cirobank

cirosec

CSRF erkennen

- Aus Sicht der Webapplikation schwierig, da
  - Legitimer HTTP-Request
  - Aufruf einer legitimen Applikations-Funktion
  - Anfrage von einem authentisierten Benutzer

- Erkennung über den HTTP-Referer?
  - Beispiel:

CSRF erkennen (HTTP-Referer)

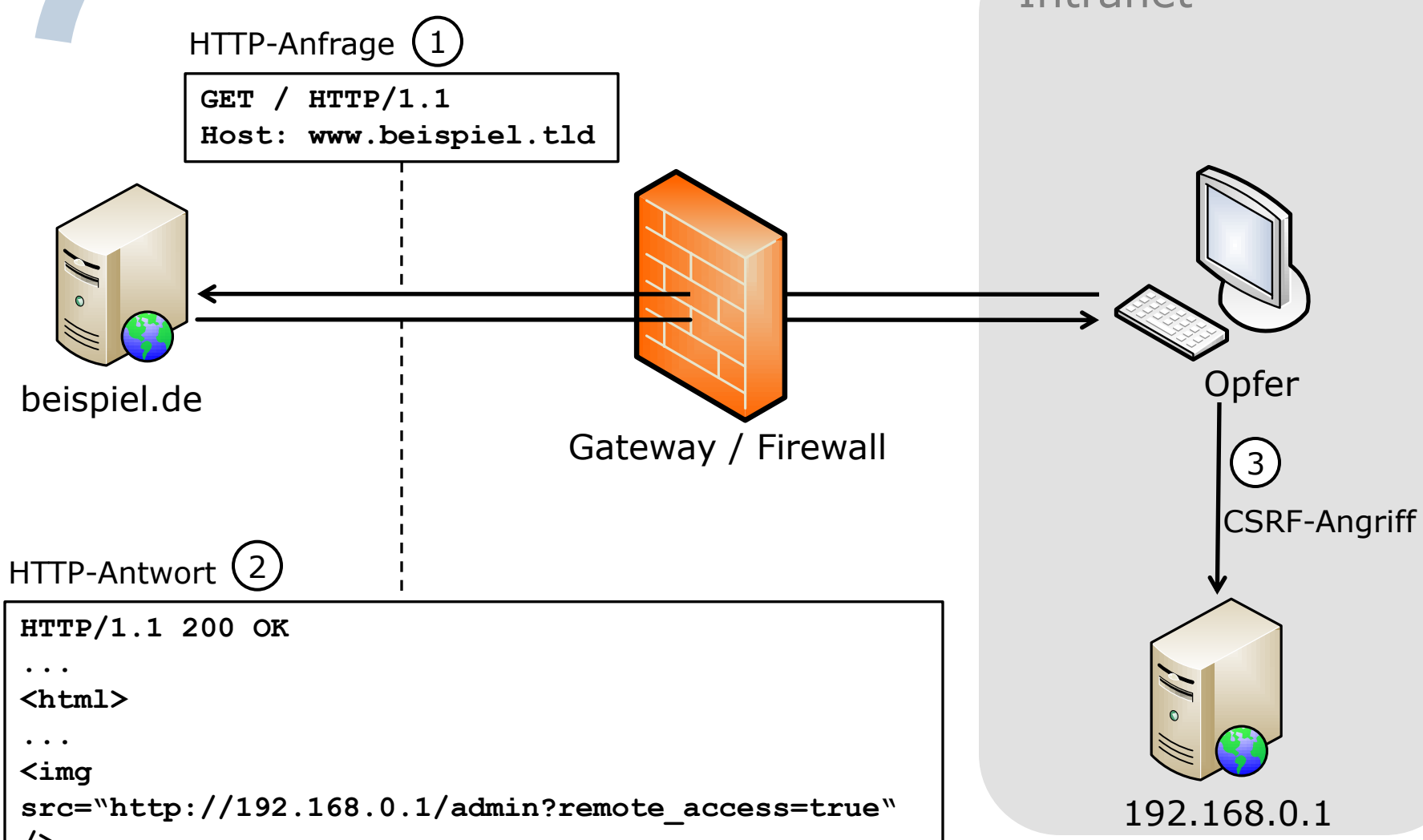
- Facebook wertet den HTTP-Referer aus:
  - Referer: `http://www.facebook.com/` ✓
  - Referer: `http://www.angreifer.de/csrf/` ✗
- Was passiert, wenn kein Referer vorhanden ist?

cirosec

CSRF erkennen (HTTP-Referer)

- Facebook wertet den HTTP-Referer aus:
  - Referer: `http://www.facebook.com/` ✓
  - Referer: `http://www.angreifer.de/csrf/` ✗
- Was passiert, wenn kein Referer vorhanden ist?
  - Warnmeldung wird eingeblendet
  - Anfrage wird dennoch ausgeführt
  - Hintergrund: Proxies, Browser, Anti-Viren-Software etc. Filtern den HTTP-Referer → Benutzer würden ausgesperrt werden

CSRF ins interne Netz



cirosec

CSRF-Angriff auf die FRITZ!Box

- Standardmäßig ist das Webinterface der FRITZ!Box nicht passwortgeschützt
- Die FRITZ!Box prüft ebenfalls den HTTP-Referer ☺

Demonstration

- Aktivieren des Remote-Zugangs einer FRITZ!Box über CSRF

cirosec

Schutz vor CSRF

- Behebung der Schwachstelle auf Server-Seite:
  - Zufalls-Token, so dass eine Anfrage nicht vorhergesagt werden kann
  - Einsatz von CAPTCHA / TAN-Nummern
- Schutz auf Client-Seite:
  - **Abmelden** nach Benutzung einer Applikation
  - Browser-Plugins, wie z.B. NoScript (Firefox)

cirosec

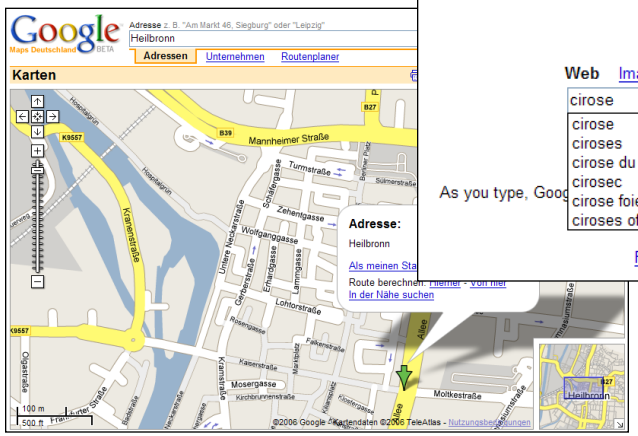
Gliederung

- Schwachstellen-Überblick
- Präsentation aktueller Angriffstechniken
  - XPath-Injection
  - Cross-Site Request Forgery
- Neue Sicherheitsrisiken durch „Web 2.0“-Technologien wie AJAX

cirosec

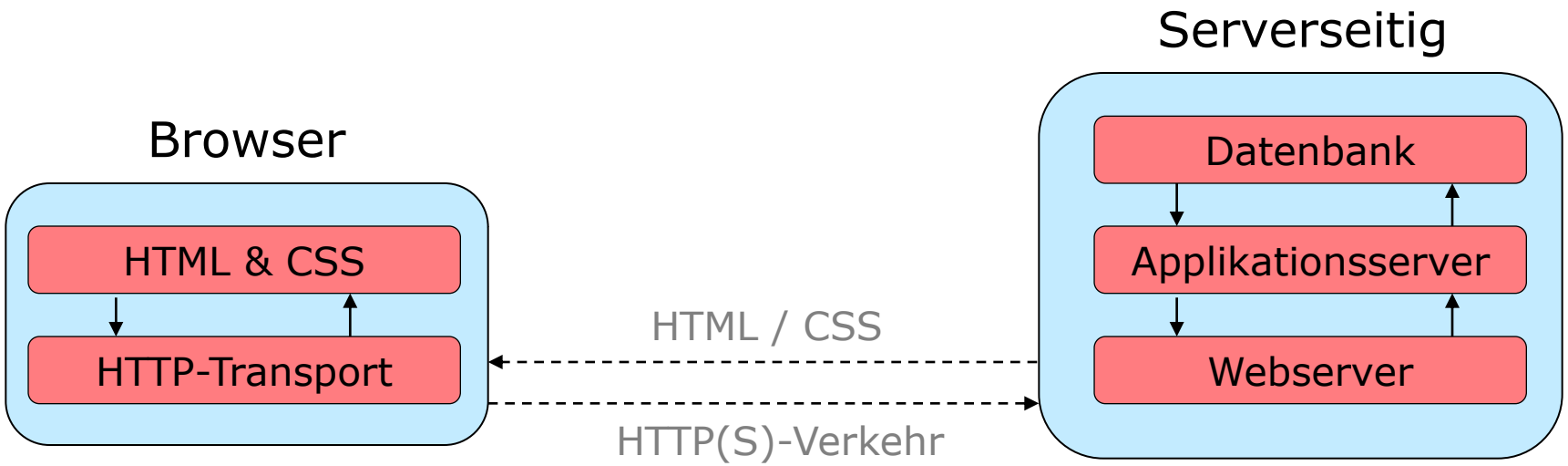
Was ist AJAX?

- **A**synchronous **J**avaScript **a**nd **X**ML
- HTTP-Anfragen innerhalb einer HTML-Seite, ohne die Seite komplett neu laden zu müssen
- Seit 2005, Technik existiert in vergleichbarer Form aber schon seit 1998 (Outlook Web Access/IE4)
- Nutzung in vielen bekannten Websites:
  - Google Suggest
  - Google Maps
  - Flickr
  - Del.icio.us
  - …



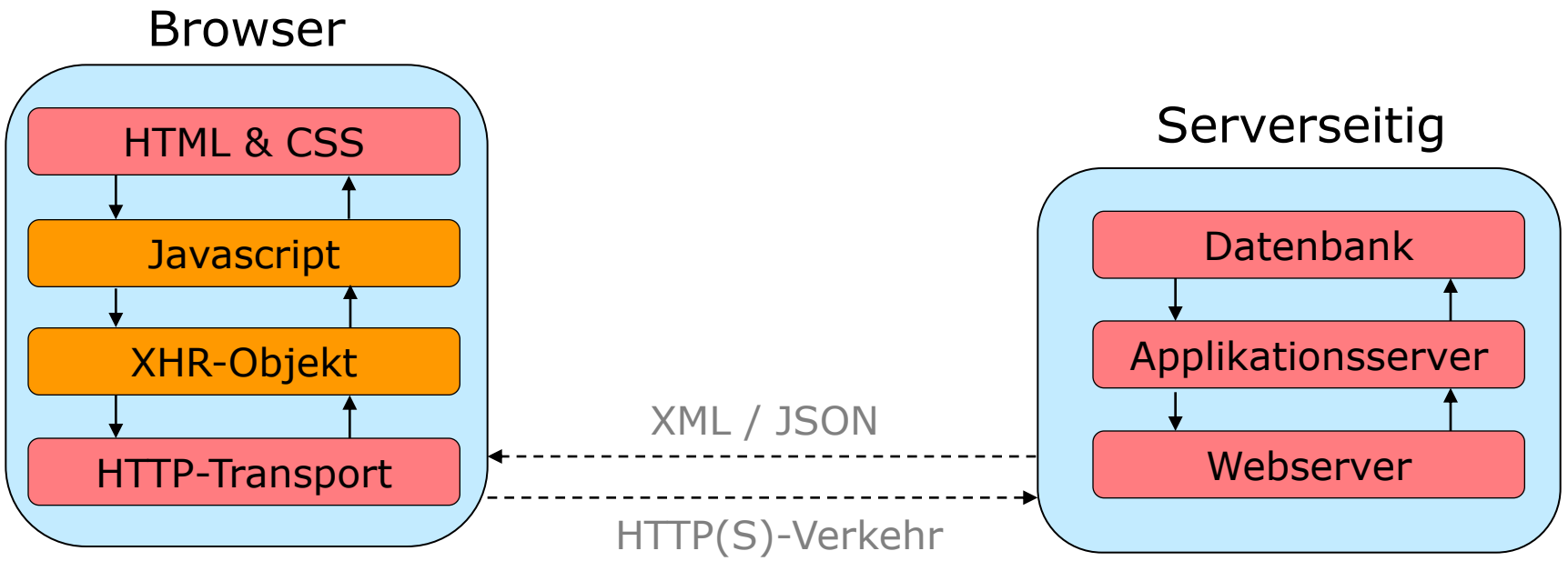
cirosec

Klassisches Modell einer Webanwendung



cirosec

AJAX-Modell einer Webanwendung



- Verlagerung der Applikationslogik auf Clientseite

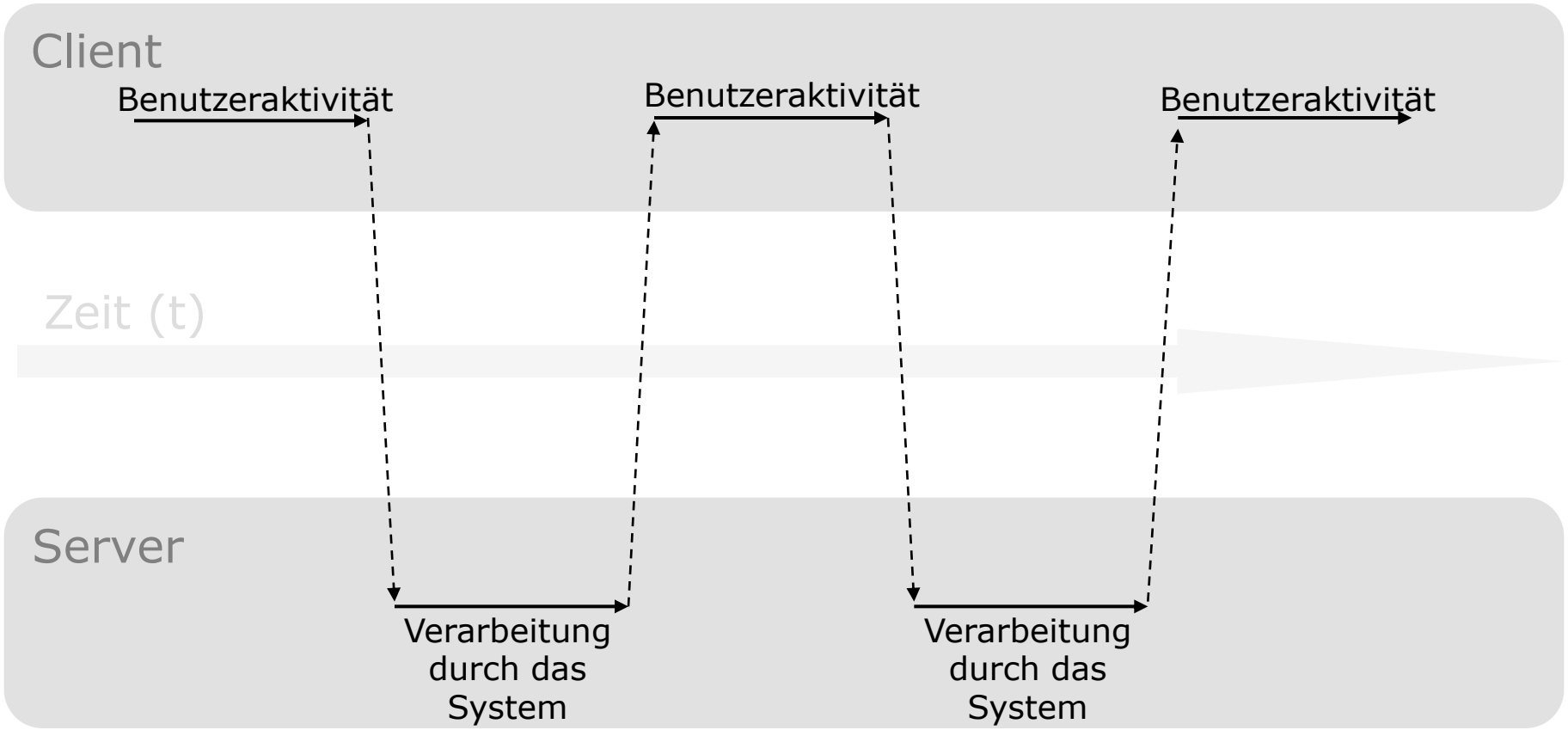
cirosec

Kombination von Technologien

- HTML/XHTML
- *Document Object Model* zur Repräsentation der Inhalte
- JavaScript zur Manipulation des DOM
- XMLHttpRequest-Objekt für den asynchronen Datenaustausch mit dem Webserver

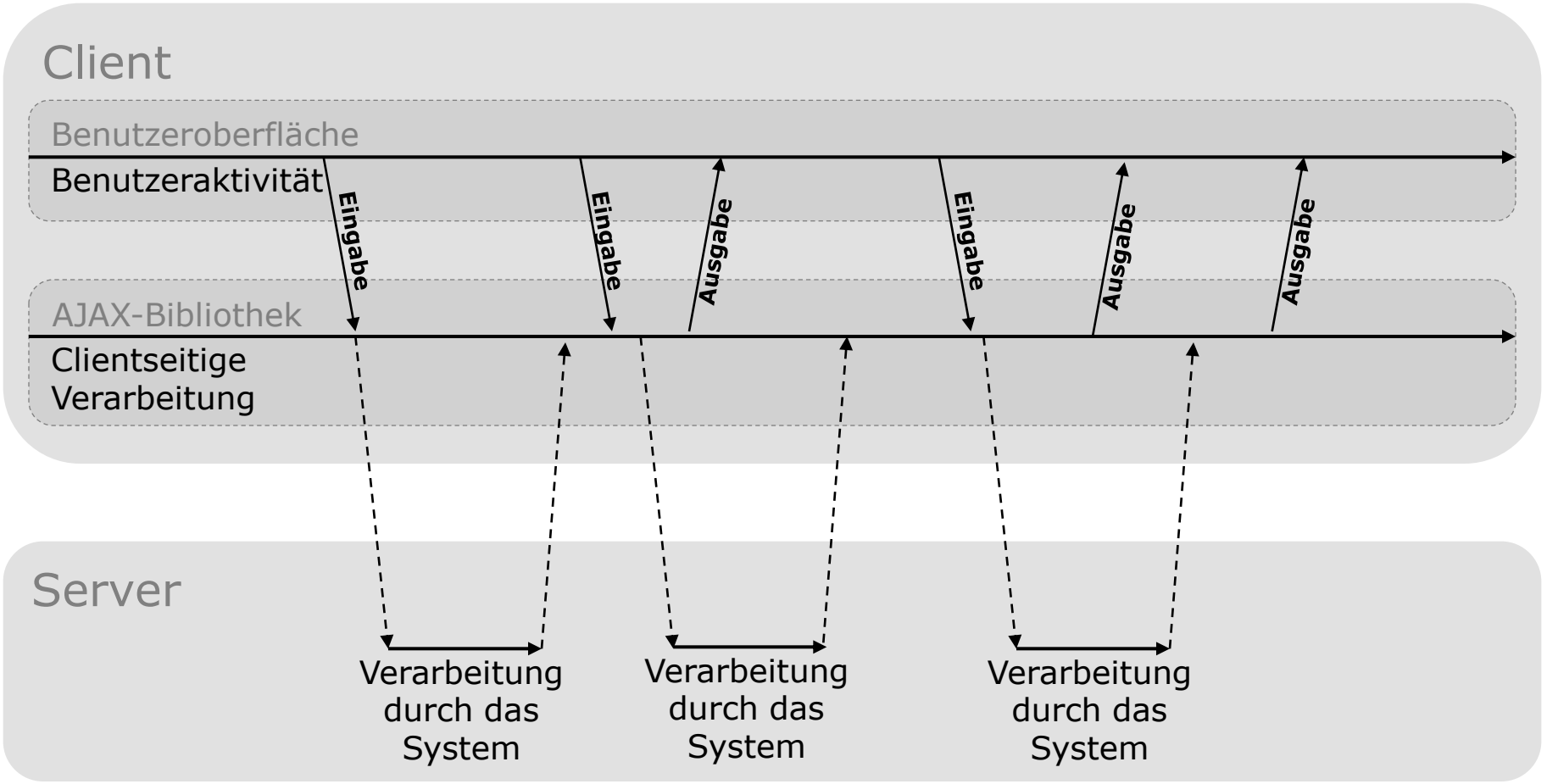
cirosec

klassischer Prozessfluss



cirosec

AJAX Prozessfluss



cirosec

Codebeispiel AJAX

```
xmlHttp = new XMLHttpRequest();
…
xmlHttp.open('GET', 'beispiel.xml', true);
xmlHttp.onreadystatechange = function () {
   if (xmlHttp.readyState == 4) {

       alert(xmlHttp.responseText);
   }
};
…
```

cirosec

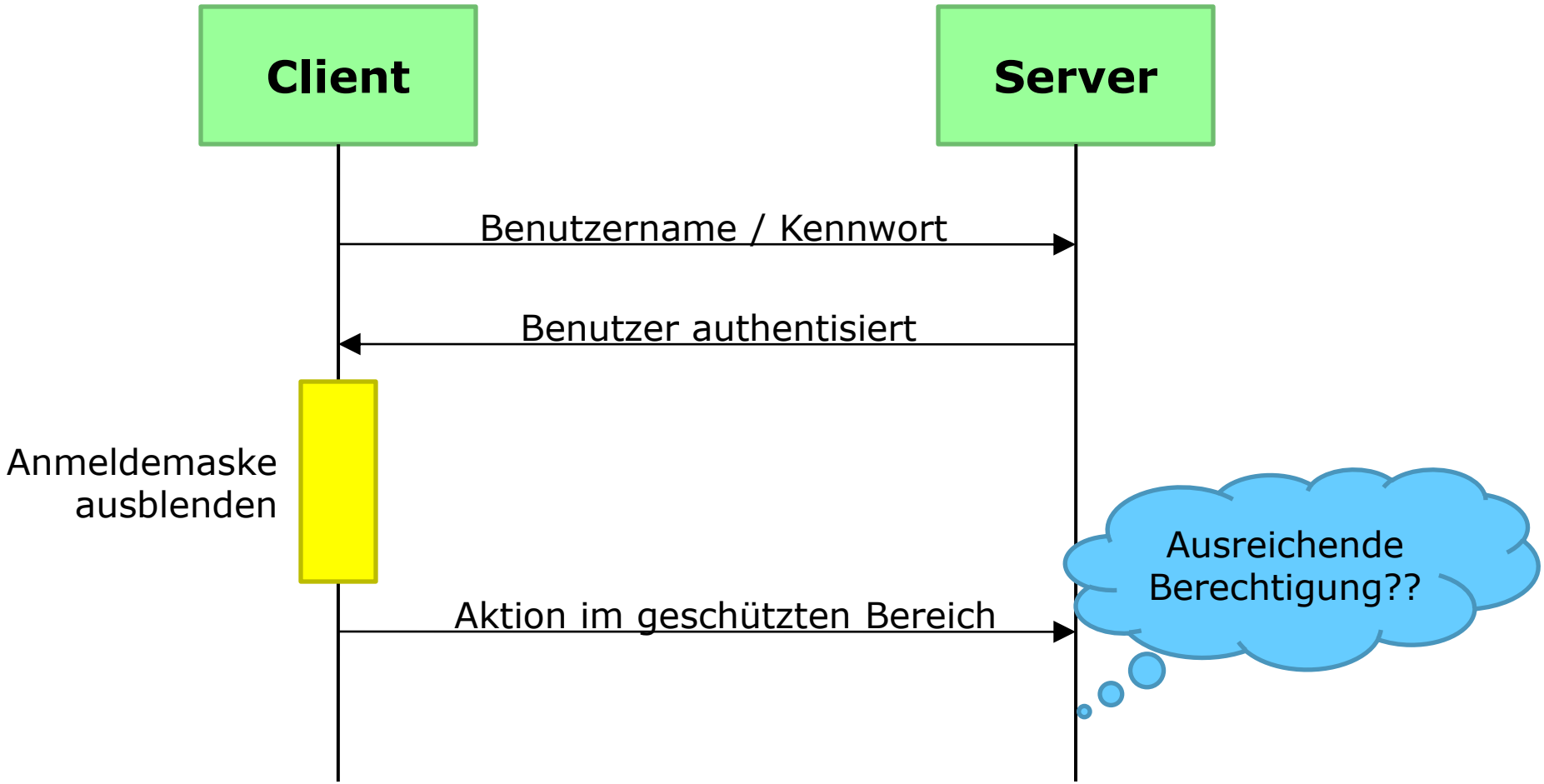
Alte Gefahren…

- …bleiben die Gleichen
- AJAX-Anfragen sind ganz normale HTTP-Requests, die der Webserver nicht unterscheiden kann
- Bekannte Angriffe wie SQL-Injection, XSS oder File-Inclusion bestehen auch auf AJAX-Basis weiter fort

Neue Sicherheitsrisiken von AJAX

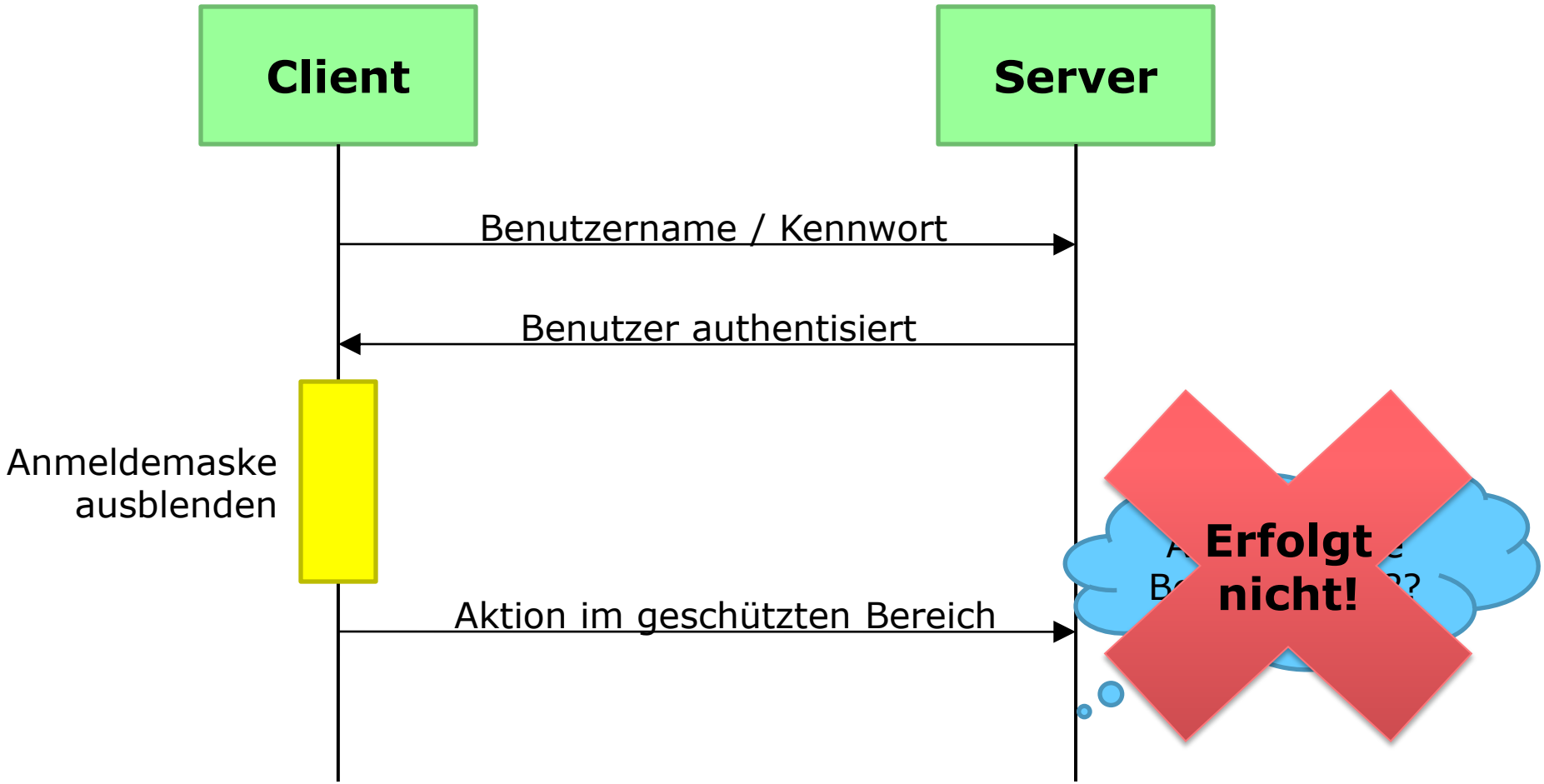
- Server-seitig:
  - Vergrößerung der Angriffsoberfläche durch mehr Parameter, die geprüft werden müssen
  - Unzureichende Eingabevalidierung der Anfragen
  - Unauthentisierte/unautorisierte Nutzung von AJAX-Schnittstellen
- Client-seitig:
  - Verlagerung der Logik auf Client-Seite
  - Ausführung von JavaScript-Code in AJAX-Responses auf dem Client

Umgehung der Authentisierung



cirosec

Umgehung der Authentisierung



Demonstration

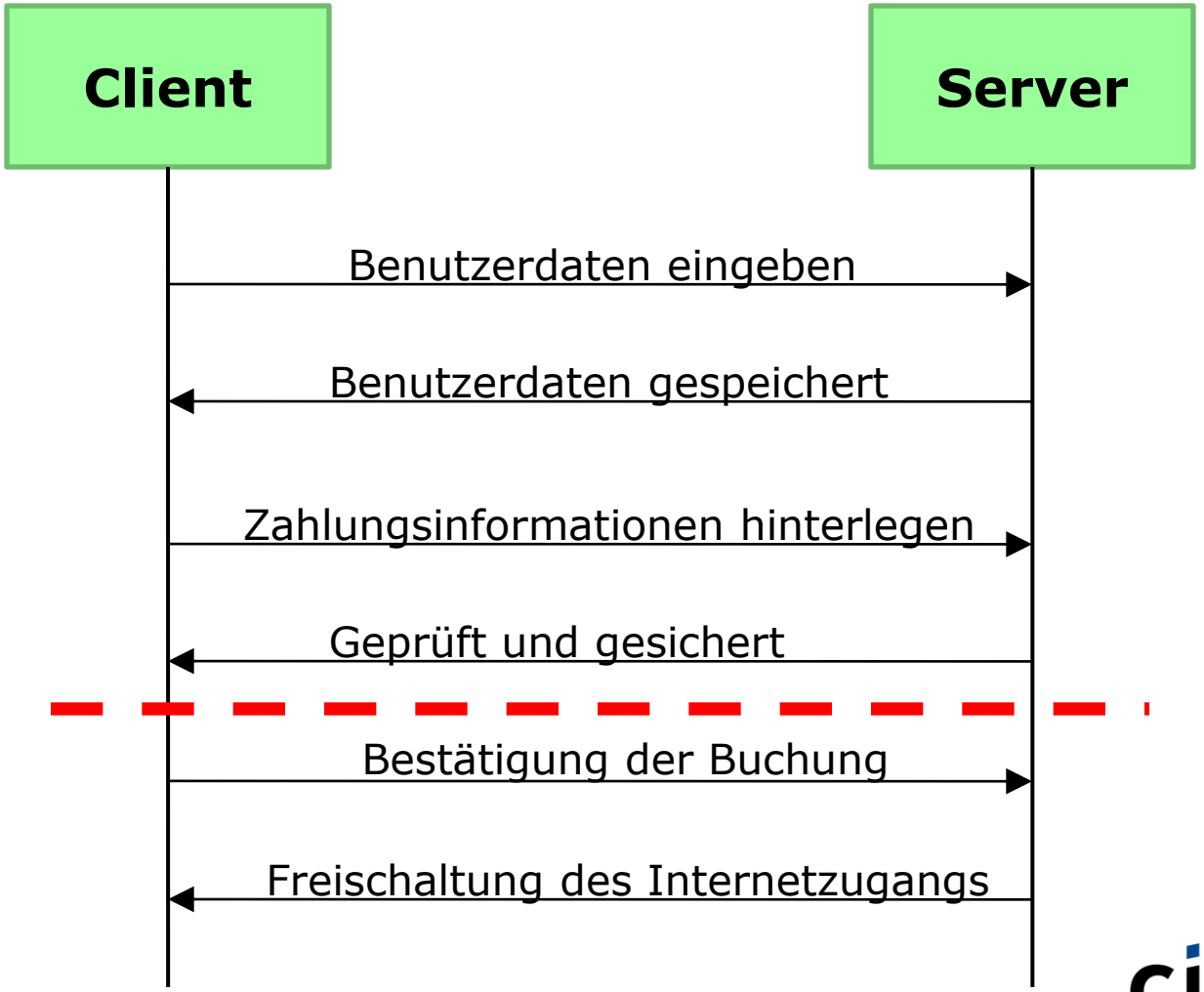
- Umgehung einer AJAX-basierten Authentisierung

cirosec

Race Conditions

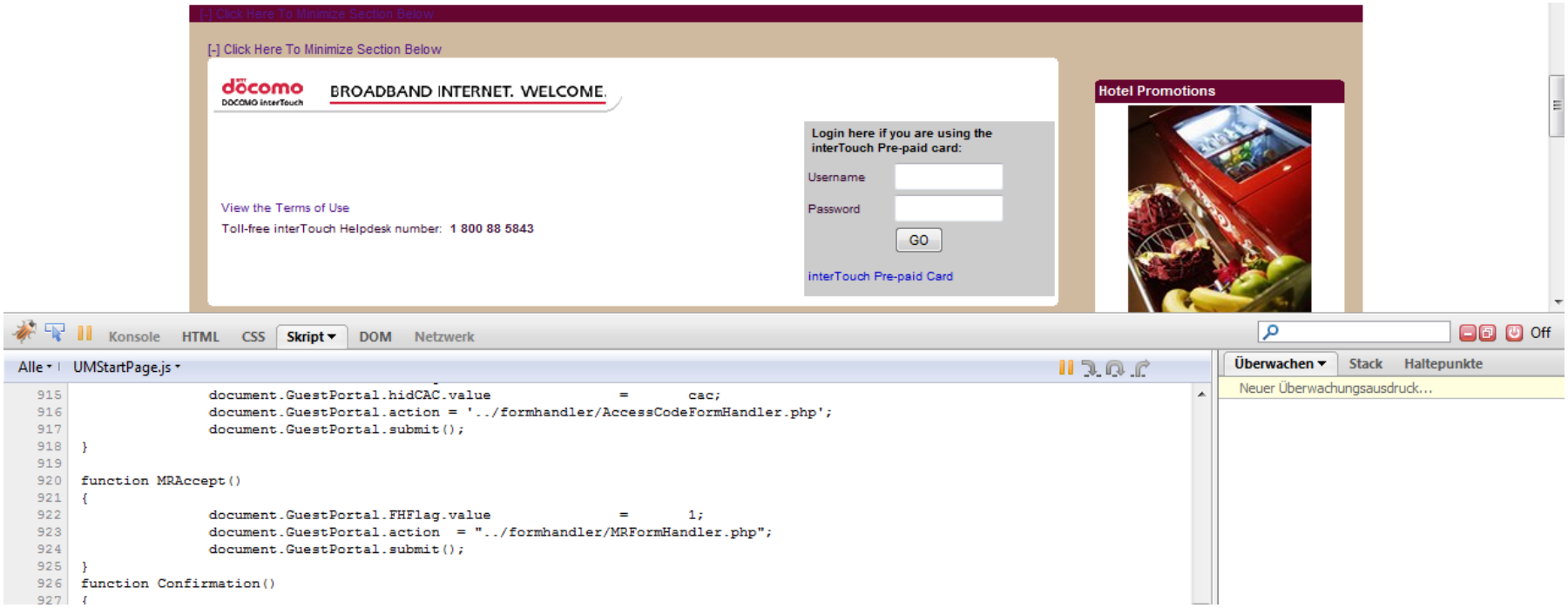
- Auch ein Resultat der Verlagerung von Logik auf Clientseite
- Anfragen werden vom Browser in einer bestimmten Reihenfolge an den Webserver gesendet (Reihenfolge wird vom Client vorgegeben!)
- Angriffspunkt: Reihenfolge manipulieren bzw. bestimmte Anfragen „abfangen“
- Praxis-Beispiel: Internet Hotspot

Race Conditions



cirosec

Race Conditions




```
function Confirmation()
{
                document.GuestPortal.FHFlag.value                    =        1;
                document.GuestPortal.action   = "../formhandler/ConfirmationFormHandler.php";
                document.GuestPortal.submit();
```

cirosec

Demonstration

- Verlagerung von Logik auf Client-Seite

cirosec

Fragen?

cirosec