Netzwerkbasierte Man-in-the-Middle-Angriffe auf die Online-Authentisierung mit dem elektronischen Personalausweis

DFN Workshop 2012

Christian J. Dietrich, Christian Rossow,
Norbert Pohlmann
{dietrich | rossow | pohlmann}[at]internet-sicherheit.de

Institut für Internet-Sicherheit
https://www.internet-sicherheit.de
FH Gelsenkirchen

internet security.

Hintergrund

- Nov. 2010: elektronischer Perso
- **eID Funktion**
  - **Gegenseitige Authentisierung** zwischen Perso und eID-Server
  - **Sichere Übermittlung** (personenbezogener) Merkmale
- Extended Access Control (by BSI)
- 3 Arten an Lesegeräten
  - Basis (kein Pinpad)
  - Standard
  - Komfort

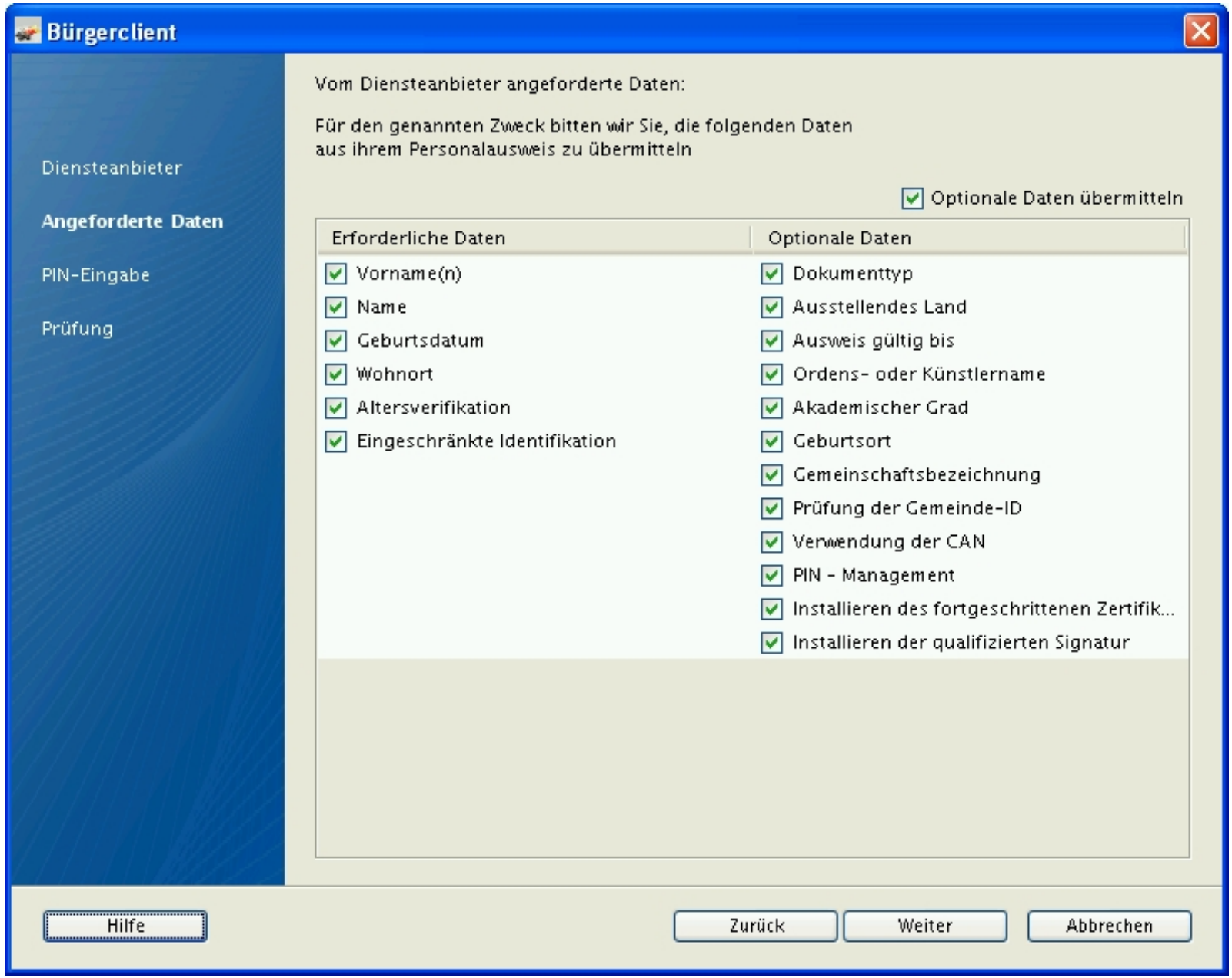
Basisleser

Standardleser

Komfortleser

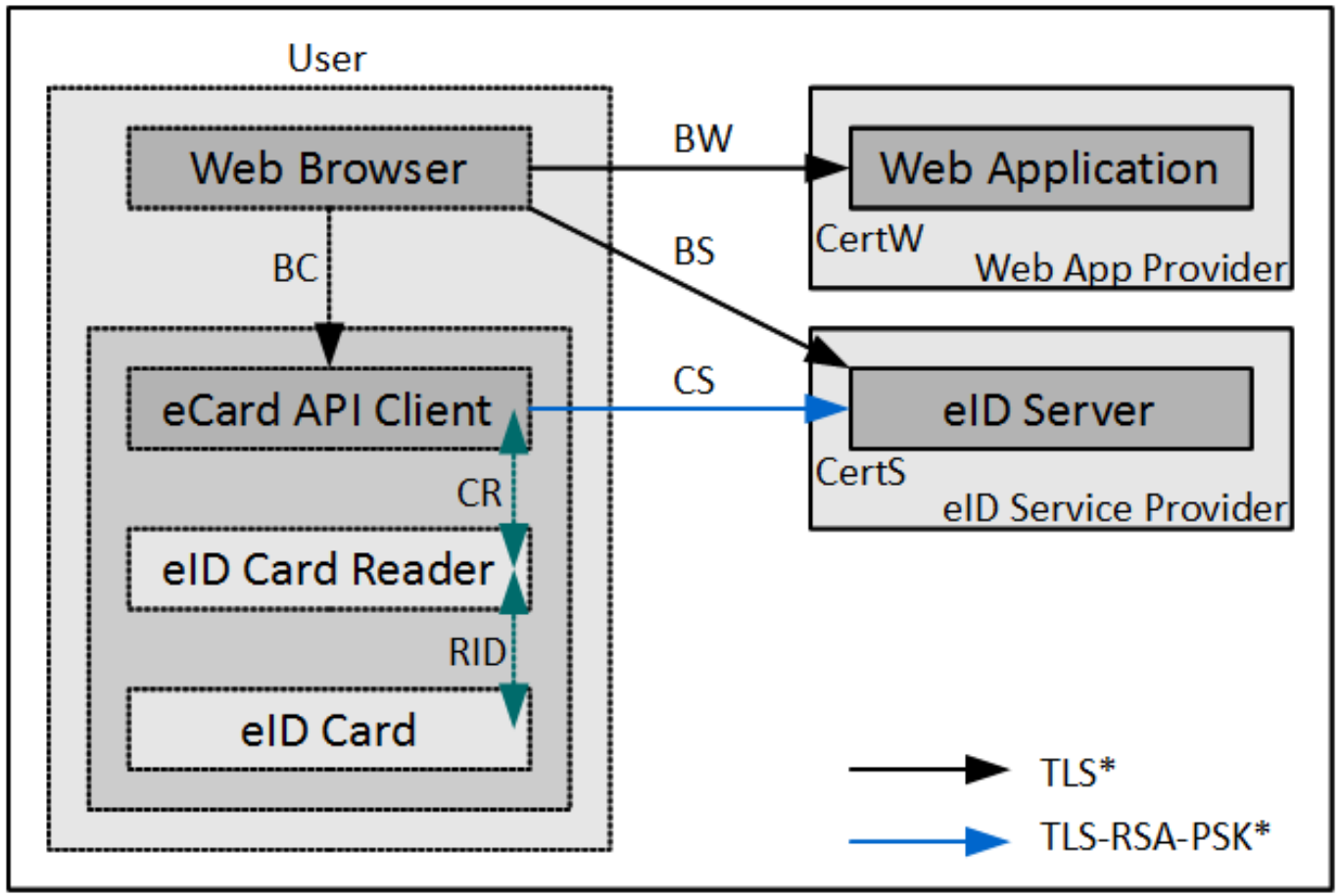
Quelle: https://www.bsi.bund.de/cln_165/sid_0F9C58D2212B77E9DF944B2C45EBA29F/DE/Themen/ZertifizierungundAnerkennung/ZertifizierungnachTR/ZertifizierteProdukte/zertifizierteprodukte_node.html

AusweisApp



KOMMUNIKATIONSMODELL

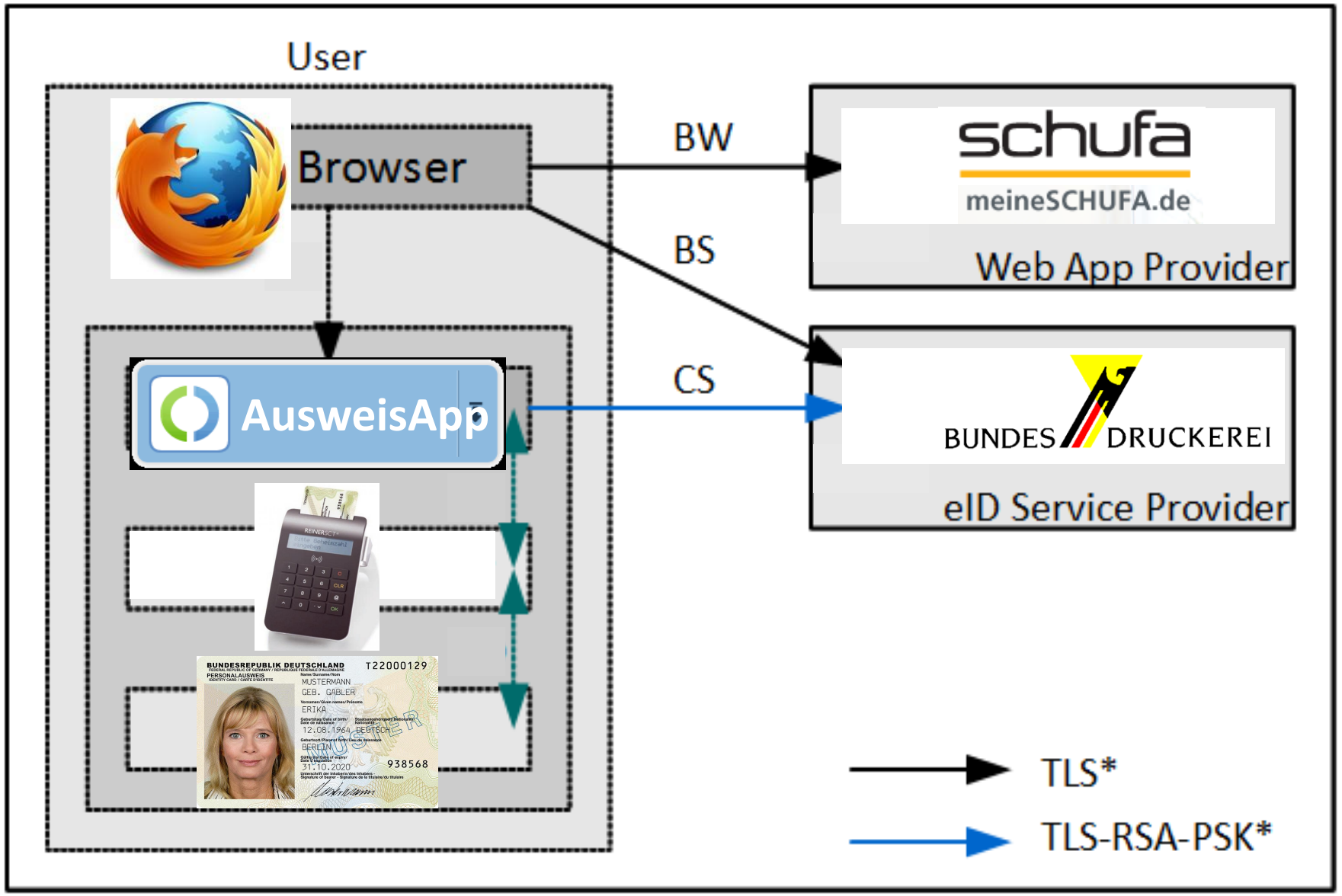
Kanäle der eID Authentisierung



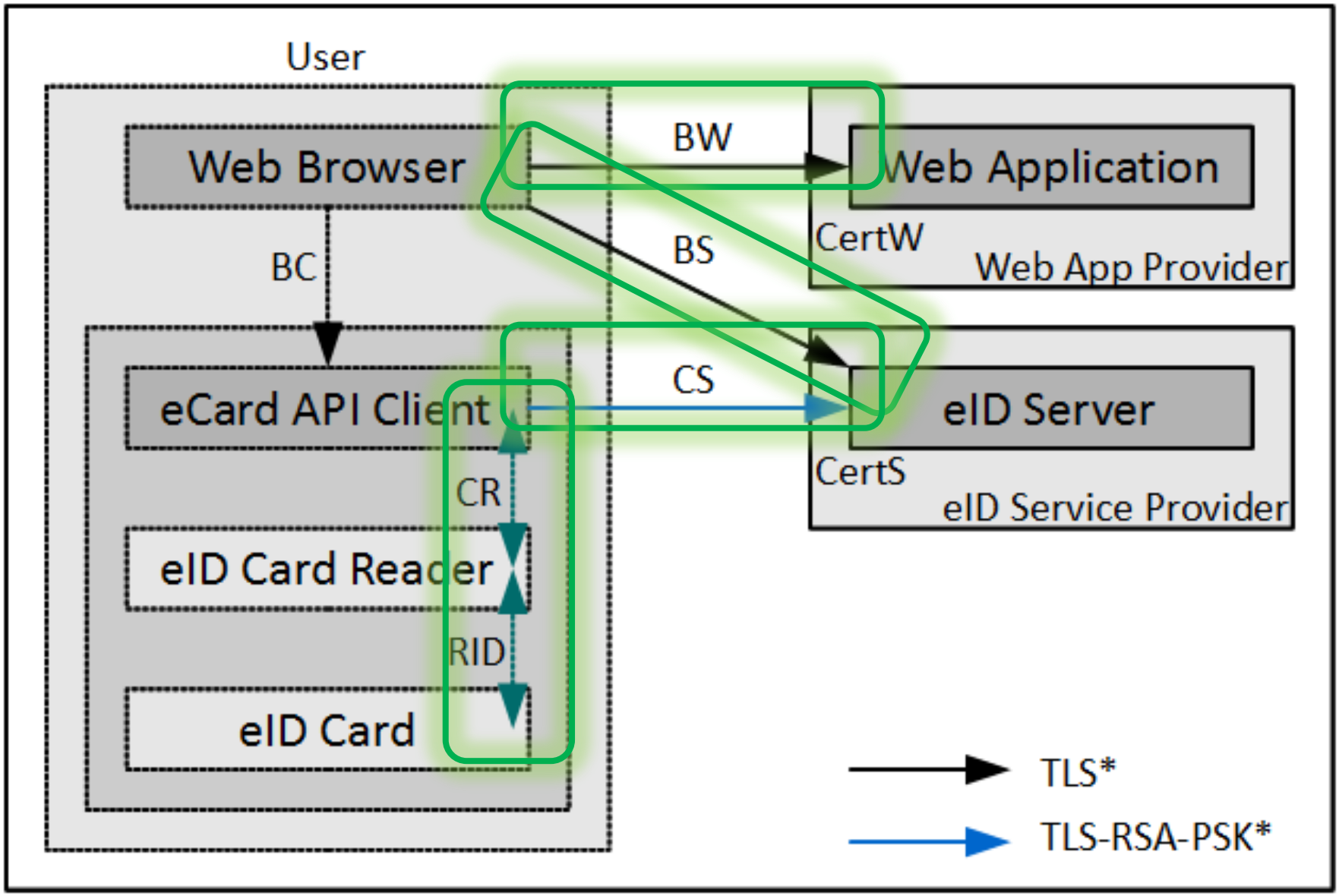
TLS-Kanäle

- Kanal vs. Verbindung
- Das Verbindungs-Management ist Sache des Browsers!!

Beispielhafte Rollenverteilung



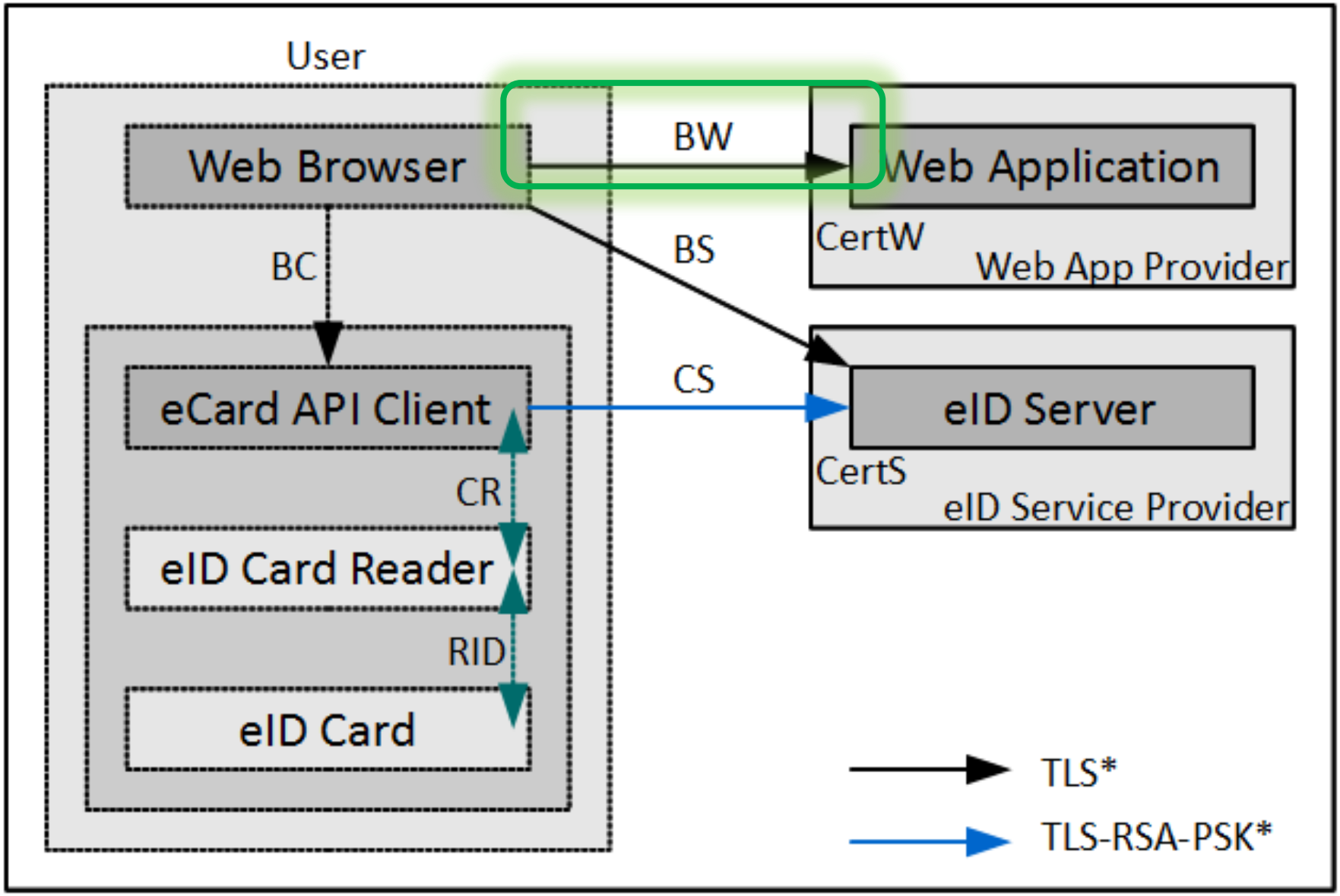
Kanäle der eID Authentisierung



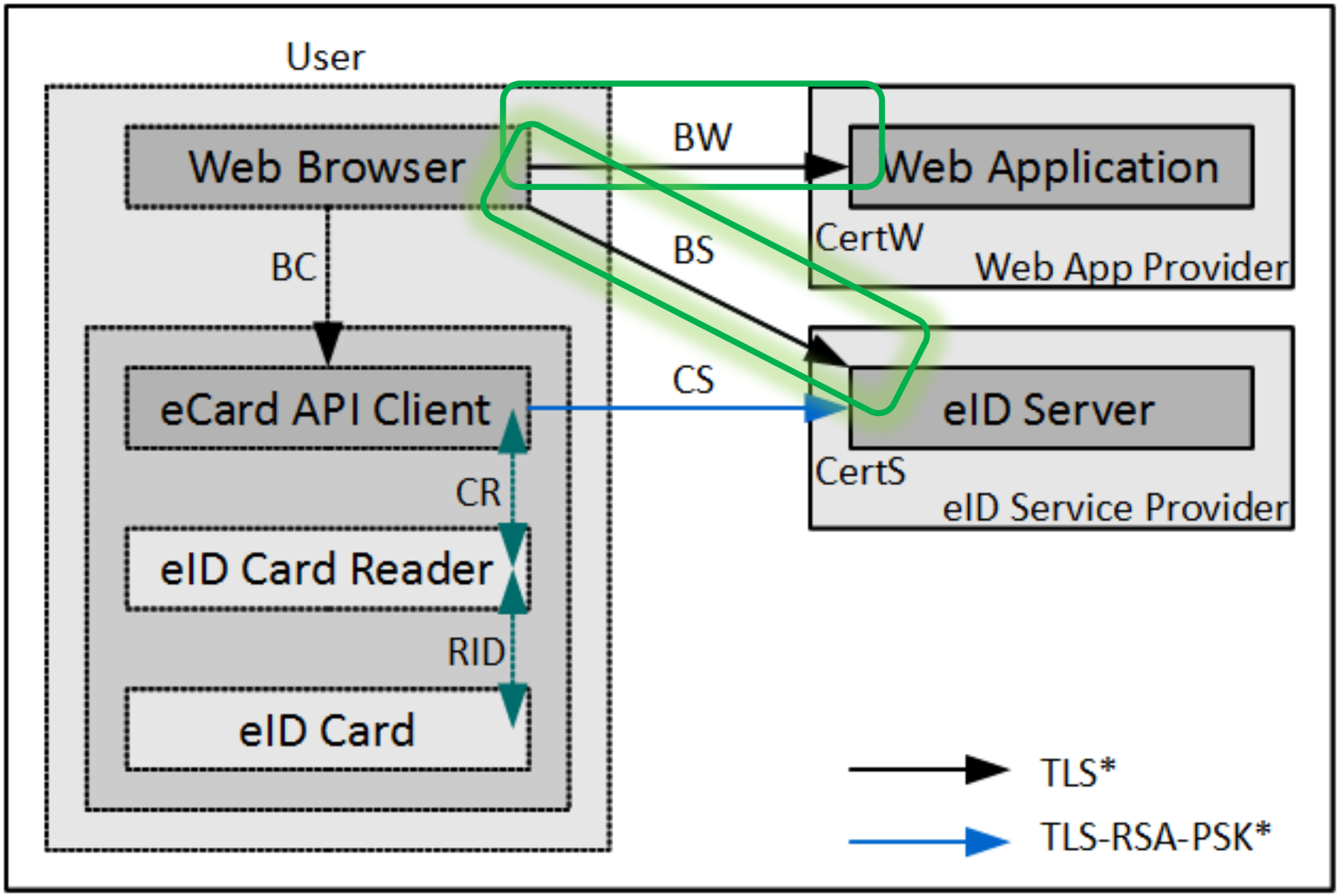
Ein Beispiel

IM ZEITLICHEN ABLAUF

1. Aufruf einer Webseite, die die Online-Authentisierung nutzt



2. Eingebundenes Element des eID-Servers wird geladen



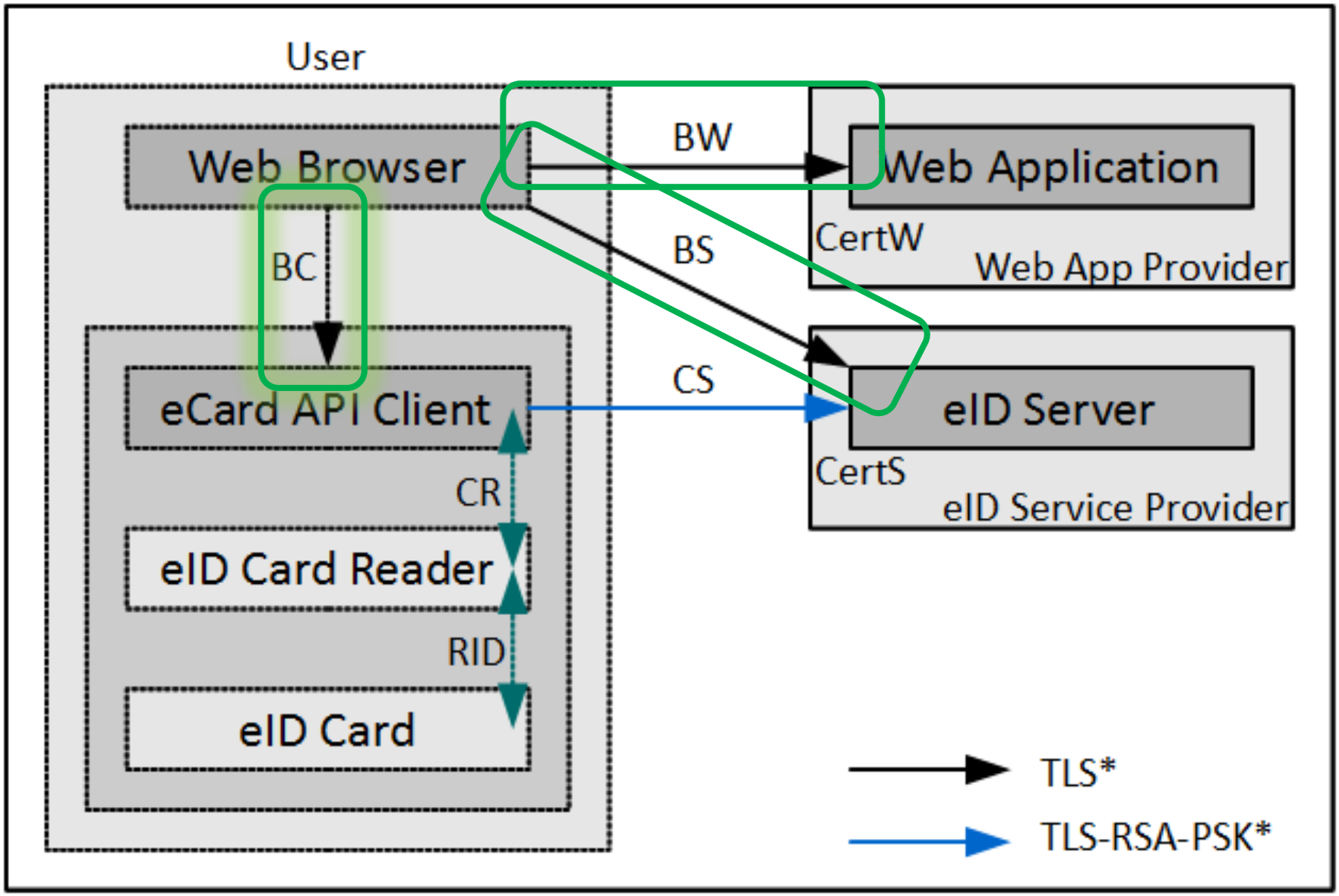
Trigger für die Online-Authentisierung

```
<object type="application/vnd.ecard-client">
  <param name="ServerAddress" value="eid-ref.eid-service.de:443"/>
  <param name="Binding" value="urn:liberty:paos:2006-08"/>
  <!-- RFC 4279 specifies TLS-RSA-PSK -->
  <param name="PathSecurity-Protocol" value="urn:ietf:rfc:4279"/>

  <!-- this is the PSK client identity for the channel CS -->
  <param name="SessionIdentifier" value="54dcf4d212c990c7a768ce51efad"/>

  <!-- this is the PSK for the channel CS -->
  <param name="PathSecurity-Parameters" value="<PSK>c033f5..cf4837</PSK>"/>
  <param name="SHA256ofSAMLRequest" value="MDEwDQYJYIZIAW..ChnhhAxzs7Cy"/>
  <param name="RefreshAddress"
    value="https://eid-ref.eid-service.de:443/epa/plugin?UEsDAL2..QAA%3D%3D"/>
</object>
```

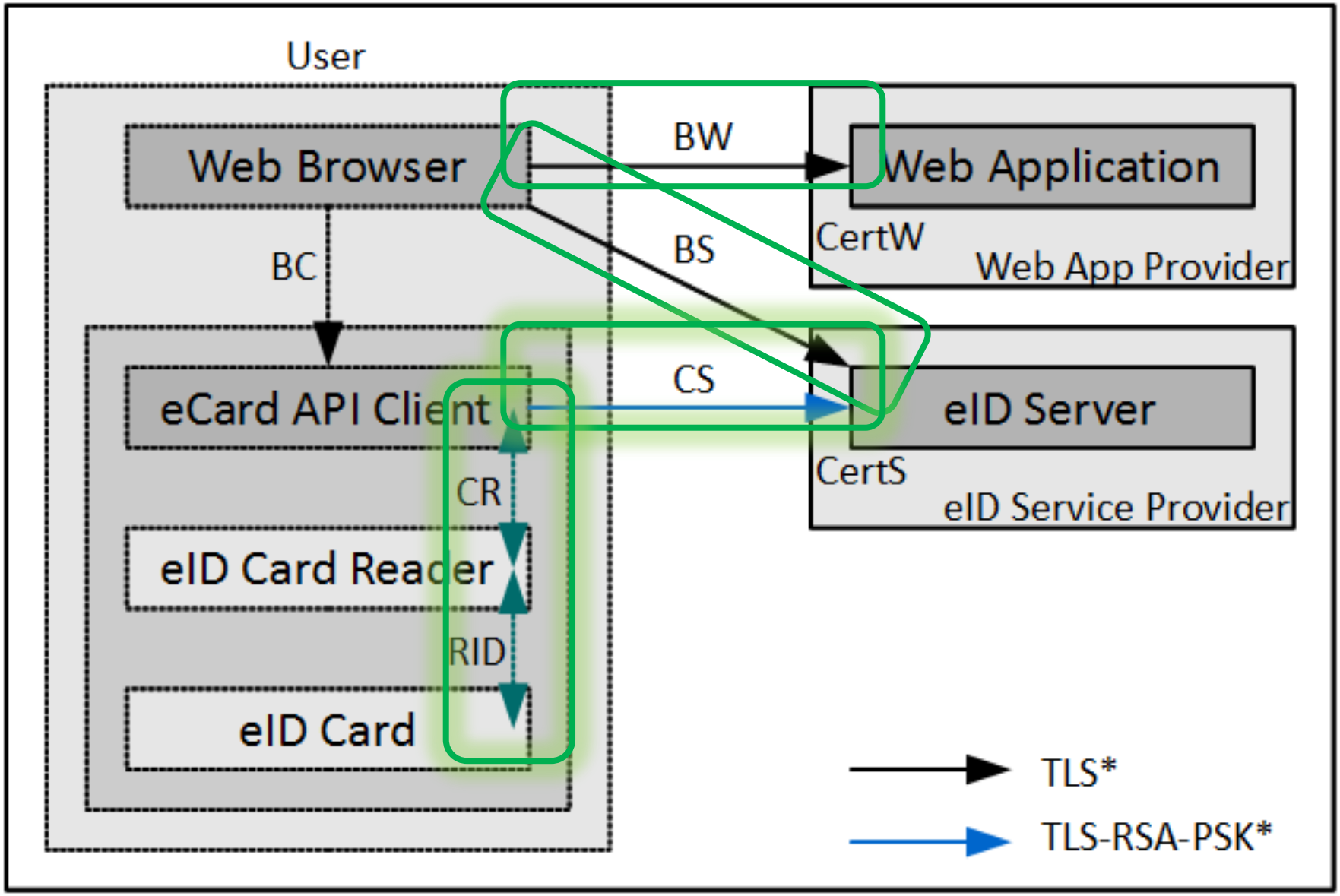
Trigger für die Online-Authentisierung

```
1  <object type="application/vnd.ecard-client">
2   <param name="ServerAddress" value="eid-ref.eid-service.de:443"/>
3   <param name="Binding" value="urn:liberty:paos:2006-08"/>
4   <!-- RFC 4279 specifies TLS-RSA-PSK -->
5   <param name="PathSecurity-Protocol" value="urn:ietf:rfc:4279"/>
6
7   <!-- this is the PSK client identity for the channel CS -->
8   <param name="SessionIdentifier" value="54dcf4d212c990c7a768ce51efad"/>
9
10  <!-- this is the PSK for the channel CS -->
11  <param name="PathSecurity-Parameters" value="<PSK>c033f5..cf4837</PSK>"/>
12  <param name="SHA256ofSAMLRequest" value="MDEwDQYJYIZIAW..ChnhhAxzs7Cy"/>
13  <param name="RefreshAddress"
14    value="https://eid-ref.eid-service.de:443/epa/plugin?UEsDAL2..QAA%3D%3D"/>
15 </object>
```

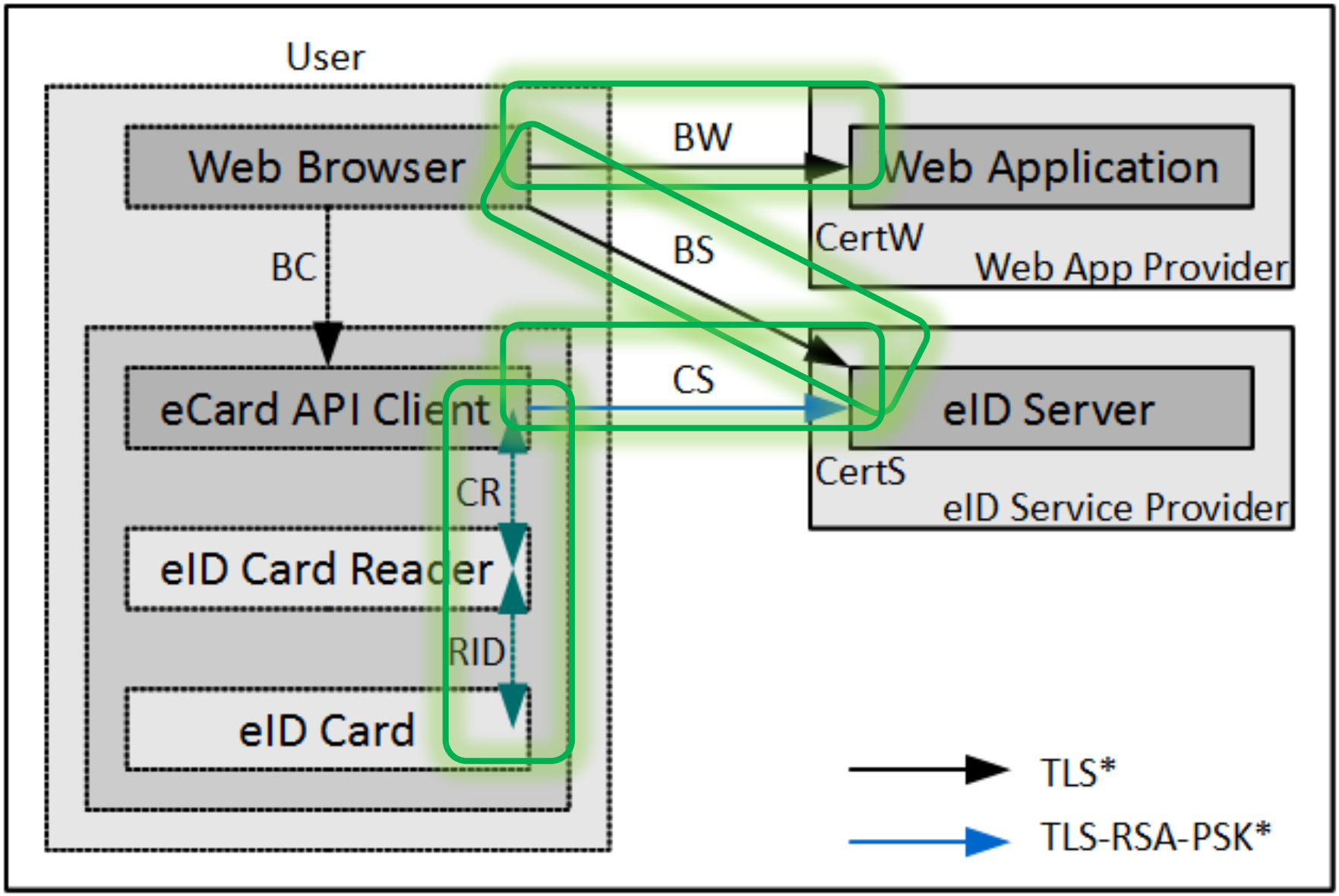
3. eCard API Client wird getriggert



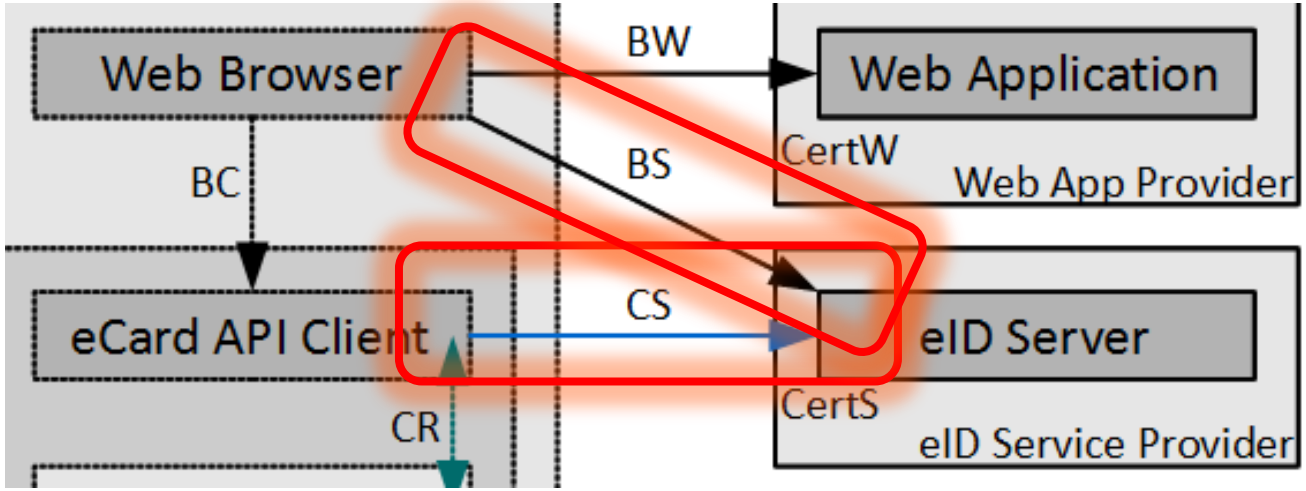
4. eCard API Client initiiert Verbindung mit eID-Server und dem Personalausweis



Ende-zu-Ende nach der Authentisierung

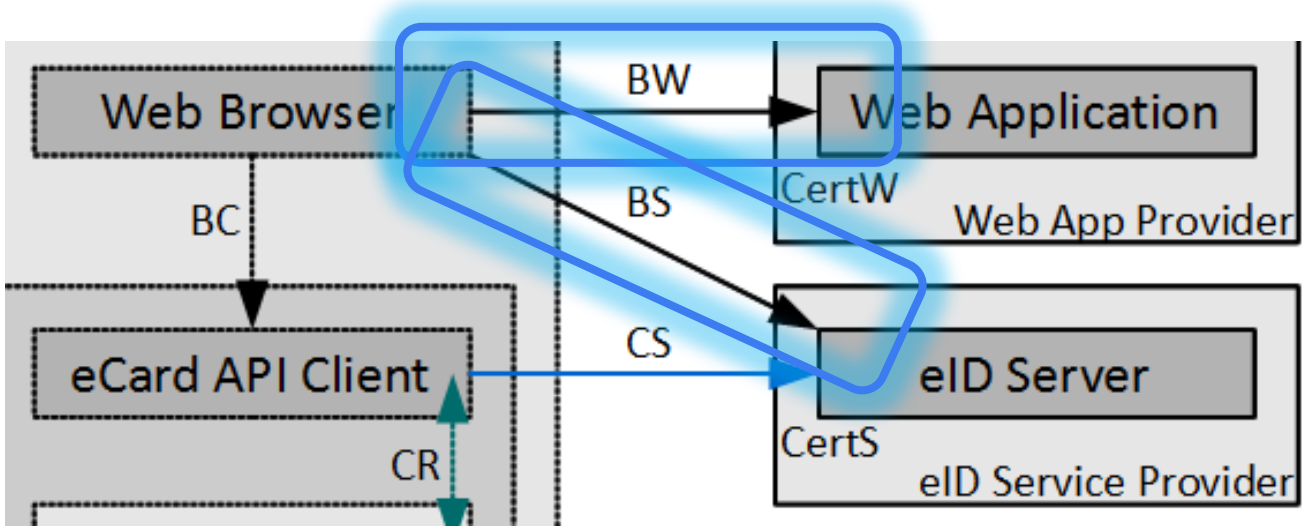


Kanal *CS*: Warum TLS mit RSA-PSK?



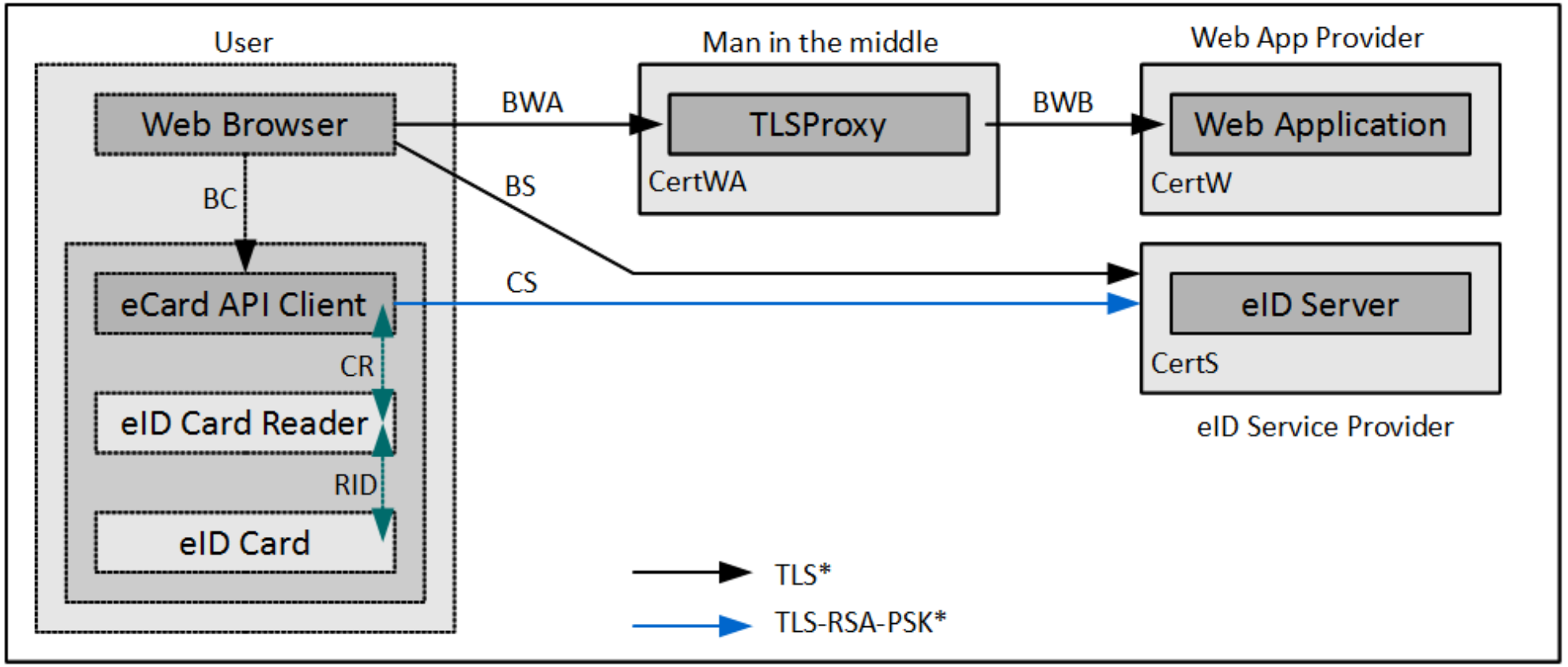
- **PSK** bindet den Kanal *BS* an den Kanal *CS*
  - Über *BS* wurde im Trigger-Objekt der PSK übertragen, der für *CS* benutzt werden muss.
- Wofür die **RSA Server-Authentisierung**?
  - Die Authentisierung ist immer an ein eID-Server (TLS-Zertifikat CertS) und einen Web-App-Server (CertW) gebunden.
  - Die Menge an eID-Servern ist klein.

Binden von BW und BS an einen eID-Server



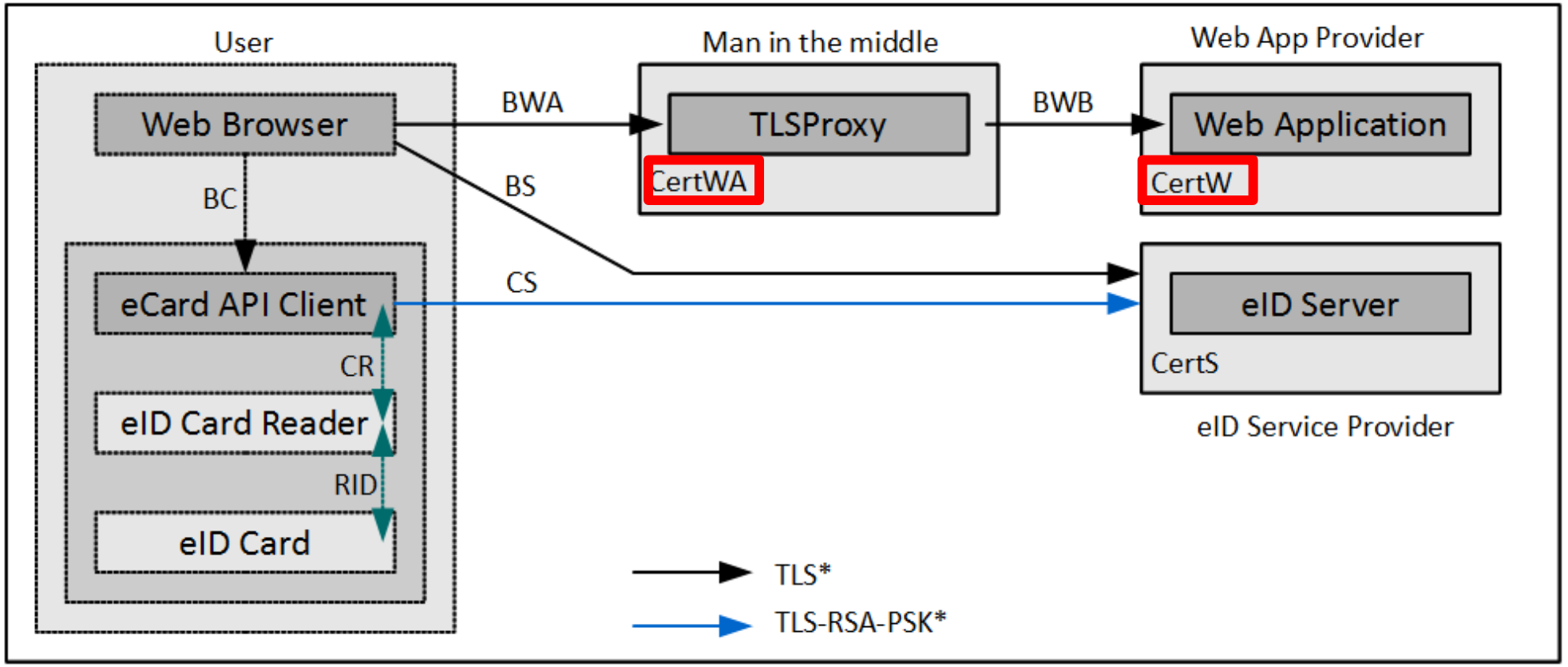
- RSA Server-Authentisierung
- Berechtigungszertifikat enthält die Hashes von **`CertW`** (Web-App-Server) und **`CertS`** (eID-Server)

Man-in-the-middle Angriff auf BW



- Angreifer hat ein eigenes Zertifikat **`CertWA`** (darf sogar gegen eine bekannte Root-CA validierbar sein)

Man-in-the-middle Angriff auf BW



- Nach *BW* und *BS* wird *CS* mit dem echten eID-Server aufgebaut
- Abbruch durch den eCard API Client (AusweisApp), da
  H(**`CertWA`**) ≠ H(**`CertW`**)

Zwischenfazit

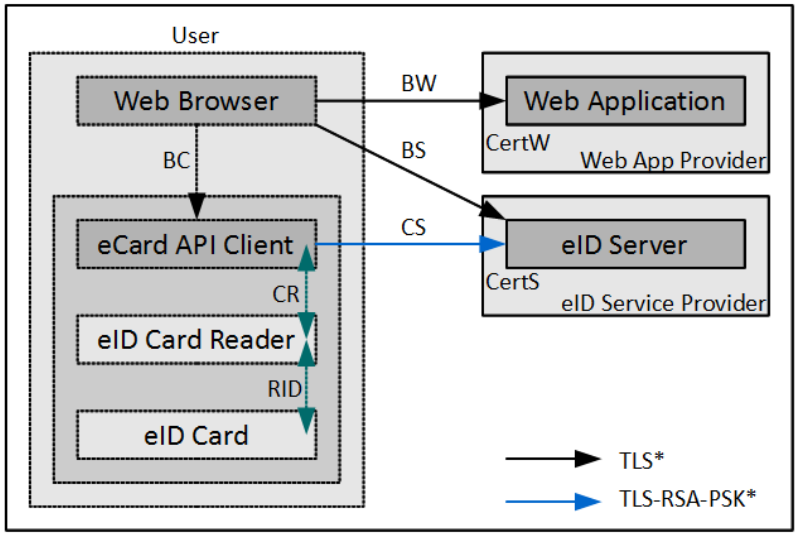
- eCard API Client prüft die Zertifikate aller bestehenden TLS-Kanäle (`CertW` und `CertS`) zum Zeitpunkt der Online-Authentisierung

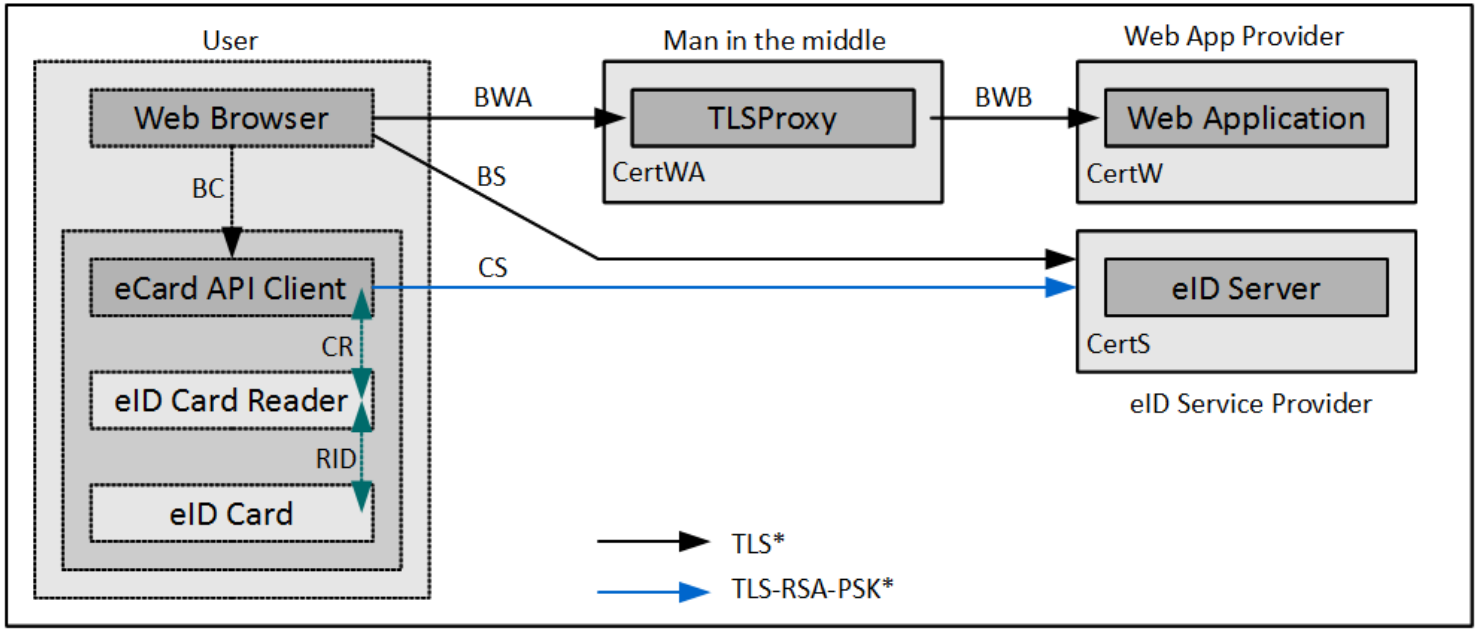
- Das Berechtigungszertifikat enthält die korrekten Hashwerte für `CertW` und `CertS`, eine Modifikation invalidiert das Berechtigungszertifikat und führt zum Abbruch.

- Werden auch nachfolgende Verbindungen innerhalb der TLS-Kanäle **während/nach** der Online-Authentisierung geprüft?

MITM von BW während der Online-Authentisierung

1. Online-Authentisierung (OA) anstoßen
2. AusweisApp führt Zertifikatscheck durch
3. OA verzögern, sodass die Verbindung in BW abgebaut wird
4. MITM starten
5. Online-Authentisierung abschließen
6. MITM erfolgreich





Zwischenfazit

- eCard API Client prüft die Zertifikate aller bestehenden TLS-Kanäle (**`CertW`** und **`CertS`**) zum Zeitpunkt der Online-Authentisierung

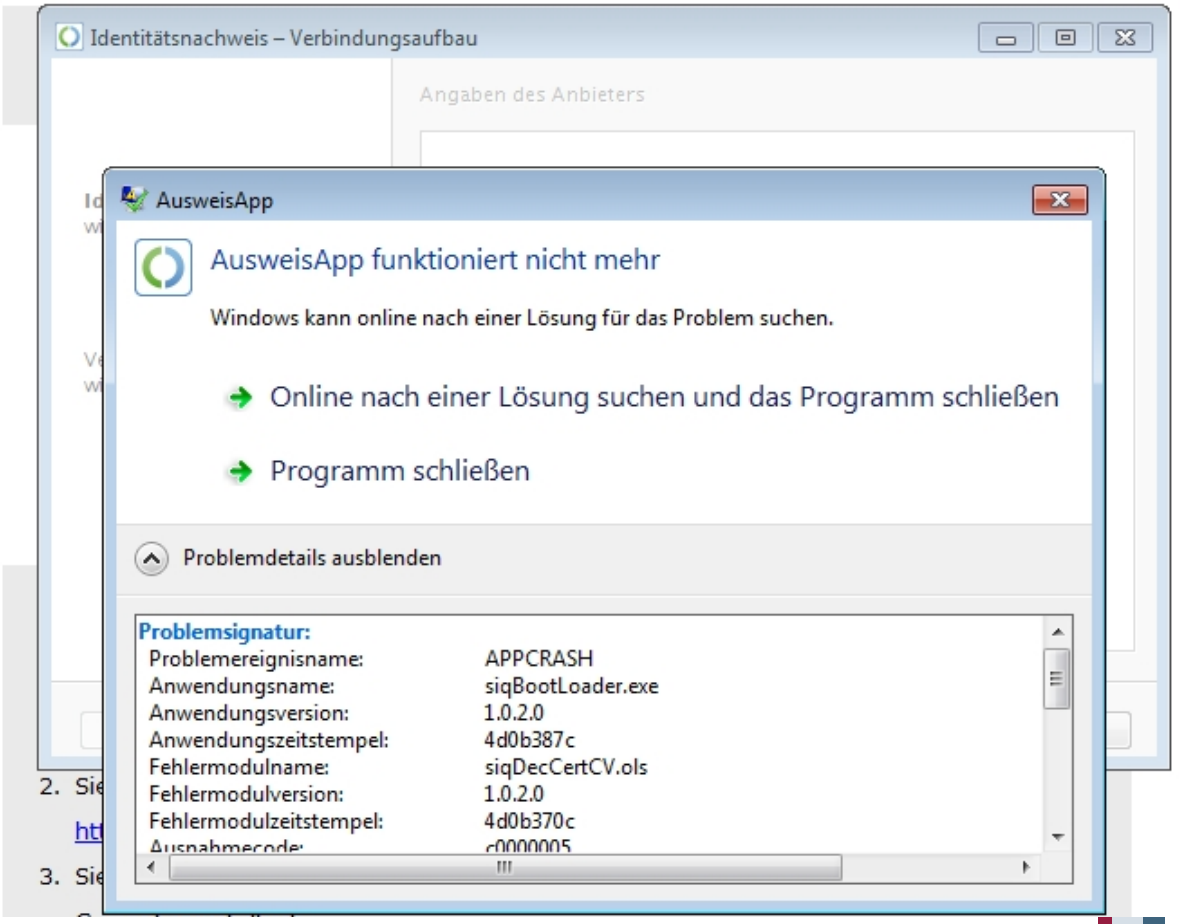
- Das Berechtigungszertifikat enthält die korrekten Hashwerte für **`CertW`** und **`CertS`**, eine Modifikation invalidiert das Berechtigungszertifikat und führt zum Abbruch.

- Werden auch nachfolgende Verbindungen innerhalb der TLS-Kanäle **während/nach** der Online-Authentisierung geprüft?

Fazit

- TLS-Zertifikate: nur „einmalig“ während der Online-Authentisierung geprüft
- Folgende Verbindungen z.B. im Kanal *BW* werden nicht mehr geprüft, sodass hier ein MITM-Angriff auf Folge-Verbindungen erfolgreich ist.
- Aber: Nicht immer gelingt es, die erste Verbindung in *BW* zu brechen. Für typische Angriffe reichen jedoch auch beliebige folgende Verbindungen aus (z.B. um das Cookie für Session Hijacking auszulesen).
- Praxisrelevanz
  - Annahmen: Gültiges TLS-Zertifikat für Opfer-Web-App oder Benutzerfehler, MITM-Position, fehlende Spezifikation
- TLS-Kanäle werden vollständig aufgebaut
  ➔ erlaubt Angriffe auf eCard API Client und eID-Server
- Die Online-Authentisierung sollte ein im Vergleich zu TLS gesteigertes Schutzniveau erreichen – gilt jedoch maximal für die Authentisierungstransaktion.

Fragen?



OpenSSL TLS-RSA-PSK Patch:
https://www.internet-sicherheit.de/service/tools/patches/

Christian J. Dietrich, Christian Rossow,
Norbert Pohlmann
{dietrich | rossow | pohlmann}[at]internet-sicherheit.de

Institut für Internet-Sicherheit
https://www.internet-sicherheit.de
FH Gelsenkirchen

if(is) internet security.