1 Client Authentifizierung (Login mit Zertifikat) unter Apache HTTP Server 1.3.x und 2.x (Dokumentversion 1.4)

Table of Contents

1.1 Allgemeines

Dieses Dokument unterstützt den Ablauf zur Umsetzung einer Client-Authentifizierung auf Zertifikats-Basis für Ihren Apache 1.3.x und 2.x Webserver (http://www.apache.org). Sie bietet Ihren Mitarbeitern und/oder Kunden eine interessante und sichere Alternative zur Anmeldung via Benutzername/Kennwort.

1.2 Voraussetzungen

Um eine Client-Authentifizierung in Ihren Apache 1.3.x oder 2.x Webserver einbinden zu können, sind einige Grundvoraussetzungen notwendig.

Zunächst müssen Sie den Apache Webserver „SSL-fähig" machen – dazu benötigen Sie zusätzlich die zum Apache passende mod_ssl Version (http://www.modssl.org) zur Einbindung des SSL-Modules und openssl (http://www.openssl.org) zur Erzeugung eines Schlüsselpaares und des Certificate Requests. Auf Basis dieses Requests kann Ihnen a.trust ein a.sign corporate SSL Zertifikat ausstellen (Bestellung und Informationen auf http://www.a-trust.at/info.asp?node=243&lang=GE&ch=2).

Ist das SSL Zertifikat in Ihren Webserver eingebunden, sind alle Voraussetzungen geschaffen, um ein Login mit Zertifikat auf Ihrer Homepage zu ermöglichen.

Wie Sie ein SSL Zertifikat in Ihren Apache 1.3.x und 2.x Webserver einbinden können, entnehmen Sie bitte unserer Anleitung

„http://www.a-trust.at/Anleitungen/client_authentification_apache_V104.pdf"

1.3 Client Authentifizierung unter der Version 1.3.x

Wichtiger Hinweis: Bevor Sie Änderungen an der Apache Konfiguration vornehmen, legen Sie bitte unbedingt eine Sicherheitskopie der httpd.conf-Datei an!

Wir zeigen Ihnen, wie Sie in wenigen Schritten Ihren Apache 1.3.x Webserver mit der Möglichkeit einer Client Authentifzierung ausstatten und wie Sie sich mit A-Trust Benutzerzertifikaten bequem und vor allem sicher ausweisen können.

Wir gehen von der SSL-Grundkonfiguration des httpd.conf Files in der Beschreibung

„http://www.a-trust.at/Anleitungen/client_authentification_apache_V104.pdf"

Kapitel 1.5 (Einbinden des SSL-Zertifikates unter der Version 1.3..x) aus.

Zuerst nehmen wir uns die Einträge unter <Virtual Host> vor (die SSL-Befehle außerhalb des Virtual Host Bereiches betreffen nur die SSL-Verbindung selbst, nicht die Client Authentifizierung). Sie haben also folgende Konfiguration:

```
<VirtualHost www.my-domain.com:443>
SSLEngine On
# Angabe und Ort des Server-Zertifikates
SSLCertificateFile conf/ssl/*_b64.crt
# Angabe und Ort des privaten Server-Schlüssels
SSLCertificateKeyFile conf/ssl/my-server.key
</VirtualHost>
```

Zunächst muss ein Root-Zertifikat angegeben werden – alle Zertifikate, die von diesem Root-Zertifikat ausgestellt wurden, können sich authentifizieren:

```
SSLCACertificateFile conf/ssl.crt/A-Trust-Qual-02a.pem
```

Die aktuellen A-Trust Produkte a.sign premium, a.sign light und a.sign token haben z.B. das A-Trust-Qual-02a als Root-Zertifikat – daher ist in unserem Beispiel mit diesen Produkten eine Anmeldung möglich. Produkte, welche nicht von diesem Root-Zertifikat ausgestellt wurden, sind für eine Client Authentifzierung daher unzulässig.

Eine genaue Auflistung der Root-Zertifikate für A-Trust-Benutzerzertifikate finden Sie im Kapitel „1.7 Auflistung der A-Trust Root-Zertifikate (SSLCACertificateFile)"

Gibt es mehrere Root-CA-Zertifikate, die angegeben werden müssen (weil verschiedene Client-Zertifikate berechtigt werden sollen), kann auch nur der Pfad angegeben werden, wo sich diese befinden:

```
SSLCACertificatePath conf/ssl.crt
```

Und nun der wichtigste Teil – die Befehle, um die Client Authentfizierung durchzuführen:

```
SSLVerifyClient require
```

SSLVerifyClient require erfordert eine Authentifzierung mit Zertifikat. Statt require kann auch „optional" (Anmeldung mit Zertifikat möglich, aber nicht zwingend) oder „none" (keine Anmeldung mit Zertifikat erforderlich) angegeben werden.

```
SSLVerifyDepth 2
```

SSLVerifyDepth bestimmt die Zertifikats-Ebene, die beim Client-Zertifikat durchsucht wird. A-Trust Zertifikate haben übergeordnete Ebenen (Zwischeninstanz-Zertifikat, Root-Zertifikat), daher muss SSLVerifyClient 2 angegeben werden.

Zur Information: Apache unterstützt unseres Wissens nach keine Zwischeninstanz-Zertifikate und benötigt daher lediglich das Root-Zertifikat als SSLCACertificateFile.

So sieht also unsere Beispiel-Konfiguration aus:

```
<VirtualHost www.my-domain.com:443>
SSLEngine On
SSLCertificateFile conf/ssl/*_b64.crt
SSLCertificateKeyFile conf/ssl/my-server.key
SSLCACertificateFile conf/ssl.crt/A-Trust-Qual-02a.pem
# Optional oder wenn es mehrere CACertificateFiles gibt:
SSLCACertificatePath conf/ssl.crt
SSLVerifyClient require
SSLVerifyDepth 2
</VirtualHost>
```

Mit *_b64.crt ist Ihr SSL-Zertifikat gemeint, welches Ihnen A-Trust ausgestellt hat.

Eine Client-Authentifzierung macht aber nur Sinn, wenn die Seite nur verschlüsselt erreichbar ist – denn dann kann lt. unserer Beispiel-Konfiguration wirklich nur derjenige die Homepage erreichen, der über einen privaten Schlüssel verfügt, welches das Root-Zertifikat A-trust-Qual-02a.pem besitzt. Also fügen wir noch folgende Zeilen hinzu:

```
<Directory />
SSLRequireSSL
</Directory>
```

Zur Erklärung: Ohne SSLRequireSSL kann man sich bei einer https-Verbindung zwar nur mit Zertifikat einloggen – aber die Seite ist dann auch über die gewöhnliche http-Verbindung erreichbar!

Wenn Sie nun den Apache Webserver neu starten, kann die Homepage nur noch via https und nur mittels Client-Authentifzierung betreten werden!

Wie man nur bestimmte Bereiche einer Homepage mit einer Client-Authentifzierung ausstattet, wird unter „1.5 Client Authentifzierung für Teilbereiche des Webservers“ beschrieben.

1.4 Client Authentifizierung unter der Version 2.x

Die Client-Authentifizierung unter Apache 2.x verläuft ähnlich. Nur gibt es bei der Version 2.x eine eigene SSL-Konfigurationsdatei (ssl.conf).

Auch hier gilt: Bevor Sie Änderungen an der Apache Konfiguration vornehmen, legen Sie bitte unbedingt eine Sicherheitskopie der httpd.conf- und der ssl.conf-Datei an!

Wir gehen von der SSL-Grundkonfiguration des httpd.conf Files und des ssl.conf Files in der Beschreibung

„http://www.a-trust.at/Anleitungen/client_authentification_apache_V104.pdf"

Kapitel 1.6 (Einbinden des SSL-Zertifikates unter der Version 2.x) aus.

Viele der für die Client Authentifizierung benötigten Befehle sind im ssl.conf File bereits vorgegeben und müssen nur noch aktiviert (durch Entfernen des # Symbols) und angepasst werden.

Suchen Sie sich im Virtual Host Bereich folgende Zeilen heraus:

```
SSLCACertificateFile
SSLVerifyClient
SSLVerifyDepth
```

Nun vervollständigen Sie die Befehle, z.B. so:

```
SSLCACertificateFile conf/ssl.crt/A-Trust-Qual-02a.pem
# Optional oder wenn es mehrere CACertificateFiles gibt:
SSLVerifyClient require
SSLVerifyDepth 2
```

Und nicht vergessen eine SSL-Verbindung zu erzwingen:

```
<Directory />
SSLRequireSSL
</Directory>
```

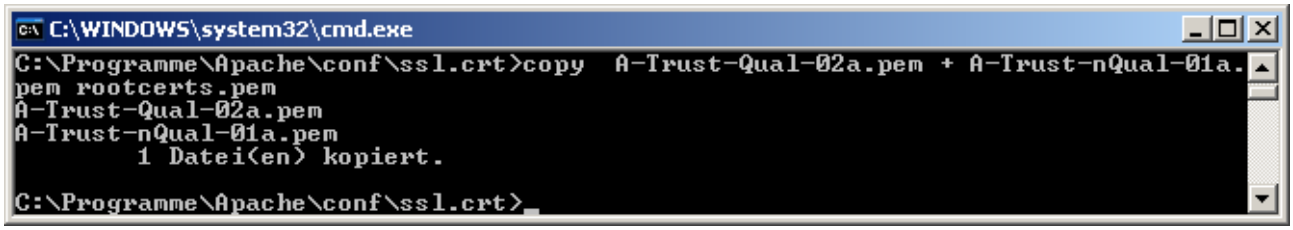
WICHTIG: Unter Apache2 funktioniert der Befehl SSLCACertificatePath nicht! Sollten Sie mehr als 1 Root-Zertifikat angeben müssen, gehen Sie bitte wie folgt vor:

In unserem Beispiel haben wir 2 Root-Zertifikate: A-Trust-Qual-02a.pem und A-Trust-nQual-01a.pem. Öffnen Sie die Eingabeaufforderung und wechseln Sie in den Pfad, in dem die Root-Zertifikate abgelegt sind. Setzen Sie folgenden Befehl ab:

```
copy  A-Trust-Qual-02a.pem + A-Trust-nQual-01a.pem rootcerts.pem
```

rootcerts.pem ist die Ausgabedatei, Sie können Sie benennen wie immer Sie wollen.

Das Ergebnis sollte in der Eingabeaufforderung wiem folgt aussehen:



Was haben wir gemacht? Es wurden beide Root-Zertifikate in ein pem-File ausgegeben – dieses File können Sie nun als SSLCACertificateFile angeben, also z.B.

```
SSLCACertificateFile conf/ssl.crt/rootcerts.pem
```

Wenn Sie nun den Apache2 Webserver neu starten, kann die Homepage nur noch via https und nur mittels Client-Authentifzierung betreten werden!

1.5 Client Authentifzierung für Teilbereiche des Webservers

Natürlich besteht auch die Möglichkeit, nur für bestimmte Bereiche einer Homepage eine Client Authentifzierung einzurichten. Wir zeigen Ihnen wie:

Bitte wieder beachten: Bevor Sie Änderungen an der Apache Konfiguration vornehmen, legen Sie bitte unbedingt eine Sicherheitskopie der httpd.conf Datei an (bei Apache 1.3.x) bzw. der httpd-conf und der ssl.conf Datei an (bei Apache 2.x)!

Such Sie sich bitte – ausgehend von unserer Beispiel-Konfiguiration – folgende Einträge im Virtual Host Bereich heraus:

```
<VirtualHost www.my-domain.com:443>
SSLEngine On
SSLCertificateFile conf/ssl/*_b64.crt
SSLCertificateKeyFile conf/ssl/my-server.key
SSLCACertificateFile conf/ssl.crt/A-Trust-Qual-02a.pem
# Optional oder wenn es mehrere CACertificateFiles gibt:
SSLCACertificatePath conf/ssl.crt
SSLVerifyClient require
SSLVerifyDepth 2
</VirtualHost>
```

Wie bereits öfters erwähnt, ist zu beachten, dass diese Einträge bei der Version 1.3.x im httpd.conf File und bei 2.x im ssl.conf File einzutragen sind.

Bitte entfernen Sie die Einträge

```
SSLVerifyClient require
SSLVerifyDepth 2
```

aus dem Virtual Host Bereich. Nun geben Sie das Directory an, das nur mittels Client Authentifzierung erreichbar sein soll. In unserem Beispiel haben wir einen Order „test" im htdocs-Verzeichnis erstellt:

```
<Directory "C:/Programme/Apache/htdocs/test">
SSLRequireSSL
SSLVerifyClient require
SSLVerifyDepth 2
</Directory>
```

Der komplette Inhalt im Ordner test kann nun nur noch mittels Login mit Zertifikat eingesehen werden.

Sie können natürlich beliebig viele Directiories in Ihrer Apache Konfiguration angeben, die eine Client Authentifzierung erfordern. Wenn Sie den Zugriff nicht auf Client-Zertifikate

mit bestimmten Root-Zertifikaten, sondern noch weiter einschränken wollen, dann sind Sie im nächsten Kapitel bestens aufgehoben.

1.6 SSLRequire – Zertifikats-Anmeldung einschränken

Es macht Sinn, nicht nur Schlüssel bestimmter Aussteller auf bestimmte Verzeichnisse zugreifen zu lassen, sondern den Zugriff auf bestimmte Zertifikatsinhalte einzuschränken. mod_ssl bietet hier mit SSLRequire eine Vielzahl an Möglichkeiten:

Wir gehen wieder von unserer Grundkonfiguration und dem Ordner „test" aus. Auf den Inhalt dieses Verzeichnisses soll nur eine bestimmte Person Zugriff erhalten, z.B. Hans Meier. Die Befehlszeile dazu sieht wie folgt aus:

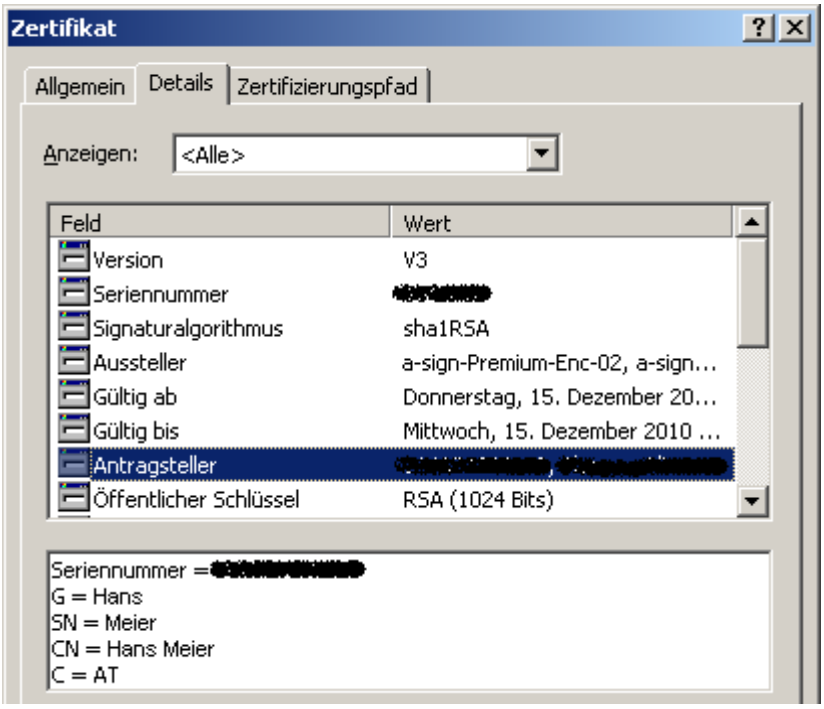
```
SSLRequire  %{SSL_CLIENT_S_DN_CN} eq "Hans Meier"
```

Mit diesem Befehl verlangen wir, dass nur ein Zertifikat, lautend auf den Common Name „Hans Meier", Zugriff erhält. Da wir in unserer Standard-Konfiguration bereits angegeben hatten, dass nur Zertifikate mit dem Root-Zertifikat A-Trust-Qual-02a Zugriff erhalten, haben wir bereits eine sehr genaue Einschränkung.

So sieht also unsere Directory-Konfiguration aus:

```
<Directory "C:/Programme/Apache/htdocs/test">
SSLRequireSSL
SSLVerifyClient require
SSLVerifyDepth 2
SSLRequire  %{SSL_CLIENT_S_DN_CN} eq "Hans Meier"
</Directory>
```

Um sich das bildlich vor Augen zu halten, sehen Sie in der folgenden Abbildung den öffentlichen Auszug des Zertifikates, lautend auf Hans Meier:



Wir stellen also fest, dass wir mit

```
SSLRequire %{SSL_CLIENT_S_DN_CN} eq "Hans Meier"
```

auf den Common Name im Feld Antragsteller zugreifen.

An dieser Stelle sei gesagt, dass Sie natürlich bei jeder Änderung Ihren Apache Webserver neu starten müssen, damit die neuen Einstellungen übernommen werden.

Es gibt natürlich auch die Möglichkeit, mehr als nur eine Person zu berechtigen. Wenn Sie z.B. nicht nur den Hans Meier, sondern auch den Paul Panther berechtigen wollen, tragen Sie folgende Zeile ein:

```
SSLRequire %{SSL_CLIENT_S_DN_CN} in {"Hans Meier", "Paul Panther"}
```

Sie können beliebig viele Einträge innerhalb der geschwungenen Klammern vornehmen.

Beachten Sie bitte, dass Sie nicht in einem Directory den gleichen Befehl 2 Mal angeben können – es wird immer nur die erste Zeile ausgelesen. Z.B.:

```
SSLRequire %{SSL_CLIENT_S_DN_CN} eq "Hans Meier"
SSLRequire %{SSL_CLIENT_S_DN_CN} eq "Paul Panther"
```

In diesem Fall wird nur Hans Meier berechtigt, apache ignoriert die 2.Zeile. Es ist aber natürlich kein Problem, verschiedene Befehle in ein Directory zu schreiben.

Es gibt aber noch viele Variablen, die wir aus dem Zertifikat auslesen und mit SSLRequire prüfen können. Wir haben für Sie eine Liste erstellt (mit Erklärung um welche Variable es sich handelt) - Sie können selbst entscheiden, welche Berechtigungs-Eingrenzungen für Sie mehr oder weniger von Vorteil sind:

```
# Auslesen des Given Names (Vorname)
SSLRequire  %{SSL_CLIENT_S_DN_G} eq "Hans"
```

```
# Auslesen des Surnames (Nachname)
SSLRequire  %{SSL_CLIENT_S_DN_S} eq "Meier"
# Auslesen des Common Names (Vor- und Nachname)
SSLRequire  %{SSL_CLIENT_S_DN_CN} eq "Hans Meier"
# Auslesen des Country-Codes des Zertifikats-Inhabers
SSLRequire  %{SSL_CLIENT_S_DN_C} eq "AT"
# Auslesen der Hex-Seriennummer des Zertifikates. Dies ist die genaueste Eingrenzung, da eine Seriennummer nur ein einziges Mal pro Aussteller vergeben wird
SSLRequire  %{SSL_CLIENT_M_SERIAL} eq "018F8A"
# Auslesen des Common Names des Ausstellers
SSLRequire  %{SSL_CLIENT_I_DN_CN} eq "a-sign-Premium-Enc-02"
# Auslesen der Organisation Unit des Ausstellers
SSLRequire  %{SSL_CLIENT_I_DN_OU} eq "a-sign-Premium-Enc-02"
# Auslesen der Organisation des Ausstellers
SSLRequire  %{SSL_CLIENT_I_DN_O} eq "A-Trust Ges. f. Sicherheitssysteme im elektr. Datenverkehr GmbH"
# Auslesen des Country-Codes des Ausstellers
SSLRequire  %{SSL_CLIENT_I_DN_C} eq "AT"
# Wenn z.B. mehr als ein Common Name Zugriff erhalten soll.
SSLRequire %{SSL_CLIENT_S_DN_G} in {"Hans Meier", "Paul Panther"}
```

Falls Sie die Möglichkeit nutzen wollen, einen SSL-geschützten Bereich zusätzlich zeitlich einzuschränken, bietet sich folgende SSLRequire Variante an:

```
SSLRequire ( %{TIME_WDAY} >= 1 and %{TIME_WDAY} <= 5 \
and %{TIME_HOUR} >= 8 and %{TIME_HOUR} <= 17 ) \
or %{REMOTE_ADDR} =~ m/^192\.168\.1\.[0-9]+$/
```

In diesem Beispiel ist ein Seitenzugriff nur von Montag (>=1) bis Freitag (<=5), zwischen 8 und 17 Uhr erfolgen – ODER der Zugriff erfolgt im Intranet mit einer IP-Adresse 192.168.1.xxx (xxx steht hier für eine beliebige Zahl).

Sie sehen also, dass hier beinahe keine Grenzen gesetzt sind. Sie können die Befehle so miteinander verbinden, dass es genau Ihren Ansprüchen angepasst wird.

1.7 Zertfifikats-Anmeldung mit php einschränken

Wenn Sie php auf Ihrem Apache Webserver installiert haben, können Sie auch über PHP-Skripte (und Datenbanken) den Zugriff steuern. Dies lohnt sich vor allem bei der Administration von vielen Benutzern, die Sie sonst mühsam einzeln in der Apache SSL-Konfiguration berechtigen müssten. Ein php-Beispielskript finden Sie im Kapitel 1.11

1.8 Auflistung der A-Trust Root-Zertifikate (SSLCACertificateFile)

Sie können alle A-Trust Stammzertifikate auf der A-Trust Homepage herunterladen:

https://www.a-trust.at/info.asp?node=207&skip_offline=1

Beachten Sie bitte, dass Apache die Zertifikate im pem-Format benötigt, um damit umgehen zu können.

Da auch Stammzertifikate irgendwann ablaufen, kann es vorkommen, dass ein Produkt (z.B. a.sign premium) je nach Datum der Aktivierung verschiedene Stammzertifikate hat. Anbei eine Auflistung, welche Stammzertifikate Sie für welches Benutzer-Zertifikat benötigen:

Bankomatkarte a.sign premium:	A-Trust-Qual-02a.pem
Mastercard a.sign premium:	A-Trust-Qual-02a.pem
a.sign premium Karte (Enc-01):	A-Trust-nQual-01a.pem
a.sign premium Karte (Enc-02):	A-Trust-Qual-02a.pem
a.sign token Karte (Enc-01):	A-Trust-nQual-01a.pem
a.sign token Karte (Enc-02):	A-Trust-Qual-02a.pem
a-sign light (light-01):	A-Trust-nQual-01a.pem
a-sign light (light-02):	A-Trust-Qual-02a.pem
WU Studentenausweis (Enc-01):	A-Trust-nQual-01a.pem
WU Studentenausweis (Enc-02):	A-Trust-Qual-02a.pem
trust sign Karte:	A-Trust-nQual-01a.pem

1.9 Informationen zu den SSL-Befehlen

Bei den unter Punkt 1.3 bis 1.6 angegebenen SSL-Befehlen handelt es sich um Standardwerte, die errolgreich getestet wurden. Sie müssen nicht zwingend 1:1 in Ihre Apache Konfiguration übernommen werden.

Die genauen Definitionen der einzelnen Befehle können Sie auf http://www.modssl.org in der Online-Dokumentation nachschlagen.

1.10 Anregungen, Beschwerden, etc.

Diese Anleitung wurde speziell für unsere Kunden geschrieben, um Ihnen den Umgang mit Zertifikaten in Zusammenhang mit den Apache Webservern Version 1.3.x und 2.x näher zu bringen. Auch wir sind nicht perfekt. Und daher bitten wir Sie, uns fehlerhafte Angaben zu verzeihen und uns diese zu melden. Gemeinsam sind wir stark.

1.11 PHP Beispiel

```
<?php

  $subject =  ($_SERVER['CERT_SUBJECT']);
  if (isset ($subject))
  {
    if (eregi ("OID.2.5.4.5=([0-9]*)",$subject, $regs)) //Ausgabe der Cin für IIS
    {
      $cin=$regs[1]; print "$cin<br>";
     }
     if (eregi ("SN=([^,]*)",$subject, $regs)) //Abfrage des Nachnamens im IIS
     {
        $sn=$regs[1]; print "$sn<br>";
     }
     if (eregi ("G=([^,]*)",$subject, $regs)) //Abfrage des Vornamens im IIS
     {
        $g=$regs[1]; print "$g<br>";
     }
  }

  $subject2 = ($_SERVER['SSL_CLIENT_S_DN']);
  if (isset ($subject2))
  {
    print "<br>$subject2<br>";
    if (eregi ("Number=([0-9]*)",$subject2, $regs))
    {
      $cin2=$regs[1]; print "cin=$cin2<br>";
    }
    if (eregi ("SN=([^,/]*)",$subject2, $regs))
    {
      $sn=$regs[1]; print "$sn<br>";
    }
    if (eregi ("GN=([^,/]*)",$subject2, $regs))
    {
      $gn=$regs[1]; print "$gn<br>";
    }
 }


if ($cin == "760675377286") //oder Datenbankabfrage
{ print "welcome!";} //Redirect auf eine andere Site

elseif( $cin2 == "760675377286") //  CIN2 für den Apache
{  print "welcome!";} //Redirect auf eine andere Site
else
{ print "<h1>Nicht berechtigt!</H1>"; }

?>
```