1. Webserver Apache (Version 2.4).

Bis Fedora 17 wurde die Apache Version 2.2, ab Fedora 18 wird die Version 2.4 verwendet. Damit ändern sich etliche Direktiven in den Konfigurationsdateien, etliche bleiben aber auch gleich.

- *Der Webserver ist über **Port 80 (HTTP)** oder **Port 443 (HTTPS)** erreichbar.*

- Das **RPM Paket httpd** muss installiert sein.

- Der **Dienst httpd** muss gestartet werden ...

```
# service httpd start
Redirecting to /bin/systemctl start httpd.service

# service httpd status
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
     Active: active (running) since Fri 2020-09-11 20:17:10 CEST; 6s ago
       Docs: man:httpd.service(8)
   Main PID: 11271 (httpd)
     Status: "Started, listening on: port 443, port 80"
      Tasks: 213 (limit: 4580)
     Memory: 16.6M
        CPU: 57ms
     CGroup: /system.slice/httpd.service
             ├─11271 /usr/sbin/httpd -DFOREGROUND
             ├─11272 /usr/sbin/httpd -DFOREGROUND
             ├─11273 /usr/sbin/httpd -DFOREGROUND
             ├─11274 /usr/sbin/httpd -DFOREGROUND
             └─11275 /usr/sbin/httpd -DFOREGROUND

Sep 11 20:17:10 orion.gate.local systemd[1]: Starting The Apache HTTP Server...
Sep 11 20:17:10 orion.gate.local httpd[11271]: Server configured, listening on: port 80
Sep 11 20:17:10 orion.gate.local systemd[1]: Started The Apache HTTP Server.
```

... und zum Start beim Booten eingerichtet werden:

```
# systemctl enable httpd.service
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service
              → /usr/lib/systemd/system/httpd.service.
```

  - Weitere nützliche RPM Pakete: **mod_ssl** (SSL Verschlüsselung für HTTPS Zugriffe, nach Installation Dienst httpd restarten), **httpd-manual** (Apache Manual, unter der eigenen URL des installierten Servers erreichbar, http://127.0.0.1/manual).

1.1 Webseiten im Verzeichnis /var/www/html und darin angelegten Unterverzeichnisse.

Im Verzeichnis **/var/www/html** und **darin** neu **angelegten Unterverzeichnissen** können Webseiten zur Auslieferung angelegt werden, ohne dass man sich mit SELinux beschäftigen muss.

```
# mkdir /var/www/html/test
```

Nachdem eine einfache index.html Datei im neu erstellten Unterordner test erstellt wurde, wird diese auch vom Webserver ausgeliefert (Kommando **curl** kann zum Testen an der Konsole benutzt werden).

```
# ls -l /var/www/html/test
insgesamt 4
-rw-r--r--. 1 root root 41  8. Sep 17:46 index.html

# curl http://127.0.0.1/test/
<html>
      Dies ist eine Testseite.
</html>
```

Um Benutzern und Benutzerinnen das Anlegen von Webseiten zu gestatten, legt man zum Beispiel für die Benutzerin corinna ein Unterverzeichnis an und erteilt ihr Rechte an diesem Ordner.

```
# mkdir /var/www/html/corinna

# chown corinna:corinna /var/www/html/corinna

# ls -l /var/www/html/
drwxr-xr-x. 2 corinna corinna 4096  8. Sep 18:10 corinna
```

Nachdem die Userin zum Beispiel dort eine einfache Testdatei index.html erstellt hat, kann diese auch ausgeliefert werden.

```
# ls -l /var/www/html/corinna
-rw-rw-r--. 1 corinna corinna 36  8. Sep 18:59 index.html

# curl http://127.0.0.1/corinna/
<html>
      Corinna's Testpage.
</html>
```

1.2 Webseiten in anderen Verzeichnissen als /var/www/html.

Sollen **andere Verzeichnisse als /var/www/html mit Unterverzeichnissen für die Auslieferung von Webseiten** genutzt werden, so sind diese Verzeichnisse und evtl. schon vorhandene Dateien und tiefergehende Unterverzeichnisse rekursiv mit dem Sicherheitssystem SELinux zu bearbeiten. Dazu sind die **Kommandos semanage**, **restorecon** und setsebool zu benutzsen. Wichtig, es müssen **reguläre Ausdrücke beim Kommando semanage** verwendet werden, um rekursiv zu arbeiten!

- Für die **SELinux Sicherheitseinrichtung** lesen Sie die **Manpage httpd_selinux** (installieren Sie das RPM Paket selinux-policy-doc).

Zuerst müssen wieder Verzeichnisse bzw. Unterverzeichnisse angelegt werden und den Benutzern Rechte an diesen Unterordnern gegeben werden. Anlegen eines Userwebspace zum Beispiel in /usr/local/webspace mit seinen Unterverzeichnissen für die verschiedenen Benutzer:

```
# mkdir /usr/local/webspace
# mkdir /usr/local/webspace/anna
# mkdir /usr/local/webspace/bernd
     usw.

# chown anna:anna /usr/local/webspace/anna
# chown bernd:bernd /usr/local/webspace/bernd
     usw.

# ls -l /usr/local/webspace
drwxr-xr-x. 2 anna  anna  4096  6. Sep 17:56 anna
drwxr-xr-x. 2 bernd bernd 4096  6. Sep 17:56 bernd
     usw.
```

In /var/www/html sind symbolische Links auf die Userverzeichnisse in /usr/local/webspace anzulegen.

```
# ln -s /usr/local/webspace/anna/ /var/www/html/anna
# ln -s /usr/local/webspace/bernd/ /var/www/html/bernd
     usw.
# ls -l /var/www/html
lrwxrwxrwx. 1 root root 25  6. Sep 19:12 anna -> /usr/local/webspace/anna/
lrwxrwxrwx. 1 root root 26  6. Sep 19:12 bernd -> /usr/local/webspace/bernd/
     usw.
```

1.2.1 Beispiel 1, SELinux und Benutzerwebspace.

In diesem Falle sollen Dateien nur über den Webserver ausgeliefert werden. Dazu muss diese Verzeichnisstruktur als **httpd_read_user_content** gelabelt und der **Schalter httpd_enable_homedirs** aktiviert werden. Benutzern gehörende Dateien in den entsprechenden Verzeichnissen werden vom httpd Server ausgeliefert, Scripte wie PHP und Perl einwandfrei ausgeführt.

```
# ls -Z /usr/local/webspace/
unconfined_u:object_r:usr_t:s0 anna  unconfined_u:object_r:usr_t:s0 bernd
     usw.

# semanage fcontext -a -t httpd_user_content_t "/usr/local/webspace(/.*)?"
# restorecon -R /usr/local/webspace

# ls -Z /usr/local/webspace/
unconfined_u:object_r:httpd_user_content_t:s0 anna
unconfined_u:object_r:httpd_user_content_t:s0 bernd
     usw.

# getsebool httpd_enable_homedirs
httpd_enable_homedirs --> off
# setsebool -P httpd_enable_homedirs on
# getsebool httpd_enable_homedirs
httpd_enable_homedirs --> on
```

Nachdem Userin anna eine einfache index.html Datei in /usr/local/webspace/anna/ abgelegt hat wird diese vom Webserver ausgeliefert:

```
# curl http://127.0.0.1/anna/
<html>
      Anna is here!
</html>
```

Wollen Sie die SELinux Einstellungen wieder rückgängig machen, weil Sie den Webspace für die Benutzer nicht mehr zur Verfügung stellen wollen, so gehen Sie wie folgt vor:

```
# setsebool -P httpd_enable_homedirs off
# getsebool httpd_enable_homedirs
httpd_enable_homedirs --> off

# restorecon -F -R /usr/local/webspace
# ls -Z /usr/local/webspace/
system_u:object_r:usr_t:s0 anna  system_u:object_r:usr_t:s0 bernd
     usw.
```

1.2.2 Beispiel 2, SELinux und öffentlicher Inhalt:

Soll der Inhalt von Verzeichnissen nicht nur allein über den Webserver, sondern auch anderweitig wie FTP, SMB etc. ausgeliefert werden, so ist die Verzeichnisstruktur als **public_content_t** zu labeln. Ein Schalter ist in diesem Fall nicht zu aktivieren. Benutzern gehörende Dateien in den entsprechenden Verzeichnissen werden vom httpd Server ausgeliefert, Scripte wie PHP und Perl einwandfrei ausgeführt.

```
# ls -Z /usr/local/webspace/
system_u:object_r:usr_t:s0 anna  system_u:object_r:usr_t:s0 bernd
     usw.

# semanage fcontext -a -t public_content_t "/usr/local/webspace(/.*)?"
# restorecon -R /usr/local/webspace

# ls -Z /usr/local/webspace/
system_u:object_r:public_content_t:s0 anna  system_u:object_r:public_content_t:s0 bernd
```

Nachdem Userin anna eine einfache index.html Datei in /usr/local/webspace/anna/ abgelegt hat wird diese vom Webserver ausgeliefert:

```
# curl http://127.0.0.1/anna/
<html>
      Anna is here!
</html>
```

Wollen Sie die SELinux Einstellungen wieder rückgängig machen, weil Sie den Webspace für die Benutzer nicht mehr zur Verfügung stellen wollen, so gehen Sie wie folgt vor:

```
# restorecon -F -R /usr/local/webspace
# ls -Z /usr/local/webspace/
system_u:object_r:usr_t:s0 anna  system_u:object_r:usr_t:s0 bernd
     usw.
```

Wollen Sie die SELinux Einstellungen wieder rückgängig machen, weil Sie den Webspace für die Benutzer nicht mehr zur Verfügung stellen wollen, so gehen Sie wie folgt vor:

```
# semanage fcontext -d -t public_content_t "/usr/local/webspace(/.*)?"
# restorecon -R /usr/local/webspace

# ls -Z /usr/local/webspace/
system_u:object_r:usr_t:s0 anna  system_u:object_r:usr_t:s0 bernd
     usw.
```

1.3 Grundkonfiguration Apache.

- Die **Webdateien** (und weitere **Unterverzeichnisse** mit Webdateien) sind im **Verzeichnis /var/www/html** abzulegen (befindet sich in /var/www/html eine index.html Datei, erscheint nicht mehr bei der Server URL im Browser der Clients die Fedora Testpage).

  - Bei **Webdateien in Unterverzeichnissen** von /var/www/html lautet die URL Adresse in den Browsern des Clients http://server-FQDN/Verzeichnisname/webdatei.

  - Soll nicht nur **index.html**, sondern auch andere Webdateien ohne deren explizite Angabe der URL im Browser der Clients angezeigt werden, so sind diese weiteren Dateinamen unter dem **Eintrag DirectoryIndex** in der **Konfigurationsdatei /etc/httpd/conf/httpd.conf** aufzuführen. *Beispiel, durch den zusätzlichen Eintrag index.htm (die beiden Einträge index.html und index.html.var sind schon vorhanden), werden auch index.htm Dateien ohne Nennung in der URL angezeigt:*

    ```
    ...
    # DirectoryIndex: sets the file that Apache will serve if a directory
    DirectoryIndex index.html index.htm index.html.var
    ...
    ```

- Damit der Server das **Meta-Tag für den Characterset in einer Webseite** nicht ignoriert, er liefert sonst immer in UTF-8 aus, auch wenn ein Meta-Tag in einer Webseite einen anderen Characterset angibt (es kommt dann dazu, das im Browser des Clients z.B. Umlaute falsch dargestellt werden) **kommentieren Sie in der Konfigurationsdatei die Zeile AddDefaultCharset UTF-8**:

  ```
  ...
  # AddDefaultCharset UTF-8
  ...
  ```

- Konfigurieren Sie die **Sicherheitseinstellungen von Verzeichnissen**. Die **grundlegende Einstellung** findet sich im Eintrag der **Konfigurationsdatei /etc/httpd/conf/httpd.conf** und gilt für alle Unterverzeichnisse des Wurzelverzeichnis des Servers, solange für diese keine anderen Einstellungen vorgenommen werden und sollte entsprechen restriktiv sein. Sie müssen natürlich darauf achten, dass in Directory Eintragungen für untergeordnete Verzeichnisse Ihre restriktiven Festlegungen nicht wieder außer Kraft gesetzt werden:

  ```
  <Directory />
      AllowOverride none
      Require all denied
  </Directory>
  ```

Die Direktiven bedeuten:

- **AllowOverride none**
  Die Direktive legt fest, ob User in .htaccess Dateien Änderungen der Konfiguration vornehmen dürfen. Die Angabe none bedeutet, dass gar keine Änderungen vorgenommen werden dürfen.
- **Require all denied**
  Diese Direktive legt fest, wer Zugriff auf die Dateien hat. Das Schlüsselwort **all** gilt für alle, **denied** verbietet Zugriffe (im Gegensatz dazu würde **Require all granted** allen den Zugriff erlauben).
  Die Require Direktive ersetzt die Direktiven Order, Allow und Deny der Version 2.2

- Konfiguration des Webspace:

```
<Directory "/var/www">
    AllowOverride None
    # Zum Beispiel nur Intranet:
    Require ip 127.0.0.1 192.168.185.0/255.255.255.0
</Directory>
```

Die Direktiven:

- **Require ip**
  Clients haben nur Zugriff, wenn Sie in der Liste der IP Nummern bzw. Netzwerken aufgeführt sind.
- **Require host**
  Clients haben nur Zugriff, wenn Sie in der Liste der FQDN bzw. Domainen aufgeführt sind.

```
# Links zu den Benutzerverzeichnissen im Dateisystem ermoeglichen, kein Fancy Indexing
# (Anzeige Inhaltsverzeichnis, wenn keine index.html Datei im Verzeichnis) erlauben,
# Programmausführung verbieten:
<Directory "/var/www/html">
   Options -Indexes +FollowSymLinks +IncludesNoExec
   AllowOverride None
   Require ip 127.0.0.1 192.168.185.0/255.255.255.0
 </Directory>
```

```
# Benutzer duerfen keine Programme ausfuehren, aber in eigenen .htaccess Dateien
# kann User/Passwort Identifikation und Indexing genutzt werden (s. auch SSLRequireSSL).
# FollowSymLinks nicht erlauben, da sonst auch Links auf
# Dateien in /etc usw. von Benutzern gesetzt werden können!
<Directory "/var/www/html/*">
   Options -FollowSymLinks
   AllowOverride AuthConfig Options=Indexes
 </Directory>
```

- Zum Verwalten von Passwort geschützten Zugriffen dient das **Kommando htpasswd**.

```
# Nur In den cgi-bin Verzeichnissen (rwxr-xr-x) der Benutzer Ausfuehrung von Programmen
# erlauben, das erleichtert dem Administrator die Kontrolle.
<Directory "/var/www/html/*/cgi-bin">
   SSLRequireSSL
   Options ExecCGI
   SetHandler cgi-script
</Directory>
```

- **SSLRequireSSL**
  Lässt nur HTTPS Verbindungen zu, unverschlüsselte HTTP Verbindungen werden abgelehnt. Sollte auf jeden Fall in Zusammenhang mit Passwort geschützten Webverzeichnissen genutzt werden (kann auch in .htaccess Dateien vom User benutzt werden).

- Um **PHP** auf dem Webserver zu ermöglichen, ist die Installation der **Pakete php und php-common** nötig. Beispiel Testpage hallo.php:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-15">
<meta name="author" content="Dr. J. Kubiak">
<meta name="date" content="2010-11-03T15:49:13">
<title>Hallo Welt</title>
</head>

<body>
<h1>PHP Code - Hallo Welt.</h1>

<p style="background: yellow; ">
<?php
echo "Hallo Welt!";
?>
</p>

</body>
</html>
```

- **Perl** ist standardmäßig auf Linux Rechnern installiert, damit können ohne weitere Paketinstallationen in Perl geschriebene dynamische Webseiten erstellt werden.