Wichtige Konsolenbefehle

Allgemeine Hinweise

> Die meisten Befehle können mit `Tab` automatisch vervollständigt werden.

> Falls mehr Informationen zu einem Befehl benötigt werden, kann der Befehl idR. mit dem Argument `--help` aufgerufen werden, um eine kurze Beschreibung zu erhalten. Alternativ kann auch in der jeweiligen Dokumentation nachgelesen werden.

Wichtige Befehle

Hier ist eine kurze Übersicht der wichtigsten Befehle, die für die Navigation der Konsole benötigt werden:

Verzeichnis wechseln mit *cd*

Beschreibung
Mit dem Befehl `cd` kann das aktuelle Verzeichnis gewechselt werden.

Verwendung
`cd [directory]`

- `directory` gibt hier den Pfad zum neuen Verzeichnis an. Dieser Pfad darf relativ oder absolut sein.

Beispiel

- `cd ..` wechselt in das überliegende Verzeichnis
- `cd /SmartDog/` wechselt in das Verzeichnis `/SmartDog/`

Dokumentation

- Dokumentation

Elemente in Verzeichnis anzeigen mit *ls*

Beschreibung

Mit dem Befehl `ls` können Elemente in einem Verzeichnis und deren Eigenschaften angezeigt werden.

Verwendung

`ls [options] [file/directory]`

- Mit `options` kann die Ausgabe des Befehls verändert werden
  - `-a` zeigt alle Elemente an, auch solche, die normalerweise ausgeblendet werden
  - `-l` zeigt Elemente in einer alphabetisch geordneten Liste an
- `file/directory` gibt an, worauf sich der Befehl beziehen soll
  - Standardmäßig wird das aktuelle Verzeichnis verwendet
  - Wird hier ein Verzeichnis angegeben, so kann dessen Inhalt angezeigt werden, ohne vorher in dieses zu wechseln

Beispiel

- `ls -l` listet die Elemente im aktuellen Verzeichnis auf
- `ls -l /SmartDog/` listet die Elemente im Verzeichnis `/SmartDog/` auf, unabhängig vom aktuellen Verzeichnis

Dokumentation

- Dokumentation

Elemente kopieren mit *cp/scp*

Beschreibung

Mit dem Befehl `cp` können Elemente kopiert werden. Sollen Elemente von einem Gerät zu einem anderen kopiert werden, so muss der Befehl `scp` verwendet werden.

Verwendung

`cp [source] [destination]`

- `source` gibt den Pfad des Elements an, das kopiert werden soll
- `destination` gibt den Pfad der Kopie an, die erstellt werden soll
- Sollte `destination` bereits existieren, so wird dies ohne Nachfrage überschrieben

`scp [source] [destination]`

- Analog zu `cp`, allerdings kann sich ein Pfad auf einem anderen Gerät befinden
- Geräteübergreifende Pfade haben folgende Signatur: `Benutzername@IP/Hostname:/Pfad_auf_anderem_Gerät`

Beispiel

- `cp SmartDog SmartDog_old` kopiert die Datei `SmartDog` zu `SmartDog_old`
- `scp SmartDog root@192.168.178.xxx:/SmartDog/SmartDogN` kopiert die Datei `SmartDog` auf das Gerät mit der IP-Adresse `192.168.178.xxx` und werdendet den Benutzernamen `root`
  - Hier muss anschließend noch das zum Benutzernamen gehörende Passwort eingegeben werden
  - Statt der IP-Adresse kann auch der Hostname des Geräts verwendet werden

Dokumentation

- Dokumentation `cp`
- Dokumentation `scp`

Netzwerkeinstellungen anzeigen mit *ipconfig*

Beschreibung

Mit dem Befehl `ipconfig` können die aktuellen Netzwerkeinstellungen angezeigt werden.

Verwendung

`ifconfig [interface]`

- `interface` gibt eine Netzwerkschnittstelle an, die betrachtet werden soll
  - Standardmäßig werden die Einstellungen für alle Schnittstellen angezeigt

Beispiel

- `ifconfig` zeigt Eisntellungen für alle Netzwerkschnittstellen an
- `ifconfig eth0` zeigt nur die Einstellungen für die Schnittstelle `eth0` an

Dokumentation

- Dokumentation

Netzwerkkommunikation überprüfen mit *ping*

Beschreibung

Mit dem Befehl `ping` kann ein anderes Netzwerkgerät *angepingt* werden.

Verwendung

`ping [ip/hostname]`

- `ip/hostname` gibt hier die IP-Adresse bzw. den Hostname des Ziel-Gerätes an
- Wenn das Gerät über Netzwerk erreichbar ist, so wird für jedes gesendete Paket die Antwortzeit angezeigt
- Sollte das Gerät nicht erreichbar sein, so wird entweder eine Fehlermeldung angezeigt oder man erhält nie eine Antwort und es wird nichts angezeigt
- Das Ziel-Gerät wird durchgehend angepingt, bis der Befehl abgebrochen wird (`Strg + C`)
- In seltenen Fällen ist es möglich, dass Geräte nicht auf Pings antworten, obwohl diese im Netzwerk erreichbar sind. In diesem Fall sollte zusätzlich geprüft werden, ob der benötigte Port geöffnet ist

Beispiel

- `ping 192.168.178.xxx` pingt das Gerät mit der IP-Adresse `192.168.178.xxx` an

Dokumentation

- Dokumentation

Port überprüfen mit *netcat* bzw. *echo*

> *netcat* ist nur auf dem SmartDog verfügbar
>
> Auf dem PowerDog muss der Port mit *echo* überprüft werden

SmartDog

Beschreibung

Mit dem Befehl `netcat` bzw. `nc` kann geprüft werden, ob ein bestimmter TCP- oder UDP-Port geöffnet ist.

Verwendung

`netcat [options] [ip/hostname] [port]` `nc [options] [ip/hostname] [port]`

- `options` gibt an, welche Operationen durchgeführt werden sollen
  - `-v` zeigt eine ausführlichere Ausgabe an
  - `-z` prüft nur, ob das Gerät unter angegebener IP-Adresse und Port erreichbar ist
  - `-u` legt fest, dass *UDP* statt *TCP* für die Verbindung verwendet werden soll
- `ip/hostname` gibt hier die IP-Adresse bzw. den Hostname des Ziel-Gerätes an
- `port` gibt den zu testenden Port an

Beispiel

- `netcat -v -z 192.168.178.xxx 502` prüft, ob der TCP-Port `502` auf dem Gerät mit der IP-Adresse `192.168.178.xxx` geöffnet ist
  - Ist der Port geöffnet, wird eine Erfolgsmeldung ausgegeben
  - Ist dies nicht der Fall, so wird entweder eine Fehlermeldung ausgegeben oder es wird nichts angezeigt
  - Sollte länger nichts angezeigt werden, kann der Befehl abgebrochen werden (`Strg` + `C`)
- `netcat -v -z -u 192.168.178.xxx 502` prüft, ob der UDP-Port `502` auf dem Gerät mit der IP-Adresse `192.168.178.xxx` geöffnet ist
  - ACHTUNG: Dies funktioniert nicht immer, da UDP hierfür nicht gedacht ist! D.h. hier kann angezeigt werden, dass der Port geöffnet ist, obwohl dies eigentlich nicht der Fall ist!

Dokumentation

- Dokumentation

PowerDog

Beschreibung

Mit dem Befehl `echo` kann geprüft werden, ob ein bestimmter TCP-Port geöffnet ist.

Verwendung

`echo > /dev/tcp/[ip/hostname]/[port] && echo "Port is open"`

- `ip/hostname` gibt hier die IP-Adresse bzw. den Hostname des Ziel-Gerätes an
- `port` gibt den zu testenden Port an

Beispiel

- `echo > /dev/tcp/192.168.178.xxx/502 && echo "Port is open"` prüft, ob der TCP-Port `502` auf dem Gerät mit der IP-Adresse `192.168.178.xxx` geöffnet ist
  - Ist der Port geöffnet, wird die Meldung `Port is open` ausgegeben
  - Ist dies nicht der Fall, so wird nichts angezeigt
  - Sollte länger nichts angezeigt werden, kann der Befehl abgebrochen werden (`Strg + C`)

Speicherplatz überprüfen mit *df/du*

Beschreibung

Mit dem Befehl `df` kann der verwendete Speicherplatz von Verzeichnissen und mit `du` der Speicherplatz von Dateien überprüft werden.

Verwendung

`df [options] [directory]` `du [options] [directory]`

- Mit `options` kann die Ausgabe des Befehls verändert werden
  - `-h` gibt Speichergrößen in Byte statt 1K-Blöcken an
- `directory` gibt das Verzeichnis an, das überprüft werden soll

Beispiel

- `df -h` gibt den Speicherverbrauch auf dem Gerät an
- `df -h /SmartDog/` gibt an, wie viel Speicher von dem Verzeichnis `/SmartDog/` benötigt wird
- `du -h /SmartDog/` listet die Dateigrößen von allen Dateien im Verzeichnis `/SmartDog/` und dessen Unterverzeichnissen auf
- `du -h SmartDog` gibt den verwendeten Speicherplatz der Datei `SmartDog` im aktuellen Verzeichnis an

Dokumentation

- Dokumentation `df`
- Dokumentation `du` {.links-list}

Netzwerk-Daten senden mit *curl*

Beschreibung

Mit dem Befehl `curl` können Daten über Netzwerk an einen Server gesendet und bzw. davon empfangen werden. Hiermit können einfach *HTTP-GET* und *HTTP-POST* Anfragen getestet werden.

Verwendung

`curl [options] [URL]`

- `-k` überprüft Zertifikat des Servers nicht

- `-X` gibt den Typ der Anfrage an
- `-H` legt den HTTP-Header fest
- `-d` gibt die Daten an, die in einer POST-Anfrage übertragen werden
- `-c` speichert Cookies in eine Datei
- `-b` lädt Cookies aus einer Datei

Beispiel

- `curl -k https://ipinfo.io/ip` gibt die öffentliche IP-Adresse aus
- `curl -k -c cookies.txt -X POST https://192.168.178.xxx/api/login -H "Content-Type: application/json" -d '{"a" : "b"}'` sendet eine HTTP-POST Anfrage an die URL `https://192.168.178.xxx/api/login` mit dem Inhalt `{"a" : "b"}` und speichert eventuell empfangene cookies in die Datei `cookies.txt`
- `curl -k -b cookies.txt https://192.168.178.xxx/api/data` sendet eine HTTP-GET Anfrage an die URL `https://192.168.178.xxx/api/data`, wobei die in `cookies.txt` gespeicherten Cookies verwendet werden

Dokumentation

- Dokumentation

Prozesse und Threads auflisten mit *ps*

Beschreibung

Mit dem Befehl `ps` können alle auf dem Gerät aktiven Prozesse augelistet werden.

Verwendung

`ps [options]`

- Mit `options` kann die Ausgabe des Befehls verändert werden
  - `x` wird benötigt, damit alle Prozesse angezeigt werden
  - `a` zeigt auch Prozesse, die von einem anderen Benutzer gestartet wurden

    - `-T` zeigt Threads an (inklusive Name, falls festgelegt)
    - `-p` legt eine ProzessID fest, auf die der Befehl angewandt werden soll

Beispiel

- `ps xa` gibt alle aktuell ausgeführten Prozesse an
- `ps -T -p PID` listet alle Threads des Prozesses mit der ID `PID`
- `ps -o nlwp PID` gibt die Anzahl der aktiven Threads des Prozesses mit der ID `PID` an

Dokumentation

- Dokumentation

Ordner erstellen mit *mkdir*

Beschreibung

Mit dem Befehl `mkdir` können Ordner erstellt werden.

Verwendung

`mkdir [directory]`

- `directory` gibt den Ordnerpfad an, der erstellt werden soll

Beispiel

- `mkdir test` legt einen Ordner mit dem Namen `test` im aktuellen Verzeichnis an

Dokumentation

- Dokumentation

Dateien und Verzeichnisse löschen mit *rm*

Beschreibung

Mit dem Befehl `rm` können Dateien und Verzeichnisse gelöscht werden.

Verwendung

`rm [options] [file/directory]`

- Mit `options` kann das Verhalten beim Löschen festgelegt werden
  - `-f` erzwingt das Löschen von geschützten Dateien
  - `-R` löscht Dateien rekursiv (hiermit können Verzeichnisse gelöscht werden)

Beispiel

- `rm testFile` löscht die Datei `testFile`
- `rm -R testFolder/` löscht den Ordner `testFolder` und dessen Inhalt

Dokumentation

- Dokumentation

Archive entpacken mit *tar*

Beschreibung

Mit dem Befehl `tar` können Archive entpackt werden.

Verwendung

`tar [options 1] [source] [options 2]

- Mit `options 1` kann festgelegt werden, wie das Archiv ent-/verpackt werden soll
- `source` gibt die Qelldatei an (idR. ein `.tar.gz`-Archiv)
- Mit `options 2` können weitere Optionen, wie das Zielverzeichnis angegeben werden

Beispiel

- `tar xvf update.tar.gz -C ./` entpackt das Archiv `update.tar.gz` in das aktuelle Verzeichnis

Dokumentation

- Dokumentation

Prozesse von FileDescriptor ermitteln mit *fuser*

Beschreibung

Mit dem Befehl `fuser` können die Prozesse ermittelt werden, die auf eineen gewissen FileDescriptor zugreifen. Dies beinhaltet sowohl Netzwerk-Ports, als auch serielle Schnittstellen.

Verwendung

`fuser [options] [port]` - Mögliche `options` - `-k` gibt an, dass die Prozesse beendet werden sollen, die auf den FileDescriptor zugreifen - `port` gibt eine serielle bzw. Netzwerk-Schnittstelle an

Beispiel

- `fuser 2404/tcp` gibt alle Prozesse aus, die den lokalen TCP-Port *2404* verwenden
- `fuser -k 2404/tcp` beendet alle Prozesse, die den lokalen TCP-Port *2404* verwenden
- `fuser /dev/ttySC1` gibt alle Prozesse aus, die auf die serielle Schnittstelle von Bus 1 zugreifen

- `fuser /dev/ttySC0` gibt alle Prozesse aus, die auf die serielle Schnittstelle von Bus 2 zugreifen

Dokumentation

- Dokumentation

TODO: tcpdump

TODO: netstat

Revision #2
Created 24 October 2024 15:05:42 by Philipp Kreutzer
Updated 25 October 2024 09:45:14 by Philipp Kreutzer