

KonnektorTool.py

Table of Contents

1	KonnektorTool.py für das automatische Setzen von KonnektorKonfigurationen	6
1.1	Einleitung	6
1.2	Allgemeine Hinweise	6
1.2.1	Angabe der Konnektor-IP oder IP-Range	6
1.2.2	Sonderzeichen / Übergabe eines Kennwort	6
1.3	Basis Funktionsparameter	7
1.3.1	Mandatorische Parameter	7
1.3.2	Optionale Parameter	7
1.3.3	Modispezifische Parameter	7
1.4	Config File	7
1.5	KonnektorExport - Tool für DVO	8
1.6	Programmcode	8
2	KonnektorExport	9
2.1	KonnektorExport.exe:	9
2.1.1	Kompilieren der KonnektorExport.exe aus dem KonnektorTool	9
2.1.2	Nutzung des KonnektorExports für DVOs	10
3	KonnektorTool Modi / Funktionen	11
3.1	KonnektorTool: Modi und ihre modi-spezifischen Funktionsparameter	11
3.1.1	changePW Modus - Konnektorpasswort eines Benutzers ändern	11
3.1.2	addUser Modus - Neuen Benutzer hinzufügen	11
3.1.3	modUser Modus - Benutzerrechte ändern	12
3.1.4	rmUser Modus - Benutzer vom Konnektor löschen	12
3.1.5	Export Modus - Konnektorkonfiguration exportieren oder ausgeben und Backup erstellen	13
3.1.6	Import Modus - Konnektorkonfiguration von Koco-Box oder Secunet importieren	14

3.1.7	MacToKT Modus - Nach MAC-Adressen benannte KT's im Infomodell werden in ihre angemeldeten CT_IDs umbenannt	16
3.1.8	configParameter Modus - Einzelne Konfigurationsparameter auslesen oder setzen	16
3.1.9	showKt Modus - Kartenlesegeräte ausgeben	17
3.1.10	showCards Modus - Karten ausgeben	18
3.1.11	receiptkey Modus - VSDM Receipt Key setzen oder ausgeben	18
3.1.12	installReady Modus	18
3.1.13	resetError	19
3.1.14	check Modus	19
3.1.15	showRoute Modus - Routen ausgeben	21
3.1.16	addRoute Modus - Routen hinzufügen	21
3.1.17	rmRoute Modus - Routen löschen	21
3.1.18	addKt Modus - Kartenterminal mit dem Konnektor hinzufügen	22
3.1.19	activateKt Modus - Bereits gepairtes Kartenterminal auf aktiv setzen	22
3.1.20	rmKt Modus - Kartenterminal löschen	22
3.1.21	register Modus - Registrierung mit der TI herstellen	23
3.1.22	addInfo Modus - Infomodell anlegen	23
3.1.23	mergeInfo Modus - Infomodell importieren und mit dem bestehenden Infomodell mergen	24
3.1.24	rmInfo Modus - Infomodell löschen	26
3.1.25	renInfo Modus - Infomodellparameter umbenennen	26
3.1.26	showInfo Modus - Infomodell-Basiseinträge anzeigen	27
3.1.27	clientCert Modus - Clientzertifikat generieren und auf den Konnektor hochladen	27
3.1.28	rmClientCert Modus - Clientzertifikat löschen	28
3.1.29	tslUpdate Modus - TSL / CRL / BNetzA Liste aktualisieren	29
3.1.30	fwUpdate Modus - Firmwareupdate hochladen	29
3.1.31	ktupdate Modus - KT Firmware Update starten	30
3.1.32	reboot Modus - Konnektor neustarten	30

3.1.33 synctime Modus - Uhrzeit synchronisieren	30
3.1.34 printip Modus - Konnektor IP-Adresse anzeigen, sofern der Konnektor mit der TI verbunden ist.....	31
3.1.35 genrsa Modus - RSA Public + Private Key generieren	31
3.1.36 changePin Modus - SMC-B Pin am Kartenterminal ändern.....	31
3.1.37 pinStatus Modus - Status des SMC-B Pin abfragen	32
3.1.38 pin Modus - SMC-B Pin Funktionen durchführen.....	32
3.1.39 getlogs Mo.....	32
3.1.40 getlogs Modus - Log-Dateien herunterladen.....	32
3.1.41 chargect Modus - Karten im Kartenterminal erneut abfragen	33
3.1.42 updatect Modus - Admin-User und Admin-Pin zu einem Kartenterminal hinzufügen.....	33
3.1.43 getcert Modus - Zertifikat des Konnektor-Management Interface anzeigen.	34

se

Untergeordnete Seiten:

- [KonnektorExport](#) (see page 9)
- [KonnektorTool Modi / Funktionen](#) (see page 11)

1 KonnektorTool.py für das automatische Setzen von KonnektorKonfigurationen

1.1 Einleitung

Mit dem KonnektorTool können Konfigurationsänderungen über die Webschnittstelle (Port 9443) des Konnektors durchgeführt werden.

Die Webschnittstelle setzt auf https sowie den Login über ein HTML Formular, sie unterstützt weder zertifikatsabsierte Anmeldung, noch die HTTP-Basis-Authentifizierung.

Es können durch angeben einer IP-Range viele Konnektoren auf einmal bearbeitet werden - hier liegt der große Vorteil in der Nutzung des Programmes.

1.2 Allgemeine Hinweise

1.2.1 Angabe der Konnektor-IP oder IP-Range

Grundsätzlich können ein oder mehrere Konnektoren angegeben werden. Die Angabe mehrerer Konnektoren erfolgt als IP-Range in dem Format: 192.168.0.1-17. Hierbei wird der Range auf Plausibilität geprüft, d.h. es muss sich um eine gültige IP Adresse handeln und das zweite Oktett muss höher als das Erste sein. Es werden im Falle der Angabe eines Range alle Konnektoren seriell (d.h. hintereinander) angesprochen und die entsprechende Konfiguration gesetzt.

1.2.2 Sonderzeichen / Übergabe eines Kennwort

Achten sie bitte darauf, dass bestimmte Sonderzeichen innerhalb einer Kommandoshell interpretiert werden, u.a. * \$! & " ' { [()] } / \ :

Eingegebene Kennwörter können innerhalb der Shell in einfachen Anführungszeichen escaped werden. Diese Anführungszeichen werden im Rahmen des Skript-Aufrufes von der Shell nicht übergeben und sind somit nicht Bestandteil des eingegebenen Kennwortes.

'P4ssword!123' führt zum Setzen des Passwortes auf: P4ssword!123

Es wird empfohlen, für die Shell problematische Sonderzeichen grundsätzlich zu meiden und nur die folgenden Sonderzeichen zu nutzen: # + - _ \$ % .

1.3 Basis Funktionsparameter

1.3.1 Mandatorische Parameter

Die folgenden Parameter sind mandatorisch zu übergeben oder in der Config-File zu hinterlegen.

- -I / --ip: IP oder IP Range des Konnektors / der Konnektoren
- -J / --iprange: Mehrere Ranges auf einmal angeben. Achtung: Hier gibt es Erkennungsprobleme mit dem 'positional Argument' des Modus-Parameter
Optimal wie folgt anwenden: `KonnektorTool.py -J <Range1> <Range2> ... <RangeN> -C config.cfg -Q <Modus> <Modusspezifische Parameter>`
- -H / --host: Hostname des Konnektors (statt IP-Adresse, nur einzelner Hostname möglich)
- -U / --user: (Admin-)Benutzername, mit dem der Befehl ausgeführt werden soll, im Falle eines Kennwortwechsels: Benutzer, für den das Kennwort geändert werden soll
- -P / --pass: Aktuelles Kennwort des Benutzers
- Modus: "changepw", "adduser", (Case-Sensitive! Grundsätzlich nur Kleinbuchstaben!), der Modus kann nicht in der Config-File gespeichert werden!

1.3.2 Optionale Parameter

- -C / --config <FileName>: Konfigurationsdatei, siehe nächster Abschnitt.
- -D / --debug: Ignoriere Sanitizing der Input-Variablen
- -Q / --quiet: Zeige keine Login successful Meldung nach dem Einloggen.
- -T / --test: Schreibe keine Änderungen auf den Konnektor. ("Dry run", aber das -D war schon belegt für Debug).
Hilfreich, um bestimmte Fehler vorab zu sehen, z.B. die Überschneidung von Infomodell-Parametern bei mergeinfo.
- -X / --use-proxy: Benutze Proxy zum Herstellen einer Konnektorverbindung.
Default: Ignoriere Systemproxy (Keine Auswirkung auf die Nutzung des Systemproxys zum Download der TSL, CRL und BNetzA Dateien)

1.3.3 Modispezifische Parameter

[\(see page 11\)](#)

1.4 Config File

Die im vorigen Abschnitt "Basis Funktionsparameter" können auch in einer Config-Datei hinterlegt werden. Dies ist insbesondere für Benutzername + Kennwort interessant, damit das Kennwort nicht in der Prozessliste auftaucht. Mit dem Parameter -C bzw. --config <FileName> kann eine Konfigurationsdatei übergeben werden. Diese muss wie folgt gestaltet sein:

```
[defaults]
user = koko-root
password = Geh3im123!
ip = 192.168.1.1
debug = True
```

Achtung: Bool-Werte wie z.B. "debug" oder "use-proxy" können in der Config-File auf True gesetzt werden, jedoch nicht auf "False", da die verwendete Funktion zum Einlesen der Parameter nur Strings zurückliefert und keine Boolean Werte. "debug = False" oder "debug = no" liefert dem KonnektorTool den String "False" oder "no" zurück, und diesen gesetzten String wertet Python als True. Um "debug" auf False zu setzen, müssen sie die gesamte Zeile "debug" oder "use-proxy" aus der Config entfernen.

1.5 KonnektorExport - Tool für DVO

[\(see page 9\)](#)

1.6 Programmcode

v.11.1



2 KonnektorExport

2.1 KonnektorExport.exe:

2.1.1 Kompilieren der KonnektorExport.exe aus dem KonnektorTool

Die KonnektorExport.exe exportiert die Konfiguration von KocoBoxen und das Infomodell sowie die Zertifikate von Secunet Konnektoren. Diese Parameter können im Anschluss in eine KocoBox importiert werden. Dabei können von Secunet Konnektoren lediglich die Zertifikate und das Infomodell verwendet werden. Das Migrieren des Infomodells (merge Infomodell) von Secunet Konnektoren auf KocoBoxen mit bestehender Konfiguration ist möglich. (Weitere Informationen zum Unterschied zwischen Infomodell-Import und Infomodell merge siehe unter "mergeinfo Modus").

Das KonnektorExport-Tool beherrscht ein sehr eingeschränktes Subset von Funktionen des KonnektorTools. Die Konvertierung in eine .exe Datei geschieht mit einem installierten Pyinstaller. Hierfür wird die Quellcode-Datei von KonnektorTool.py umbenannt in KonnektorExport.py. Das KonnektorTool prüft, ob der eigene Name KonnektorTool oder KonnektorExport lautet, und stellt im zweiten Fall lediglich das eingeschränkte Subset bereit. Dabei hilft es **nicht**, die spätere .exe Datei in wieder von KonnektorExport in KonnektorTool zu benennen, um die Funktion wiederherzustellen, allerdings ist es nicht unmöglich, die Datei zu dekompileieren. Hierzu wurden allerdings keine weitergehenden Tests durchgeführt.

Um vollen Funktionsumfang des Export-Tools zu erlangen (inkl. AES-Zip mit integrierter RSA-Verschlüsselung des Zip-AES-Schlüssels), müssen vor dem Kompilieren neben dem pyinstaller auch weitere Tools via pip nachinstalliert werden:

```
pip install pycryptodome
pip install pyzipper
pip install openssl
```

Im Anschluss helfen folgende Kommando bei der Konvertierung in eine .exe Datei.

```
rmdir /s /q latestExe
mkdir toExe
copy KonnektorExport.py ./toExe
pyinstaller --noconfirm --onefile --console --add-data "./toExe;."
"KonnektorExport.py"
rmdir /s /q build
rmdir /s /q toExe
del KonnektorExport.spec
ren dist latestExe
explorer latestExe
```

pause

2.1.2 Nutzung des KonnektorExports für DVOs

Es wird eine Config-Datei benötigt, die der DVO mit den passenden Werten füllen kann. Diese hat den folgenden Aufbau.

config.cfg

```
[defaults]
user = koko-root
password = Geh3im123!
ip = 192.168.1.1
```

Darüber hinaus wird eine .cmd Datei (oder Batch-Datei .bat) erstellt, die die KonnektorExport mit den passenden Parametern aufruft.

starte_export.cmd

```
KonnektorExport.exe -C config.cfg export -e
```

export : Erstellt eine unverschlüsselte, unkomprimierte JSON Datei

export -z : Erstellt ein unverschlüsseltes zip-Archiv

export -e : Erstellt ein RSA-verschlüsseltes Zip Archiv, welches nur vom Tlaas-Team ausgelesen werden kann.

export -z -b CT_ID_0001 : Fügt neben der json-Konfiguration auch ein Konnektor-Backup hinzu, welches mit dem Kartenterminal CT_ID_0001 via Pin-Abfrage freigegeben wird.



3 KonnektorTool Modi / Funktionen

3.1 KonnektorTool: Modi und ihre modi-spezifischen Funktionsparameter

3.1.1 changePW Modus - Konnektorpasswort eines Benutzers ändern

Ändert das Kennwort eines oder mehrerer Konnektoren für einen bestimmten Benutzer. Achten sie auf den Hinweis bzgl. der Sonderzeichen von Kennwörtern.

Achtung: Das Ändern von Kennwörtern klappt leider nicht zuverlässig für bestehende Benutzer. Es wird eine positive Rückmeldung vom Konnektor geliefert, obwohl das Kennwort nicht geändert wurde. Daher wird empfohlen die Funktion nicht direkt zu verwenden, sondern stattdessen den User zu löschen und neu anzulegen ('rmuser' gefolgt von 'adduser'). Hierfür muss sich ein (zusätzlicher) SuperAdmin User auf dem Konnektor befinden.

Funktionsspezifische Parameter:

- Eingabe des neuen Kennwortes

```
# Kennwort ändern
$./KonnektorTool.py -I 172.18.112.129 -U koko-root -P PASSWORD changew NEWPASSWORD1#
Login successful: 172.18.112.129
Passwort geändert!
```

3.1.2 addUser Modus - Neuen Benutzer hinzufügen

Erstellt einen neuen Benutzer mit einer vordefinierten Rolle und ändert das Standardkennwort des Benutzers auf einen spezifischen, angegebenen Wert.

Achten sie darauf, dass das neue Kennwort an eine Richtlinie gebunden ist (z.B.: Mind. 1 Sonderzeichen). Geben sie in Kennwort falsch ein, erhalten sie ggf. dennoch eine "ok" Quittung, obwohl das Kennwort nicht geändert wurde!

Funktionsspezifische Parameter:

- Benutzername des zu erstellenden Benutzers
- Benutzerrolle: ("SuperAdmin", "Admin", "LogDownloader", "Supporter")
- Kennwort des zu erstellenden Benutzers

Optionale Parameter (Funktionieren nicht mit jeder Benutzerrolle!)

- -f / --factory-reset : Benutzer hat das Recht, den Konnektor auf Werkseinstellung zu setzen.
- -r / --remote-session : Benutzer hat das Recht, eine Remote-Session aufzubauen.

```
# Benutzer hinzufügen
$./KonnektorTool.py -C config.cfg adduser super-mario SuperAdmin NEWPASSWORD1#
Login successful: 172.18.112.129
User super-mario angelegt. Logge mit neuem User ein und ändere Kennwort...
```

3.1.3 modUser Modus - Benutzerrechte ändern

Ändert einen bestehenden Benutzer im Hinblick auf die Rolle oder die Berechtigung für Werksreset und Remote-Session.

Funktionsspezifische Parameter:

- Benutzername des zu modifizierenden Benutzers
- Benutzerrolle: ("SuperAdmin", "Admin", "LogDownloader", "Supporter")

Optionale Parameter (Funktionieren nicht mit jeder Benutzerrolle!)

- -f / --factory-reset : Benutzer hat das Recht, den Konnektor auf Werkseinstellung zu setzen.
- -r / --remote-session : Benutzer hat das Recht, eine Remote-Session aufzubauen.

```
# Benutzer-Rechte auf "factory-reset" und "remote-session" erweitern
$./KonnektorTool.py -C config.cfg moduser super-mario SuperAdmin -factory-reset
-remote-session
Login successful: 172.18.112.129
User super-mario angelegt. Logge mit neuem User ein und ändere Kennwort...
***** Es müssen zwingend Sonderzeichen im Kennwort verwendet werden, sonst steht hier
"OK" und die Änderung schlägt trotzdem fehl!*****
Login successful: 172.18.112.129
User super-mario aktualisiert.
```

3.1.4 rmUser Modus - Benutzer vom Konnektor löschen

Löscht bestehende Benutzer. Die Anmeldung am Konnektor muss mit einem SuperAdmin erfolgen und die zu löschenden Benutzer werden als funktionsspezifischer Parameter angegeben. Das Kennwort der zu löschenden Benutzers wird nicht benötigt. Der Benutzer, mit dem die Aktion ausgeführt wird kann nicht gelöscht werden, d.h. ein zu löschender Nutzer und der ausführende Benutzer dürfen nicht identisch sein.

Funktionsspezifische Parameter:

- Benutzername der zu löschenden Benutzer

```
# Benutzer löschen
$./KonnektorTool.py -C config.cfg rmuser super-mario super-luigi
Login successful: 172.18.112.129
User super-mario erfolgreich gelöscht!
User super-luigi erfolgreich gelöscht!
```

3.1.5 Export Modus - Konnektorkonfiguration exportieren oder ausgeben und Backup erstellen

Exportiert verschiedene Parameter aus dem Konnektor und

- schreibt die erhobenen JSON-Daten in eine Konfigurationsdatei
- gibt die JSON-Daten in einem übersichtlichen JSON-Format aus

Auslesen eines Secunet-Konnektors: Sollte unter der angegebenen IP Adresse keine Koco-Box antworten, versucht das Tool im Export Modus stattdessen zu einem Secunet-Konnektor zu verbinden und im Erfolgsfalle diesen auszulesen. Beim Secunet-Konnektor wird nur das Infomodell mitsamt der Zertifikate ausgelesen.

Auf einer Koco-Box kann der Modus entweder alle konnektorspezifischen Informationen auslesen oder nur spezifische Unterseiten. Die folgenden spezifischen Service-Namen existieren. Sie erhalten die Liste auch, indem sie als Parameter "export --service help" eingeben.

- "infoservice"
- "certificateservice"
- "protocolservice"
- "lanwanservice"
- "vpnservice"
- "registrationservice"
- "ntpservice"
- "dnsservice"
- "management"
- "cardservice"
- "clientsysteme"
- "fmvsdm"
- "update"
- "updates"
- "signservice"
- "cardterminalservice"
- "usermanagement"
- "umgebung"
- "clientsystem-certificates"
- "routingtable"
- "users"
- "cards"
- "cardterminals"
- "infomodell" (darf auch "infomodel" heißen, weil die Konnektor Export Funktion diesen Namen verwendet)

Funktionsspezifische Parameter:

- optional: "-s / --service <Servicename>": Exportiert alle übergebenen Services, statt alle Services.
- optional: "-p / --print": Gibt die Informationen auf der Kommandozeile aus, anstatt sie in eine Datei zu schreiben
- optional: "-f / --file <FileName>": Dateiname, unter der die Datei gespeichert wird.

- optional: "-z / --zip" : Schreibt den Export in eine Zip Datei (kombinierbar mit -file). Mehrere Konnektoren werden in eine einzige Zip-Datei geschrieben.
- optional: "-e / --encrypt" : Schreibt den Export in eine AES Verschlüsselte ZIP Datei mit Zufallspasswort. Das Zufallspasswort steht in einer unverschlüsselten Text-Datei innerhalb der Zip-File, ist aber mit einem
- optional: "-b / --backup [<CtId>]": Fügt ein Backup-Export hinzu, erwartet als Parameter die Kartenterminal ID mit SMC-B oder ICCSN für die Bestätigung.

Siehe hierzu auch die Konnektor JSON Schnittstellen API Dokumentation im Hinblick auf alle verfügbaren Services:

1

```
# Export des infomodel und cardservice
$./KonnektorTool.py -C config.cfg export -f 172.18.112.129.json -s infomodel
cardservice
Login successful: 172.18.112.129
172.18.112.129.json written

# Ausgeben der Benutzer
$./KonnektorTool.py -I 172.18.112.135 -U koko-root -P PASSWORD export -s users -p
Login successful: 172.18.112.135
{
  "users": [
    {
      "CONTACTINFO": "",
      "ID": "koko-root",
      "ROLE": "SuperAdmin",
      "USER_INIT_REMOTESESSION": false,
      "USER_RESET_PERMISSION": false
    }
  ]
}
```

3.1.6 Import Modus - Konnektorkonfiguration von Koco-Box oder Secunet importieren

Liest eine mit dem Export-Modus erstellte Konfigurationsdatei ein und schreibt sie zu einem oder mehreren neuen Konnektoren.

Client-Zertifikate und Lanwanservice werden bei einem Full-Import nur importiert, wenn -overwrite gesetzt wird. Sofern die Services einzeln als solche angegeben werden, werden diese auch ohne Overwrite importiert. Clear-Clientcerts ist ggf. empfohlen, sofern nicht mehr benötigte Zertifikate bereits auf dem Konnektor vorhanden sind.

Wenn kein Service importiert werden soll (z.B. weil ausschließlich die Kartenlesegeräte hinzugefügt werden sollen) geben sie "-service none" ein.

Funktionsspezifische Parameter:

- -f / --file <Import Datei>: Dateinamen der Konfigurationsdatei als JSON oder ZIP mit *einer* enthaltenen JSON oder als verschlüsseltes ZIP (*eine* JSON + .txt mit RSA verschlüsseltem AES Passwort)
- optional: "-s / --service <Servicename>": Importiert nur die angegebenen Services.
- optional: "-a / --add-kt" : Versucht alle Kartenterminals aus der Import-Datei erneut hinzuzufügen und zu pairen
- optional: "-b / --correlate" : Versucht alle Kartenterminals aus der Import-Datei erneut hinzuzufügen (kein Zugriff auf das KT erforderlich, aber CTID wird generiert)
- optional: "-c / --certs <clear> / <replace>": Löscht vor dem Import alle <clear>/ bereits vorhandene <replace> Zertifikate.
- optional: "-e / --encrypt <eFileName>: Import einer verschlüsselten Export-Datei. Erwartet als Parameter eine .pem-Datei (oder ZIP mit enthaltener private.pem) zum Entschlüsseln.
- optional: "-m / --macimport: Benennt CT_IDs im Infomodel in ihre zugehörigen MAC Adressen aus der Import-Datei um. Sinnvoll für die Migration auf einen neuen Konnektor.
- optional: "-l / --lanwanconfig [<key:value>] : Importiert beim Hochladen auch die lanwan-Parameter.
optional: Setzen von key:value Parametern analog zum config-Parameter Modus.
Ermöglicht das hochladen einer generalisierten Konfigurationsdatei auf viele Konnektoren durch individuelles setzen von lanwanconfig-Parametern (z.B. LAN-IP oder Konnektortname)

```
# Import des infomodel
$./KonnektorTool.py -C config.cfg import -f 172.18.112.129.json -s infomodel
Login successful: 172.18.112.129
***** Hinweis: Firmware-Unterschied erkannt: Konnektor Firmware: 4.2.22, Import
Firmware: unknown! *****
infomodel aus 172.18.112.129.json erfolgreich auf den Konnektor hochgeladen

# Config von einem Drittkonnektor importieren und während des Upload die IP-Adresse
ändern
$./KonnektorTool.py -C config.cfg import -f 10.219.1.1-export.json -l
ANLW_LAN_IP_ADDRESS:10.219.10.1
Login successful: 172.18.112.129
***** Hinweis: Firmware-Unterschied erkannt: Konnektor Firmware: 4.2.22, Import
Firmware: unknown! *****
infomodel aus 10.219.1.1-export.json erfolgreich auf den Konnektor hochgeladen
```

3.1.7 MacToKT Modus - Nach MAC-Adressen benannte KT's im Infomodell werden in ihre angemeldeten CT_IDs umbenannt

Gegenstück zu 'import -m'

Sollte eine Ziel-CT_ID bereits belegt sein, wird diese vorher in eine freie CT_ID umbenannt.

Idempotent: Kann mehrfach hintereinander aufgerufen werden, sofern Kartenterminals sukzessive angemeldet werden.

```
# Umbenennen
$./KonnektorTool.py -C config.cfg mactokt
Login successful: 172.18.112.135 mit User: TiaasUIview
KT 00:1B:B5:06:27:53 als ID CT_ID_0006 gefunden. Benenne um...
KT 00:0D:F8:03:F3:3F als ID CT_ID_0004 gefunden. Benenne um...
KT 00:1B:B5:08:C2:2D als ID CT_ID_0013 gefunden. Benenne um...
KT 00:1B:B5:07:CA:4D als ID CT_ID_0000 gefunden. Benenne um...
KT 00:0D:F8:09:24:EA als ID CT_ID_0011 gefunden. Benenne um...
KT 00:1B:B5:06:FB:36 als ID CT_ID_0010 gefunden. Benenne um...
KT 00:1B:B5:0B:85:40 als ID CT_ID_0048 gefunden. Benenne um...
KT 00:0D:F8:05:79:C2 als ID CT_ID_0016 gefunden. Benenne um...
KT 00:1B:B5:08:A3:10 als ID CT_ID_0018 gefunden. Benenne um...
KT 00:1B:B5:09:DB:DC als ID CT_ID_0047 gefunden. Benenne um...
KT 00:0D:F8:06:25:76 als ID CT_ID_0046 gefunden. Benenne um...
KT 00:0D:F8:0B:91:C0 als ID CT_ID_0012 gefunden. Benenne um...
KT 00:1B:B5:07:CA:35 als ID CT_ID_0002 gefunden. Benenne um...
KT 00:1B:B5:09:DA:EC als ID CT_ID_0019 gefunden. Benenne um...
ERFOLGREICH: infomodell hochgeladen
```

3.1.8 configParameter Modus - Einzelne Konfigurationsparameter auslesen oder setzen

Fragt einen spezifischen Wert vom Konnektor ab und gibt ihn aus oder schreibt ihn in den Konnektor. Eignet sich gut, um via Abfrage einer Konnektor-Range die Konfiguration eines spezifischen Wertes (z.B. LAN MTU) zwischen vielen Konnektoren zu vergleichen, oder um einzelne Werte auf einem oder mehreren Konnektoren zu verändern.

Funktionsspezifische Parameter:

- Service-Name (siehe Export Modus)
- Lesen: Parameter in der Form 'KEY', z.B. 'ANLW_LAN_MTU'
- Schreiben: Parameter in der Form 'KEY:VALUE', z.B. 'ANLW_LAN_MTU:1300'

Pro Programmaufruf können Parameter nur innerhalb eines ServiceName gelesen oder geschrieben werden. Es können aber mehrere Parameter auf einmal lesend oder schreibend angepasst werden. Dabei ist eine Mischung aus lesen und schreiben möglich. Die Vorgänge werden hintereinander durchgeführt, siehe Beispiel in Beispielbox.

Siehe hierzu auch die Konnektor JSON Schnittstellen API Dokumentation im Hinblick auf alle verfügbaren Parameter:

2

3

```
# Setzen der LAN MTU auf den Konnektoren .130-135 , hier können auch mehrere
Parameter innerhalb des gleichen Services gleichzeitig gesetzt / gelesen werden.
$./KonnektorTool.py -C config.cfg configparameter lanwanservice anlw_wan_mtu:1400
anlw_lan_mtu:1300
Login successful: 172.18.112.130
ANLW_WAN_MTU:1400 eingefügt, noch nicht gespeichert!
ANLW_LAN_MTU:1300 eingefügt, noch nicht gespeichert!
Änderungen wurden auf den Konnektor geschrieben!
...

# Setzen des Konnektors auf 'offline'
./KonnektorTool.py -I 172.18.112.135 -U koko-root -P PASSWORD configparameter
management mgm_lu_online:DISABLE
Login successful: 172.18.112.131
MGM_LU_ONLINE:DISABLE eingefügt, noch nicht gespeichert!
Änderungen wurden auf den Konnektor geschrieben!
```

3.1.9 showKt Modus - Kartenlesegeräte ausgeben

Gibt eine Liste aller angeschlossenen Kartenlesegeräte aus:

- optional: -e / --empty : nur KTs ohne Karten
- optional: -l / --list : gibt nur die CtId als Liste zurück
- optional: -a / --active: gibt nur die aktiven Kartenterminals zurück

```
# Anzeige aller angeschlossenen Kartenlesegeräte
$./KonnektorTool.py -I 10.219.1.1 -U koko-root -P PASSWORD showkt
Login successful: 10.219.1.1
CTID: CT_ID_0007, MAC: 00:0D:F8:05:0E:25 Typ: ORGA6100 1.2.0 / 3.8.1, IP: 10.10.1.5,
CTID: CT_ID_0017, MAC: 00:0D:F8:05:C7:E7 Typ: ORGA6100 1.2.0 / 3.8.2, IP: 10.10.1.3,
CTID: CT_ID_0021, MAC: 00:0D:F8:0B:90:22 Typ: ORGA6100 1.2.0 / 3.8.2, IP: 10.10.1.8,
CTID: CT_ID_0005, MAC: 00:0D:F8:07:AF:2B Typ: ORGA6100 1.2.0 / 3.8.2, IP: 10.10.1.4,
CTID: CT_ID_0018, MAC: 00:1B:B5:08:71:24 Typ: * noch nicht bekannt *, IP: 0.0.0.0,
```

3.1.10 showCards Modus - Karten ausgeben

Gibt eine Liste aller gesteckten Karten aus. Standardmäßig werden alle Karten ausgegeben.

Funktionsspezifische Parameter:

- -b / --smcb : Gibt nur Karten des Typs SMC-B aus
- -k / --smckt : Gibt nur Karten des Typs SMC-KT aus

```
# Anzeige aller gesteckten Karten
./KonnektorTool.py -C config.cfg showcards
Login successful: 10.219.1.1
CTID: CT_ID_0017, ICCSN: 80276003550000457566, Typ: SMC-KT
CTID: CT_ID_0023, ICCSN: 80276003550000524355, Typ: SMC-KT
CTID: CT_ID_0023, ICCSN: 80276002711610010944, Typ: SMC-B
CTID: CT_ID_0022, ICCSN: 80276003550000524333, Typ: SMC-KT
CTID: CT_ID_0020, ICCSN: 80276003550000524322, Typ: SMC-KT
```

3.1.11 receiptkey Modus - VSDM Receipt Key setzen oder ausgeben

Lässt einen Fachmodul VSDM Key durch den Konnektor generieren und lädt ihn hoch.

Funktionsspezifische Parameter:

- -m / --mandant: Legt Mandanten fest, für die ein Key generiert wird (Default: alle fehlenden)
- -k / --key: Optionaler Key, sofern kein Key generiert werden soll
- -p / --print: Gibt alle Mandanten und zugehörigen Keys aus

```
# Receipt Key generieren lassen und hochladen
./KonnektorTool.py -C config.cfg receiptkey -m Test
Login successful: 172.18.112.135 mit User: koco-root
ERFOLGREICH: Receipt für Mandant Test auf sTz3BAh(/4+%gKW5 geändert
```

3.1.12 installReady Modus

Hält einen nicht mit der TI-Verbundenen Konnektor 'install-ready', damit er mit dem SelfService Tool bereitgestellt werden kann.

Hierfür werden die folgenden Funktionen in der folgenden Reihenfolge ausgeführt:

- Abbrechen, falls mit TI verbunden
- Leistungsumfang offline stellen
- Uhrzeit synchronisieren
- TSL/BNetzA/CRL aktualisieren
- Leistungsumfang online stellen
- Automatisches "Clear FATAL-Security-Error", falls existent

```

./KonnektorTool.py -C config.cfg -I 10.219.1.6-7 installready
Login successful: 10.219.1.6 mit User: TiaasUIview
Konnektor ist mit der TI verbunden. Breche ab...
Login successful: 10.219.1.7 mit User: TiaasUIview
MGM_LU_ONLINE:DISABLE gesetzt
Datum auf 10.08.2023 17:40:30 gesetzt
TSL-ECC erfolgreich hochgeladen
CRL erfolgreich hochgeladen
BNetzA erfolgreich hochgeladen
MGM_LU_ONLINE:ENABLE gesetzt
Es besteht kein EC OTHER STATE ERROR (1), daher ist kein Reset erforderlich.

```

3.1.13 resetError

Setzt einen EC OTHER ERROR STATE (1) Fehler zurück, sofern der Konnektor nicht mit der TI Verbunden ist und der Fehler besteht

```

./KonnektorTool.py -C config.cfg -I 10.219.1.7 reseterror
Login successful: 10.219.1.7 mit User: TiaasUIview
Es besteht kein EC OTHER STATE ERROR (1), daher ist kein Reset erforderlich.

```

3.1.14 check Modus

Prüft die Konnektorkonfiguration auf logische Fehler. Die folgenden Fehler werden geprüft:

4

Funktionsspezifische Parameter:

- -v / --verbose: Ausführlicher Bericht statt nur Fehlermeldungen

```

krause@RU-JUMPHOST-TIAAS:~/KonnektorTool$ ./KonnektorTool.py -C config.cfg -I 172.18.1.12.129 check -v
Login successful: 172.18.112.129 mit User: koko-root

#### Prüfe Konnektor Firmware ####
Konnektor Firmware Version: 4.2.24

#### Prüfe TSL/CRL/BNetzA Listen ####
TSL und BNetzA Listen sind gültig!

#### Prüfe, ob Umstellung auf ECC erfolgt ist ####
TSL ist bereits auf ECC umgestellt.

```

Prüfe ob Konnektor online ist ####
Konnektor ist online!

Prüfe ob Konnektor mit der TI verbunden ist ####
Konnektor ist mit der TI verbunden!

Prüfe, ob Bestandsnetzte deaktiviert sind ####
FEHLER: Bestandsnetz 84.17.163.80/30 "Demis Meldeportal" wurde manuell deaktiviert und sollte reaktiviert werden!
FEHLER: Bestandsnetz 84.17.163.80/30 "Demis Meldeportal" wurde manuell deaktiviert und sollte reaktiviert werden!

Prüfe, ob Nexthop einer Route außerhalb des LAN Netzbereiches liegt ####
Alle Nexthop Adressen befinden sich im eigenen Netzbereich

Prüfe auf fmvsdm Receipt Keys ####
FEHLER: Mandant CKR ohne fmvsdm Receipt Key!
FEHLER: Mandant IPR ohne fmvsdm Receipt Key!
FEHLER: Mandant DRO ohne fmvsdm Receipt Key!
FEHLER: Mandant SWI ohne fmvsdm Receipt Key!
FEHLER: Mandant MZI ohne fmvsdm Receipt Key!
FEHLER: Mandant LLA ohne fmvsdm Receipt Key!
FEHLER: Mandant Test1 ohne fmvsdm Receipt Key!
FEHLER: Mandant Service_Mandant ohne fmvsdm Receipt Key!
FEHLER: Mandant MA491234567 ohne fmvsdm Receipt Key!
FEHLER: Mandant LukeMStar ohne fmvsdm Receipt Key!
FEHLER: Mandant cert ohne fmvsdm Receipt Key!
FEHLER: Mandant checkmk ohne fmvsdm Receipt Key!
*** Rufen sie das Konnektortool mit dem Parameter "-I 172.18.112.129 receiptkey" erneut auf, um alle fehlenden fmvsdm Keys zu generieren.

Prüfe, ob Kartenterminals im Status "ACTIVE" sind ####
Alle Kartenterminals im Status "ACTIVE"!

Prüfe, ob Kartenterminal Verknüpfung zu Service_Mandant und Konnektor Arbeitsplatz besitzt ####
FEHLER: CT_ID_0067 besitzt keine Verknüpfung zu Mandant "Service_Mandant"!
FEHLER: CT_ID_0067 besitzt keine Verknüpfung zu Arbeitsplatz "Konnektor"!

Prüfe, ob Kartenterminals alle Slotnummern angehakt hat ####
FEHLER: CT_ID_0067: Nicht alle Kartenterminal-Slots angehakt!

Prüfe, ob Clientsysteme über ein Zertifikat verfügen ####
FEHLER: ClientSystem ID M1 verfügt über kein Client Zertifikat
FEHLER: ClientSystem ID MEDISTAR01 verfügt über kein Client Zertifikat
FEHLER: ClientSystem ID Z1 verfügt über kein Client Zertifikat

3.1.15 showRoute Modus - Routen ausgeben

Liest die Konnektor-Routing-Table aus. Der Befehl erwartet keine optionalen Parameter. Die Routing-Tabelle ist ein zusammengesetztes Objekt und daher kein Bestandteil des GetInfo Modus.

```
$/KonnektorTool.py -C config.cfg showroute
Login successful: 172.18.112.135
IP_SEG          PROTOCOL      FWD_STATUS      NEXT_HOP          TYPE
PREFERENCE
172.18.112.0/24  kernel        eingeschaltet    172.18.112.135    statisch
5
192.168.2.0/24   kernel        eingeschaltet    192.168.2.57      statisch
5
172.31.31.0/24   kernel        eingeschaltet    172.18.112.254    statisch
5
172.17.48.0/24   kernel        eingeschaltet    172.18.112.254    statisch
5
10.2.0.0/15      kernel        eingeschaltet    172.18.112.254    statisch
5
10.0.0.0/15      kernel        eingeschaltet    172.18.112.254    statisch
5
                  kernel        eingeschaltet    192.168.2.254     statisch
5
```

3.1.16 addRoute Modus - Routen hinzufügen

Fügt eine neue, statische Route zum Konnektor hinzu. Die Route wird im Format 1.2.3.4/5 angegeben, wobei 5 die Netzmaske ist.

Funktionsspezifische Parameter:

- Route
- NextHop

```
# Hinzufügen einer Route
$/KonnektorTool.py -C config.cfg addroute 10.10.10.0/24 192.168.1.1
Login successful: 172.18.112.135
10.10.10.0/24 192.168.1.1 erfolgreich hinzugefügt.
```

3.1.17 rmRoute Modus - Routen löschen

Löscht bestehende, statische Routen aus dem Konnektor. Die Routen werden im Format 1.2.3.4/5 angegeben, wobei 5 die Netzmaske ist.

Funktionsspezifische Parameter:

- Route

```
# Route löschen
$./KonnektorTool.py -C config.cfg rmroute 10.10.10.0/24
Login successful: 172.18.112.135
10.10.10.0/24 erfolgreich entfernt.
```

3.1.18 addKt Modus - Kartenterminal mit dem Konnektor hinzufügen

Fügt Kartenterminals hinzu. Die Kartenterminals werden mit ihrer IP im Format 1.2.3.4 angegeben.

Funktionsspezifische Parameter:

- IP Adressen d. Kartenterminals
- optionaler Parameter: -r <ContractID List>: Sofern bisher kein TI Verbindung besteht, wird mit dem hinzugefügten Kartenterminal bei gesteckter SMC-B eine TI-Verbindung hergestellt.
Es muss eine JSON Liste angegeben werden, in welcher die Konnektor IP und die zugehörige Contract ID enthalten ist. Die Liste erwartet das folgende JSON-Format:

```
{
  "10.219.1.1": "CONTRACTID1234",
  "10.219.1.2": "CONTRACTID1235"
}
```

3.1.19 activateKt Modus - Bereits gepairtes Kartenterminal auf aktiv setzen

Setzt ein bereits bestehendes Kartenterminal erneut auf aktiv. Erwartet als Parameter eine IP-Adresse oder eine Kartenterminal ID.

Funktionsspezifische Parameter:

- IP Adresse d. Kartenterminals oder Kartenterminal ID

3.1.20 rmKt Modus - Kartenterminal löschen

Löscht ein bestehendes Kartenterminal aus dem Konnektor. Das Kartenterminal muss als ID (CT_ID_0XXX) oder IP-Adresse angegeben werden.

Funktionsspezifische Parameter:

- ID des Kartenterminals ODER IP-Adresse des Kartenterminals

```
# KT löschen
$./KonnektorTool.py -C config.cfg rmkt CT_ID_0001
Login successful: 172.18.112.135
Entfernt: Kartenterminal CT_ID_0001
```

3.1.21 register Modus - Registrierung mit der TI herstellen

Zeigt den Status der TI Verbindung an, stellt eine TI Verbindung her oder trennt sie.

Funktionsspezifische Parameter:

- - ohne Parameter - : Zeigt den Verbindungsstatus an
- -c / --connect <ICCSN> / <Kartenterminal ID>: Stellt die Verbindung mit der ICCSN her. Alternativ Eingabe der Kartenterminal ID.
- -d / --disconnect : Baut die Verbindung ab
- optional: -i / --contract-id <Contract ID> : Hinterlegt die Contract ID (funktioniert aufgrund von Konnektorbeschränkungen nur in Kombination mit -c)

3.1.22 addInfo Modus - Infomodell anlegen

Fügt einen Mandant, Clientsystem, Arbeitsplatz (auch mehrere), Kartenlesegerät und SMB Objekt zum Infomodell hinzu.

Funktionsspezifische Parameter:

- -m / --mandant <Mandant>
- -c / --clientsystem <Clientsystem>
- -a / --arbeitsplatz <Arbeitsplatz1> <Arbeitsplatz2> ...
- -k / --kartenterminal <Kartenterminal>
- -s / --smb <ICCSN> [<SMB-ID>] oder -s / --smb <Kartenterminal ID> [<SMB-ID>]: Erstellt die SMB Objekte SMB2M und SMBEN.
- -r / --remote <Kartenterminal> : Das unter -kt angegebene Kartenterminal enthält keine SMC-B, deshalb wird als -remote <Kartenterminal> ein Kartenterminal des gleichen Mandanten mit SMC-B zur Remote-Freischaltung angegeben. Hierzu erstellt das Tool ein RPINKTS Objekt mit dem Mandant / Arbeitsplatz / Kartenterminal, sowie das passende KTe2M Objekt mit Mandant und dem angegebenen Remote-Kartenterminal.
- -x / --safemode: Fügt das Infomodell nur dann hinzu, wenn weder Mandant / Clientsystem / Arbeitsplatz / Kartenterminal bereits existieren. Sollte verwendet werden beim Anlegen eines neuen Kunden auf dem Konnektor, um zu verhindern, dass es eine Überlappung mit einem bestehenden Infomodell eines anderen Kunden gibt.
- -y / --add-ap : Fügt einen Arbeitsplatz zu bestehenden Mandanten / Clientsystem hinzu und erstellt die zugehörigen Objekte. Mandant und Clientsystem müssen existieren, dadurch ist sichergestellt, dass der neue Arbeitsplatz einem bestehenden Kunden hinzugefügt wird.

Der SMB Parameter erwartet eine ICCSN oder alternativ eine Kartenterminal ID als Parameter. Darüber hinaus kann als zweiter Parameter eine SMB-ID angegeben werden. Ansonsten wird an an den angegebenen Mandanten der String "SMCB" angehängt und als SMB-ID verwendet.

Achtung: Im Modus rmInfo wird zum Löschen der SMB Einträge stattdessen die SMB-ID erwartet. Dort funktionieren weder die ICCSN noch die Kartenterminal ID.

Das Tool erstellt alle (fehlenden) Infomodelle Objekte, die mit den gegebenen Parametern erstellt werden können. Sofern Kartenterminal und Clientsystem, Remote und SMB angegeben wird, sind das alle Objekte:

- Arbeitsplatz
- Mandant
- Clientsystem
- Kartenterminal

- AP2M (Arbeitsplatz zu Mandant)
- CS2M (Clientsystem zu Mandant)
- KT2M (Kartenterminal zu Mandant)

- CSAPS (Mandant / Clientsystem / Arbeitsplatz Verknüpfung)

Nur bei Angabe von -remote:

- KTe2M (Extended Kartenterminal zu Mandant) [Kartenterminal mit SMC-B zur Remote Freischaltung]
- KPINKTS (Kartenterminal OHNE SMC-B zu Arbeitsplatz und Mandant Verknüpfung) mit Berechtigung zur Remote-Pin über KTe2M Objekt

Nur bei Angabe von -smb:

- SMB2M (SMB-ID zu Mandant)
- SMBEN (SMB-ID zu ICCSN)

Mandant und Arbeitsplatz sind mandatorisch.

Darüber hinaus muss mindestens einer der folgenden Parameter angegeben werden: Clientsystem, Kartenterminal, SMB.

```
# Infomodelle hinzufügen
$./KonnektorTool.py -C config.cfg addinfo -m MANDANT1 -c CLIENT1 -a Arbeitsplatz1 -k
CT_ID_0001 --safemode
Login successful: 172.18.112.135
Hinzugefügt: Mandant MANDANT1
Hinzugefügt: Arbeitsplatz Arbeitsplatz1
Hinzugefügt: Clientsystem CLIENT1
Hinzugefügt: Kartenterminal CT_ID_0001
Hinzugefügt: KT2M Objekt mit MANDANT1 und CT_ID_0001
Hinzugefügt: KT2AP Objekt mit Arbeitsplatz1 und CT_ID_0001
Hinzugefügt: AP2M Objekt mit MANDANT1 und Arbeitsplatz1
Hinzugefügt: CS2M Objekt mit MANDANT1 und CLIENT1
Gespeichert: Änderungen auf den Konnektor geschrieben
```

3.1.23 mergeInfo Modus - Infomodelle importieren und mit dem bestehenden Infomodelle mergen

Importiert ein Infomodelle (z.B. aus einem anderen Export) in ein bestehendes Infomodelle auf dem Konnektor. Kann auch Secunet Infomodelle importieren. Hierbei ist wichtig, dass es keine Überschneidung zwischen den Einträgen im bestehenden Modell und im neuen Modell gibt. Zwangsweise Überschneidungen der Default Parameter (Mandant: Service_Mandant, Clientsystem: intern, Arbeitsplatz: Konnektor) werden beim Import

ignoriert und nicht erneut angelegt, gleiches gilt für Mandant/Clientsystem/Arbeitsplatz mit dem Namen checkmk.

Die Kartenterminals werden, sofern sie bereits im Infomodell existieren, vor dem Import umbenannt. Sofern im Rahmen des Merge Kartenterminals aus der Import-Datei neu am Konnektor angemeldet werden, wird automatisch die hierbei zugewiesene Kartenterminal-ID auch im Infomodell verwendet. Beim Import werden darüber hinaus MAC-Adressen von KT's in der Import Datei mit MAC-Adressen von angemeldeten KT's verglichen, sofern Kartenterminals übereinstimmen wird beim Import die CT_ID aus der Sektion Kartenterminals im Infomodell übernommen.

Der Switch --add-kt fügt Kartenterminals aus der import-Konfiguratoin zum Konnektor hinzu. Hierzu muss der Abschnitt "Cardterminals" in der import-json-Datei enthalten sein. Das funktioniert allerdings nur, sofern sich die IP-Adressen der Kartenterminals nicht geändert haben und ist damit für viele TlaaS Fälle nicht nutzbar.

Standardmäßig wird ein Backup des Infomodells vor dem Merge abgelegt, mit --no-backup wird kein Backup angelegt.

Der Switch --certs importiert alle in der Import-Datei enthaltenen Zertifikate, überschreibt bereits vorhandene Zertifikate jedoch nicht. Der Import setzt voraus, dass die Zertifikate in der import-json-Datei enthalten sind.

Unterschiede mergeinfo / import -s infomodell:

- mergeinfo importiert das infomodell, lässt aber bestehende Einträge auf dem Zielkonnektor des Infomodells bestehen (import -s infomodell überschreibt das Infomodell des Zielkonnektors komplett mit dem Infomodell aus der Import-Datei)
- im Konfliktfall mit bestehenden Infomodell-Parametern, die mit denen in der Import-Datei übereinstimmen wird der Import abgebrochen
- Kartenterminals werden im Infomodell umbenannt, sofern die MAC Adressen der Geräte aus der Import-Datei mit denen aus dem Zielkonnektor übereinstimmen. Das heißt: Ist ein Kartenterminal mit MAC-Adresse 00:11:22:33:44:55 in der Import-Datei als CT_ID_0015 im Infomodell enthalten, und ist dieses Kartenterminal als CT_ID_0018 bereits am Zielkonnektor angemeldet, wird beim Infomodell-Merge jeder Infomodell Eintrag mit KT CT_ID_0015 durch CT_ID_0018 umgeschrieben. Wichtig dafür ist, dass das KT am Zielkonnektor bereits angemeldet ist, aber noch nicht im Infomodell des Zielkonnektors steht, denn ansonsten schlägt der Import fehl.
- import -s infomodell eignet sich ideal, um bei versehentlicher Config-Änderung ein Backup zurückzuspielen, ohne dass die Logik des Infomodells geprüft wird.
- mergeinfo eignet sich, um zwei Konfigurationen zusammenzuführen oder einen Konnektor auf eine andere Hardware umzuziehen, denn in diesem Fall bekommen die KT's eine neue CT_ID beim Anmelden zugewiesen, die ansonsten manuell umgezogen werden müsste.

Funktionsspezifische Parameter:

- -f / --file: Import-Datei mit Infomodell als JSON oder ZIP mit *einer* enthaltenen JSON oder als verschlüsseltes ZIP (*eine* JSON + .txt mit RSA verschlüsseltem AES Passwort)
- optional: -e / --encrypt <eFileName>: Import einer verschlüsselten Export-Datei. Erwartet als Parameter eine .pem-Datei (oder ZIP mit enthaltener private.pem) zum Entschlüsseln.
- optional: -z / --certs: Importiert in der import-Datei enthaltene Zertifikate, ohne bereits vorhandene Zertifikate zu überschreiben
- optional: -n / --no-backup: Erstellt kein Backup des Infomodells vor dem Import
- optional: -x / --add-kt: Fügt in der import-Datei enthaltene Kartenterminals zum Konnektor hinzu
- optional: -m / --mandant <Mandant>: Liste von Mandanten, die nicht importiert werden (Default: Service_Mandant, checkmk)

- optional: -c / --clientsystem <Clientsystem>: Liste von Clientsystemen, die nicht importiert werden (Default: intern, checkmk)
- optional: -a / --arbeitsplatz <Arbeitsplatz>: Liste von Arbeitsplätzen, die nicht importiert werden (Default: Konnektor, checkmk)
- optional: -k / --kartenterminal <Kartenterminal>: Liste von Kartenterminals, die nicht importiert werden (Default: none)
- optional: -s / --smb <SMD ID>: Liste von SMB-IDs, die nicht importiert werden (Default: none)

```
# Infomodellparameter mergen
$ ./KonnektorTool.py -C config.cfg mergeinfo --file infomodell.json
Login successful: 172.18.112.129
... (tbd)
```

3.1.24 rmInfo Modus - Infomodell löschen

Löscht Mandanten, Clientsysteme, Arbeitsplätze, Kartenlesegeräte und SMB IDs aus dem Infomodell und löscht alle zugehörigen Infomodell Objekte.

Funktionsspezifische Parameter:

- -m / --mandant <Mandant>
- -c / --clientsystem <Clientsystem>
- -a / --arbeitsplatz <Arbeitsplatz>
- -k / --kartenterminal <Kartenterminal>
- -s / --smb <SMD ID>

```
# Infomodellparameter löschen
$ ./KonnektorTool.py -C config.cfg rmInfo -kt CT_ID_0064 -cs test1 -ma Mandant1 -ap Sentinel_WS
Login successful: 172.18.112.129
Entfernt: Mandant Mandant1 aus dem Infomodell entfernt.
Entfernt: Clientsystem test1 aus dem Infomodell entfernt.
Entfernt: Arbeitsplatz Sentinel_WS aus dem Infomodell entfernt.
Entfernt: Kartenterminal CT_ID_0064 aus dem Infomodell entfernt.
Gespeichert: Änderungen auf den Konnektor geschrieben
```

3.1.25 renInfo Modus - Infomodellparameter umbenennen

Benennt Mandanten, Clientsysteme, Arbeitsplätze, Kartenlesegeräte und SMB IDs aus dem Infomodell um.

Beim Umbenennen des Clientsystems wird auch das hinterlegte Clientsystem-Zertifikat zum neuen Clientsystem umgezogen.

Beim Umbenennen eines Mandanten wird auch der VSDM Receipt Key zum neuen Mandanten umgezogen.

Funktionsspezifische Parameter:

- -m / --mandant <Mandant>

- -c / --clientsystem <Clientsystem>
- -a / --arbeitsplatz <Arbeitsplatz>
- -k / --kartenterminal <Kartenterminal>
- -s / --smb <SMD ID>

```
# Infomodelparameter umbenennen
./KonnektorTool.py -C config.cfg reninfo -a Mueller Meier -k CT_ID_0051 CT_ID_0001
Login successful: 172.18.112.135
Umbenannt: Arbeitsplatz Meier nach Mueller umbenannt!
Umbenannt: CT_ID_0051 nach CT_ID_0001 umbenannt!
Gespeichert: Änderungen auf den Konnektor geschrieben
```

3.1.26 showInfo Modus - Infomodel-Basiseinträge anzeigen

Zeigt alle Mandanten / ClientSysteme / Arbeitsplätze an, die existieren.

Funktionsspezifische Parameter:

- optional: -m / --mandant: Beschränkt die Ausgabe auf Mandanten
- optional: -c / --clientsystem: Beschränkt die Ausgabe auf ClientSysteme
- optional: -a / --arbeitsplatz : Beschränkt die Ausgabe auf Arbeitsplätze
- optional: -k / --kartenterminal : Beschränkt die Ausgabe auf Kartenterminals
- optional: -d / --dict : Gebe die Daten als Python-Dict zurück (nur sinnvoll in Kombination mit -Q)
- Hinweis: Obige Parameter können kombiniert werden. => Keine Eingabe eines Parameters entspricht -m -c -a -k

```
# Infomodel anzeigen
$./KonnektorTool.py -C config.cfg showinfo -m -c
Login successful: 172.18.112.135 MA:01
MA:Service_Mandant
MA:checkmk
CS:intern
CS:checkmk
```

3.1.27 clientCert Modus - Clientzertifikat generieren und auf den Konnektor hochladen

Lädt für eine angegebene ClientsystemID ein Zertifikat in den Konnektor, oder generiert selbst eines mit einer zu definierenden Laufzeit (bis 1 - 10 Jahre, Default = 5 Jahre). Erwartet oder generiert eine Zertifikatsdatei im Format .crt. Eine ca.crt muss nicht hochgeladen werden. Die ClientsystemID muss mit einer ClientsystemID im Infomodel übereinstimmen. Im Falle der Strikten TLS Prüfung kann nur mit dieser ClientsystemID eine Abfrage über die SOAP Schnittstelle erfolgen.

Funktionsspezifische Parameter:

- **ClientsystemID** : Gibt die ClientsystemID an, unter der das Zertifikat hochgeladen wird
Dieser Name wird als Dateiname ("Basename") verwendet, wenn kein anderer Name angegeben.
D.h. unter diesem Namen wird die Zertifikatsdatei gesucht (oder abgespeichert mit '-generate')
- optional: **-g / --generate [<Laufzeit>]**: Generiert eine Zertifikatsdatei und speichert sie unter ab, statt diese zu laden.
- optional: **-r / --replace**: Ersetzt ein eventuell auf dem Konnektor vorhandenes Zertifikat mit der gleichen ClientsystemID
- optional: **-c / --common-name <Common Name>**: Die autorisierende Stelle nutzt den angegebenen Common Name (default: KoCoBox)
- optional: **-f / --file <FileName>** : Zertifikatsdatei zum Hochladen oder zum Abspeichern eines generierten Zertifikates

```
# Client-Zertifikat mit der ClientsystemID 'test' mit einer Laufzeit von 1000 Tagen
generieren und zum Konnektor hochladen
$./KonnektorTool.py -C config.cfg clientcert test -r -g 1000
Login successful: 172.18.112.135
-----BEGIN CERTIFICATE-----
MIIDDjCCAFagAwIBAgIEBjb1aDANBgkqhkiG9w0BAQsFADASMRAdgYDVQDDAdL
b0NvNm94MB4XDTIyMDgxODA1MzQ1NFoXDTI1MDUxNDA1MzQ1NFowDzENMA5GA1UE
AwwEdGVzdDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALXyFTLn3Bkb
kDGCR+p28cnf2EPKc8cYgK0CM9kZjeATTVA2PW0tx+uEAlGEUc4IWCauxXuBuCiL
j71mZh40vyfoFJHRA99SV7QnI2stZMUGqMybAu8MpOPfyEfaJhoafpeinK0M9xvn
DAK4Ih5TdBSXwpZv3n+CerwVIB6wS6IiEFLZlypt6w1KMYAdc7p2qtn6ha+zmKzB
zMW5fepxhQMgZKz0LI+4uu820GXDYxDZWar3CpzdEjLq0t8rwHG29BFISLG3XvFP
SbkNHyx+HdVBwf3d0u0ineo56z5J4FFsHsC7YdKneEI/NIFe6DbP6/zLvKRltLyG
0GzyFctGdRsCAwEAAANvMG0wCQYDVROTBAlwADAdBgNVHQ4EFgQU2jmj7l5rSw0y
Vb/vlWAYkK/YBwkWwYDVR0jBBgwFoAU2jmj7l5rSw0yVb/vlWAYkK/YBwkWwYD
VR0lBAwwCgYIKwYBBQUHAWIwCwYDVROPBQAQAgeAMA0GCSqGSIb3DQEBCwUAA4IB
AQASPF/KN70NMx1HyBOcCXRpLAQRHUG40M6RHrUX8wxInhgdrz0AWiU0WzFSAY6g
+Cbg0vl32Eq8j78o86om/rai8QvtwefUSFyUMGcUj0QxMyXh1R5TfUq4nz71q4eI
MtJZehsB9KYg7RJyqKNkcMa5MRkaJ80EXbt9hw9Kmh2CBrnEf9D8hbveX/8onQ6d
A+W70R98Wsvof0ae7LgZ6ouBzSpwwV7RxyAN2nQ7bt2lvGuUN4SnnTE5kfpVWemq
HL5ZTvjrZ6BtPujAaGScm4X4uwHss68oFAaXKdt3yaDkM/klcAwXZNIKn4E+Nbm
vldhivu1jD3DTnjDuVxtwkKg
-----END CERTIFICATE-----

ClientCert erfolgreich hochgeladen
```

3.1.28 rmClientCert Modus - Clientzertifikat löschen

Löscht ClientCerts vom Konnektor.

Funktionsspezifische Parameter:

- **ClientsystemID** der Zertifikate

- 'all' als ClientsystemID löscht alle Zertifikate, sofern kein Zertifikat mit ClientsystemID namens 'all' existiert (dann wird nur diese gelöscht)

```
# Clientzertifikat entfernen
$./KonnektorTool.py -C config.cfg rmclientcert test
Login successful: 172.18.112.135
Vorhandenes ClientCert test erfolgreich gelöscht!
```

3.1.29 tslUpdate Modus - TSL / CRL / BNetzA Liste aktualisieren

Aktualisiert die TSL, CRL und BNetzA Liste auf dem Konnektor. Hierbei wird geprüft, ob es sich um eine Produktiv- oder Test-Umgebung handelt sowie im Falle der TSL Liste, ob eine Eliptic Curve (ECC) oder eine RSA Datei benötigt wird.

Ohne Parameter werden alle Listen aktualisiert.

- - kein Parameter - : Sofern der Konnektor nicht mit der TI verbunden ist, werden alle Listen automatisch heruntergeladen und aktualisiert (TSL, dann CRL, dann BNetzA)
- -t / --tsl Aktualisiert die TSL Liste.
- -c / --crl Aktualisiert die CRL Liste.
- -b / --bnetza Aktualisiert die BNetzA Liste.
- -F / --force : Erzwingt die Aktualisierung auch dann, wenn der Konnektor mit der TI verbunden ist.
- -o / --online: Setzt den Konnektor vor dem update nicht auf offline, sondern führt das Update online durch. Schlägt manchmal fehl, aber obligatorisch für aktiv genutzte Konnektoren außerhalb eines Wartungszeitraumes.

```
# TSL und CRL Datei aktualisieren
$./KonnektorTool.py -C config.cfg upload --tsl --crl
Login successful: 172.18.112.135
Online Status des Konnektors auf "Disable" gesetzt
TSL-ECC Datei erfolgreich hochgeladen
CRL Datei erfolgreich hochgeladen
Online Status des Konnektors auf "Enable" gesetzt
```

3.1.30 fwUpdate Modus - Firmwareupdate hochladen

Führt ein Firmwareupdate durch.

Funktionsspezifische Parameter:

- optional: -s / --set-date <DD.MM.YYYY>: Setzt das Datum auf einen definierten Wert, wird kein Wert angegeben, wird der 01.06.2022 gewählt. Der Konnektor muss hierfür auf 'offline' gesetzt werden.
- optional: -f / --file <Sig File> <Zip File> : Lädt manuell eine Firmwareupdate Datei sowie eine Signaturdatei hoch, anstatt den Konnektor selbst die aktuelle Firmware-Version herunterladen zu lassen.

- optional: -F / --force : Erzwingt das Firmwareupdate, auch wenn der Konnektor mit der TI verbunden ist.
- optional: -p / --poke: Bringt den Konnektor dazu, ein bereits heruntergeladenes Firmwareupdate unmittelbar zu installieren

```
# Firmwareupdate durchführen
$./KonnektorTool.py -C config.cfg fwupdate -f update.sig update.zip
Login successful: 172.18.112.135
Firmware Signatur hochgeladen
Firmware Update hochgeladen
Firmware Update gestartet
```

3.1.31 ktupdate Modus - KT Firmware Update starten

Führt alle Firmwareupdates aller angeschlossenen KTs durch

```
./KonnektorTool.py -C config -I 172.18.4.30-31 ktupdate
Login successful: 172.18.4.30 mit User: koko-root
*****CT_ID_0002: Update 4.0.0 auf 4.0.25 geplant!*****
*****CT_ID_0001: Update 3.8.2 auf 3.9.0 geplant!*****
*****CT_ID_0000: Update 3.8.2 auf 3.9.0 geplant!*****
Updates gestartet!
Login successful: 172.18.4.31 mit User: koko-root
*****CT_ID_0002: Update 3.8.2 auf 3.9.0 geplant!*****
*****CT_ID_0005: Update 3.8.2 auf 3.9.0 geplant!*****
Updates gestartet!
```

3.1.32 reboot Modus - Konnektor neustarten

Führt einen Reboot durch.

```
# Reboot
$./KonnektorTool.py -C config.cfg reboot
Login successful: 172.18.112.135
Reboot durchgeführt
```

3.1.33 syntime Modus - Uhrzeit synchronisieren

Synchronisiert die Zeit mit der aktuellen Systemzeit.

Wenn der Konnektor mit der TI verbunden ist, wird eine Synchronisation mit der TI angestoßen.

Wenn der Konnektor nicht mit der TI verbunden ist, wird er offline gesetzt, die aktuelle Systemzeit des Hosts wird übermittelt und im Anschluss wieder online gesetzt.

```
$/KonnektorTool.py -C config.cfg synctime  
Login successful: 172.18.112.135  
Datum auf 05.09.2022 14:32:10 gesetzt
```

3.1.34 printip Modus - Konnektor IP-Adresse anzeigen, sofern der Konnektor mit der TI verbunden ist

Gibt die IP Adresse des Konnektors auf der Kommandozeile aus, wenn der Konnektor online ist. Mit -Q kombiniert erhält man eine Liste aller Konnektoren, die mit der TI verbunden sind.

```
# IP Adresse aller Online-Konnektoren anzeigen  
$/KonnektorTool.py -C config.cfg -Q -I 10.219.1.1-143 printip  
10.219.1.1
```

3.1.35 genrsa Modus - RSA Public + Private Key generieren

Generiert eine Zip-Datei mit einem private.pem + public.pem, welches für die zip-Verschlüsselung genutzt werden kann. Achtung: public.pem muss in das Konnektor-Tool kopiert werden und bestehende Archive (z.B. aus dem Konnektor Export Tool) können mit den neuen Zertifikaten nicht mehr entschlüsselt werden. Vermutlich keine Funktion, die du wirklich nutzen möchtest. (Muss i.d.R. einmal beim Aufbau einer Infrastruktur genutzt werden)

```
# IP Adresse aller Online-Konnektoren anzeigen  
$/KonnektorTool.py -C config.cfg -Q -I 10.219.1.1-143 genrsa  
'private.pem' written to 'rsa.zip'  
'public.pem' written to 'rsa.zip'
```

3.1.36 changePin Modus - SMC-B Pin am Kartenterminal ändern

Ändert den SMC-B Pin am Kartenterminal.

Funktionsspezifische Parameter:

- -s / --smb [<CTID> | <ICCSN>]
- -m / --mandant <Mandant>

```
# SMC-B Pin am Kartenterminal ändern  
./KonnektorTool.py -C config.cfg -I 172.18.112.135 changepin -s CT_ID_0008 -m Mandant  
Login successful: 172.18.112.135  
Bitte ändern sie den Pin am Kartenterminal!  
ERFOLGREICH: Pin geändert!
```

3.1.37 pinStatus Modus - Status des SMC-B Pin abfragen

Prüft den SMC-B Pinstatus.

Funktionsspezifische Parameter:

- -s / --smb [<CTID> | <ICCSN>]
- -m / --mandant <Mandant>

```
# Status des SMC-B Pin abfragen
./KonnektorTool.py -C config.cfg -I 172.18.112.135 pinstatus -s CT_ID_0008 -m Mandant
Login successful: 172.18.112.135
Der Pin-Status ist VERIFIABLE
```

3.1.38 pin Modus - SMC-B Pin Funktionen durchführen

Prüft den SMC-B Pinstatus.

Funktionsspezifische Parameter:

- -f / --function : Führt eine der folgenden Funktionen aus: status, verify, unblock, change
- -s / --smb [<CTID> | <ICCSN>]
- -m / --mandant <Mandant>
- -c / --clientsystem <Clientsystem>
- -a / --arbeitsplatz <Arbeitsplatz>
- -p / --print : Schreibt das übermittelte JSON zusätzlich auch als Print-Ausgabe

```
# Status des SMC-B Pin abfragen
./KonnektorTool.py -C config.cfg -I 172.18.112.135 pin -f verify -s CT_ID_0008 -m
Mandant -c Client -a Arbeitsplatz
Login successful: 172.18.112.135
Der Pin-Status ist REJECTED. Left-Retries: 2
```

3.1.39 getlogs Mo

3.1.40 getlogs Modus - Log-Dateien herunterladen

Aktualisiert die Karten im Kartenterminal

Funktionsspezifische Parameter:

- -l / --logtype op / sys / sec / perf
- -c / --count: Anzahl an Log Einträgen (nicht bei op(erational state) logs)

- -i / -id: Log ID, ab der die Logs angezeigt werden sollen (die übermittelte Log-ID ist die erste ID, die NICHT mehr übermittelt wird, d.h. die letzte ID, die bereits übertragen wurde.)
(nicht bei operational state) logs)

Die Logs werden in einem Array an Dicts übergeben. Die Logs werden nicht aufbereitet.

```
# Kartenterminal aktualisieren
./KonnektorTool.py -C config.cfg -I 172.18.112.135 getlogs -c 3
Login successful: 172.18.2.1 mit User: TiaasUIview
[{'ID': 79658, 'TIMESTAMP': '18.09.2024 10:56:31.338', 'SEVERITY': 'ERR', 'TEXT':
'ADMINISTRATIONSSERVICE/ERROR', 'AMOUNT': 1, 'PARAMETERS': 'Bedeutung=Interner
Fehler; ErrorType=Technical; Error=4001; ErrorSeverity=Error'}, {'ID': 79657,
'TIMESTAMP': '18.09.2024 10:55:39.324', 'SEVERITY': 'ERR', 'TEXT':
'ADMINISTRATIONSSERVICE/ERROR', 'AMOUNT': 1, 'PARAMETERS': 'Bedeutung=Interner
Fehler; ErrorType=Technical; Error=4001; ErrorSeverity=Error'}, {'ID': 79626,
'TIMESTAMP': '18.09.2024 09:55:17.763', 'SEVERITY': 'ERR', 'TEXT':
'ADMINISTRATIONSSERVICE/ERROR', 'AMOUNT': 1, 'PARAMETERS': 'Bedeutung=Interner
Fehler; ErrorType=Technical; Error=4001; ErrorSeverity=Error'}]
k
```

3.1.41 chargect Modus - Karten im Kartenterminal erneut abfragen

Aktualisiert die Karten im Kartenterminal

Funktionsspezifische Parameter:

- -k / --kartenterminal <CTID>

```
# Kartenterminal aktualisieren
./KonnektorTool.py -C config.cfg -I 172.18.112.135 chargect -k CT_ID_0008
Login successful: 172.18.112.135
Kartenterminal aktualisiert
```

3.1.42 updatect Modus - Admin-User und Admin-Pin zu einem Kartenterminal hinzufügen

Fügt Admin-User und Admin-Pin zu einem Kartenterminal hinzu

Funktionsspezifische Parameter:

- -k / --kartenterminal <CTID>

```
# Kartenterminal aktualisieren
./KonnektorTool.py -C config.cfg -I 172.18.112.135 updatect -k CT_ID_0008 -u admin -p
123456
```

Login successful: 172.18.112.135
Admin-Pin aktualisiert

3.1.43 getcert Modus - Zertifikat des Konnektor-Management Interface anzeigen

Fügt Admin-User und Admin-Pin zu einem Kartenterminal hinzu

Funktionsspezifische Parameter:

- -k / --kartenterminal <CTID>

```
# Konnektor Management Interface Zertifikat herunterladen
./KonnektorTool.py -J 172.18.4.1 -C config.cfg -Q getcert
-----BEGIN CERTIFICATE-----
MIIeYDCCA7CgAwIBAgIDCBC0MA0GCSqGSIb3DQEBCwUAMG8xCzAJBgNVBAYTAkRF
MRUwEwYDVQQKDAxnZW1hdGlrIEdtYkgxMjAwBgNVBAsMKUtvbXBvbmVudGVuLUNB
IGRlcjBUZSwxZW50aWtpbmZyYXN0cnVrdHVyMRUwEwYDVQQDDAxHRU0uS09NUC1D
QTcwHhcNMjIwNzExMTEzMDEzMDEzMDEzMDEzMDEzMDEzMDEzMDEzMDEzMDEzMDE
REUxDzANBgNVBAGMBk1lcXpjbjEPMA0GA1UEBwwGQmVybGluMQ4wDAYDVQQRDAUx
MDk2MzE2bWkGA1UECQwSRGVzc2F1ZXJzdHIuIDI4LzI5MRwwGgYDVQQKBDBNLb0Nv
IENvbW5lY3RvcjBHBWJIMSYwJAYDVQQDDDB0MDI3NjAwMzY0MDAwMTM5NDA1Mi0y
MDIyMDcxMTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALMBd5rRx6Qo
sX76erZFsQRsZaXItrc6ncvh28sE0yCxlA/cgyFp8YNv4S7SHv5BAM8p0L4ZQ67
WG84iZV13TVE9+TzULIPzkf+0GtpUq4/8m3XkgNkYNaES8viprJF6Iku88pLTYX3
Qz88kMnuqymnrBmkQ4+Z5Pxs/GrbV2bxdorJ4b90Njg9WI/XCGjlbPzVf709nRJI
nrYOXzuUx83k2WlCDnhDRGnuVY7E/K7oYN0EIqXCB5nZuy/csCCgw3r7ZyK3uGSy
mXrET8aeItD8200sD80a3ys0ELY3QNdQiVmv+iXNPejl0edE0ti2oNwf5wAdG0b
So1uUP/FGtMAwEAAaOCATcwggEzMB0GA1UdDgQWBBS0826/ugnPZLmiHa32Kb0u
4zp6wjAFBgNVHSMEGDAWgBQEs/QWMEU7e5FK83zwieUJTpJRLDA+BggRBgEFBQcB
AQQyMDAwLgYIKwYBBQUHMAAGGIh0dHA6Ly9vY3NwLmtvbXBvXAtY2EudGVsZW1hdGlr
L29jc3AwDgYDVROPAQH/BAQDAgWgMCAGA1UdIAQZMBcwCgYIKoIUAwEgSMwCQYH
KoIUAwETzAbBgNVHREEFDAsgHBrb25uZWt0b3Iua29ubGFuMAwGA1UdEwEB/wQC
MAAwHQYDVROlBBYwFAYIKwYBBQUHAWEGCCsGAQUFBwMCMdUGBSskCAMDBCwwKjAo
MCYwJDAiMBUME0Fud2VuZHVuZ3Nrb25uZWt0b3IwCQYHKOIUAwEzZANBgkqhkiG
9w0BAQsFAAOCAQEApFe5pgwYwmbkENNjlvnTecwFFIQCzu0TRjAWJopH7viJfSa
u8ljFaq8r7pKYQnEMJW7mpTxkPCV+wxMvw15uUo640kLCXXLb67c/ZgoGCtC+wJS
z1ZquNyGIafByi9/zT1ZhHSUDI0zsKDKhhtqcvHpmxQqWjhHE/2LJCYNAL0rR4ZI
BJIzC5VZuzjo6oQrk65HyaqJZqvoGsDKcNfLT7kiqfuM8lQhIUBHOxgxynb/fB5o
pSNgx7iPkuqf9/044nG+iaXwFteGbgYE7JbMeJjDFpRT5Ih2VSCKtCF+lSZm9a6U
fB0yPcaIEJKmtDci3pDntx4l0B1vxuJ9CBDUNg==
-----END CERTIFICATE-----
```