TrueConf

TrueConf Border Controller

Administrator guide

Table of Contents

1. Description

The all-in-one TrueConf Enterprise solution includes the TrueConf Border Controller extension that provides external users (outside the corporate network environment) with secure access to video conferencing servers.

TrueConf Border Controller is a separate extension that acts as a border controller designed to be installed in the DMZ (demilitarized zone) of the corporate network and allowing only secure traffic from TrueConf client applications.

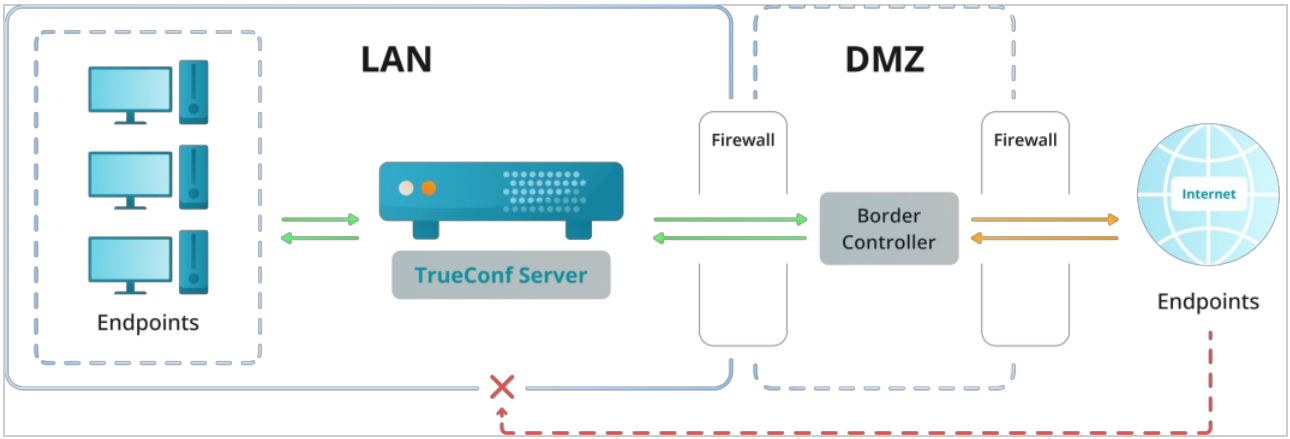
1.1. Solution Composition

The extension consists of two components that validate traffic via TrueConf and HTTP/HTTPS protocols respectively.

> *i* We recommend using HTTPS on TrueConf Server since it improves the security of the web server resources and ensures the work of the scheduler, real-time meeting management tool, browser-based conference participation and access to a user's personal area.

Components of TrueConf Border Controller have to be configured separately, and are able to work independently of each other which means that it is possible to configure traffic transfer only via TrueConf protocol, but not via HTTPS.

How TrueConf Border Controller works:



1.2. Operating Principle

1. The TrueConf Border Controller extension is installed in the DMZ.
2. The extension checks the protocols of the traffic that comes from the external network.
3. If the traffic does not come via TrueConf or HTTPS protocol, it will simply be rejected.
4. If the extension detects traffic from TrueConf client applications or via HTTPS, the connection will be accepted and a new connection will be created from TrueConf Border Controller to the selected TrueConf Server or TrueConf Enterprise instance. When the connection is established, the packets from the application will be sent via the new connection to the video conferencing server (only TrueConf and HTTPS traffic is allowed). This does not only ensure the transfer of media streams, but also ensures the work of the scheduler, access to the video conferencing server web pages, federation and other features.
5. If necessary, the traffic from TrueConf Border Controller to the video conferencing server can be encrypted with the help of multiple symmetric algorithms, including PSK (Pre-Shared Key).
6. Apart from encryption, the extension does not perform any additional operations on traffic like

analysis, saving, transfer to third-party services, and so forth.

So, the protection of a video conferencing server installed inside the corporate network is based on the following principles:

- TrueConf Border Controller does not create a new connection to TrueConf Server until it determines that the packets are coming from TrueConf client application or via the secure HTTPS protocol.
- No external traffic is directed to the video conferencing server by TrueConf Border Controller. This includes the traffic via SIP/H.323/RTP and others. For example, onlyTrueConf client applications will be able to connect to TrueConf Server from outside the network.
- The IP address of the video conferencing server inside the corporate network is hidden. The server only has to be connected to the DMZ, but it does not have to be connected to the Internet. Please note that server federation will be impossible if there is no connection with the Internet.
- Additionally, it is possible to encrypt the traffic sent via TrueConf protocol.

Every component of the extension is an executable file that does not require installation. It can be run from the console or added as a service on Windows or daemon on Linux.

2. System requirements

We recommend installing TrueConf Border Controller on a physical or virtual server in the DMZ; this server has to meet the following minimum requirements (based on the estimated 800 Mbit/s of transit traffic):

Parameter	Value
Operating system	Dedicated or virtual 64-bit operating system • Microsoft Windows Server 2008 R2/2012/2016/2019/2022 (including the Core edition) with latest updates installed • Debian 10 / 11
CPU	Any CPU with at least four physical cores
RAM	4 GB
Available disk space	Disk space available for saving log files generated by the extension (if the extension is activated)

Contact our technical support to learn more about the exact requirements depending on the number of each TrueConf Border Controller component instances running in parallel on a single machine and the expected volume of traffic.

Next, we will show you how to configure the launch of components on Windows OS and Linux family OS.

If you have any questions regarding the configuration of TrueConf Border Controller, please contact our technical support.

3. The component for the TrueConf protocol

It is an executable file named **tc_bc**. Its launch parameters can be specified in additional parameters after the path or in the configuration file connected with the help of `-c` parameter. An example of the configuration file is presented after the list of parameters.

3.1. Launching from the terminal

To launch the component as a console application, go to the terminal and run this command:

```sh
<path_to_border_controller> <parameters>
```

where `<path_to_border_controller>` is the path to the executable file while `<parameters>` is the list of parameters (check below).

For example, this is how the file can be run on **Windows** with the only required `-d` parameter (TrueConf Server address):

```sh
"C:\Program Files\TrueConf\tc_bc.exe" -d 10.110.10.10
```

Launching on **Linux** with the only required `-d` parameter:

```sh
"/opt/trueconf/enterprise/etc/bc/tc_bc" -d 10.110.10.10
```

3.2. Adding as a service or daemon

This approach is more convenient due to multiple reasons:

- The process can be started automatically on behalf of an OS account (no need to start a user session); however, at the same time, the process can be controlled manually.
- Within the OS, it is possible to create multiple services/daemons based on one application so that traffic could be directed to multiple instances of TrueConf Server without deploying multiple virtual or physical servers in the DMZ.
- The OS log will include records of when the service/daemon was started and stopped and will include messages if some errors or any other issues occur.

3.2.1. Adding a service on Windows

To add a component to the list of Windows services, you can use the command `New-Service` in PowerShell or use the utility **sc.exe** .

To make it easier and more flexible to support the extension, we recommend specifying the parameters in the linked configuration file (check the `-c` parameter) instead of listing them after the path.

Creating a service using **PowerShell**:

1. Run PowerShell as the administrator.
2. Run the command based on the following example:

```sh
$params = @{
  Name = "TrueConf Border Controller"
  BinaryPathName = "c:\trueconf\tc_bc.exe -c c:\trueconf\tc_bc.cfg --service"
  DisplayName = "TrueConf Border Controller"
  StartupType = "Automatic"
  Description = "This is a TrueConf Border Controller service."
}
new-service @params
```


where you need to specify the path to the file `tc_bc.exe` and the configuration file in the `BinaryPathName` parameter. Please note that `--service` is a **required** parameter.

* If a path, e.g., the path to the configuration file includes spaces, it should be put in double quotes " and these quotes should be escaped with this character ` (inverted single quote), for example:

```sh
`"c:\trueconf\trueconf\tc_bc.cfg`"
```

To learn more about it, check the PowerShell documentation .

Creating a service using the **sc** utility through **cmd** (terminal):

1. Run **cmd** as administrator.
2. Run the command based on the following example:

```sh
sc create TrueConf_Border_Controller binPath= ""C:\TrueConfBorderController\tc_bc.exe"
--service --ConfigFile C:\TrueConfBorderController\tc_bc.cfg"
DisplayName=TrueConf_Border_Controller type=own start=auto
```

3.2.2. Adding a daemon on Linux

When adding a daemon on Linux, one has to create a launch file for it (this file is called ***unit***) in the directory `/etc/systemd/system`. For example, to launch on behalf of **root** and read the configuration file **tc_bc.cfg**:

1. Move the executable file of the **tc_bc** component and the configuration file **tc_bc.cfg** in the same directory, e.g., `/opt/trueconf/enterprise/etc/bc/`. The list of parameters for the configuration file is presented below.
2. Create a unit file, for instance, **tbc.service**, using the command:

```sh
sudo nano /etc/systemd/system/tbc.service
```

3. Save the following content in the file:

```sh
[Unit]
Description=TrueConf Border Controller
After=network.target

[Service]
ExecStart=/usr/bin/tbc.sh
Restart=always
RestartSec=3

[Install]
WantedBy=multi-user.target
```

4. Specify the following launch command in the script `tbc.sh` that should be placed under the path from the `ExecStart` parameter:

```sh
#!/bin/bash
/opt/trueconf/enterprise/etc/bc/tc_bc -c /opt/trueconf/enterprise/etc/bc/tc_bc.cfg
```

5. To update the configuration, run this command in the subsystem **systemd**:

```sh
sudo systemctl daemon-reload
```

6. To manually start the daemon, execute the command:

```sh
sudo systemctl start tbc.service
```

7. You can check the daemon's operating status using the command:

```sh
sudo systemctl status tbc.service
```

The created daemon will be launched automatically when the system is started providing that the network configuration is checked successfully.

* To learn more about creating a Linux daemon, read the documentation for your OS or check the general guides such as Linux Handbook .

3.3. List of parameters

The TrueConf Border Controller component for redirecting TrueConf traffic supports the following launch parameters (in brackets we have specified alternatives for some of them).

3.3.1. Command-line parameters (cannot be used in the configuration file)

- `-h` (`--help`) — the display of the built-in help menu with the list of parameters and examples
- `-c <path>` (`--ConfigFile <path>`) — the path `<path>` to the configuration file
- `-v` (`--version`) — the component version.

3.3.2. General parameters

- `--Debug <level>` — the logging level from **0** (disabled) to **4**
- `--LogDirectory <path>` — the path for saving log files related to the extension
- `--LogToConsole` — log messages are displayed in the console instead of being written to log files
- `--Daemonize <path to the PID lock-file>` **(only for Linux) — start as a daemon with the path for saving the PID file
- `--Service` **(only for Windows)** — start as a service
- `--R` — automatic service restart in case of an error

3.3.3. Redirection parameters

- `-D <id>/<host>:<port>` (`--Destination <id>/<host>:<port>`) — the address or FQDN of TrueConf Server or TrueConf Enterprise where traffic will be redirected. Here:
  - `<id>` — (optional) unique string of the identifier used for combining options (when the same TrueConf Border Controller should follow several redirection rules, we **do not recommend this approach**)
  - `<host>` — IPv4, IPv6 or FQDN (IPv6 has to be specified in square brackets `[IPv6]`)
  - `<port>` — (optional) port, this parameter can be omitted if it is equal to the default value **4307**
- `-L <id>/<host>:<port>` (`--Listen <id>/<host>:<port>`) — the network interface for receiving

the incoming traffic; the options of this parameter match the options for the `-D` parameter

- `-E <id>/<cipher>:<flags>:<key>` (`--Encryption <id>/<cipher>:<flags>:<key>`) — encryption of packets sent from TrueConf Border Controller to the video conferencing server. Here:
  - `<id>` — (optional) unique string of the identifier for combining options
  - `<cipher>` — selected encryption method, it can have such values as `None` (no encryption, default), `ChaCha20`, `AES-256-CTR`, `AES-256-OFB`, `AES-192-CTR`, `AES-192-OFB`, `AES-128-CTR`, `AES-128-OFB`, `xoshiro256++`, `xoshiro256**`
  - `<key>` — encryption key (in the hexadecimal format) may be omitted if a randomly generated value is used (incompatible with PSK mode)
  - `<flags>` — if this parameter is available and is equal to `PSK`, Pre-Shared Key encryption will be used. In this case some additional configuration will be needed on the side of the video conferencing server.

3.4. An example of a configuration file:

```
LogDirectory=/opt/trueconf/enterprise/etc/bc/logs/
Listen=10.140.10.123
Destination=10.110.10.10
Encryption=ChaCha20
```


4. HTTPS component

This is an executable file named **webproxy_windows_amd64** for Windows or **webproxy_linux_amd64** for Linux. The settings for the component are specified in the configuration file `webproxy.toml` as shown below.

The component supports three launch options: from the console, as a Windows service, or a Linux daemon. They are configured in the same way as for the previously discussed traffic component for TrueConf, but with a number of differences:

- You need to configure certificate operation beforehand;
- Next to the executable file, you need to create a configuration file **webproxy.toml**;
- The file is launched with the **run** parameter (more details on the use in Windows and Linux are provided below).

4.1. Certificate configuration

1. If a self-signed certificate is configured on TrueConf Server, download it via the link **Download ca.crt** in the **Self-signed certificate** section and add it to the trusted root certificates on the machine with TrueConf Border Controller. Check the documentation for your OS to learn how this can be done.

For example, **on Debian**:

- Copy the certificate file to the certificate storage in the directory `usr/local/share/ca-certificates/` :

```sh
sudo cp ca.crt /usr/local/share/ca-certificates/
```

- Update the certificate storage with this command:

```sh
sudo update-ca-certificates -v
```

* If there is an error message indicating that the command was not found, install its package from the repository:

```sh
sudo apt install -y ca-certificates
```

- To check if your OS trusts the certificate, run this command:

```sh
openssl verify /usr/local/share/ca-certificates/ca.crt
```

2. In the TrueConf Server control panel, go to the **Web →Settings** section and specify the address of the machine with TrueConf Border Controller in the **External address of TrueConf Server web** field.

3. Create a certificate for the machine with TrueConf Border Controller. If you do not have a commercial certificate, you can create a self-signed certificate as it is described in our knowledge base.

4. Copy the certificate and key obtained at step 3 to the directory `<path_to_border_controller>\etc\crt\` where `<path_to_border_controller>` is the path to the executable file of the component.

5. Rename the certificate and key files as `<guid>.crt` and `<guid>.key` where `<guid>` is a 128-bit GUID identifier which will be the same for both files. It can be generated with the help of the online service UUID Generator .

4.2. Creation of the configuration file

Create the configuration file `webproxy.toml` in the directory where the executable file of the component is stored:

```
[certificate]
cert_extension = '.crt'
key_extension = '.key'

[dir]
executable_relative = true
installation = 'C:\TrueConf Border Controller'

[file]
configname = 'webproxy'

[interfaces]
[interfaces.list]
[interfaces.list.0]
Address = '[::]:443'
EnableTLS = true
ReadTimeout = 0
TLSConfigID = 'd25ceb20-f473-41dc-8db9-37f4dec1a3d1'
TargetID = 'a824b5cb-c92d-4a52-a5cc-434fecaba6a8'

[interfaces.list.1]
Address = '[::]:80'
EnableTLS = false
ReadTimeout = 0
TLSConfigID = ''
TargetID = '2f0dbf86-8378-41fc-9c5a-89a43728a0b7'

[proxy]
trust_client_headers = true

[targets]
[targets.list.a824b5cb-c92d-4a52-a5cc-434fecaba6a8]
address = '10.110.2.82:443'
is_secure = true

[targets.list]
[targets.list.2f0dbf86-8378-41fc-9c5a-89a43728a0b7]
address = '10.110.2.82:80'
is_secure = false

[tls]
[tls.list]
[tls.list.d25ceb20-f473-41dc-8db9-37f4dec1a3d1]
CertificateID = 'd25ceb20-f473-41dc-8db9-37f4dec1a3d1'
CertificateType = 'user-provided'
DisplayName = 'My TLS configuration'
ID = 'd25ceb20-f473-41dc-8db9-37f4dec1a3d1'
```

where you have to specify the following values:

- in the `[dir]` section:

- `installation` — the path to the executable file of the component
- in the section `[interfaces.list.0]`:
  - `Address` — HTTPS port if it is different from the standard **443**
  - `TLSConfigID` — the name of the certificate and key files received at step 5
  - `TargetID` — GUID for identifying a block of HTTPS settings from the `[targets]` section
- in the section `[interfaces.list.1]`:
  - `Address` — the port for accessing the control panel via HTTP if the port is different from the standard **80** port
  - `TargetID` — GUID for identifying a block of HTTP settings from the `[targets]` section
- for each `[targets.list.<guid>]` blocks in the `[targets]` section:
  - generate unique GUIDs and add them instead of `<guid>`
  - `address` — IP address or FQDN ofTrueConf Server and the port for the transfer of traffic from the component
  - `is_secure` — the value is equal to `true` if an HTTPS port was specified for the `address` parameter of the current `[targets.list.<guid>]` block ; otherwise it is equal to `false`
- in the `[tls]` section:
  - for the `[tls.list.<guid>]` block name, replace `<guid>` with the `TLSConfigID` value (it is also the name of the certificate file from step 5)
  - `CertificateID` and `ID` — value of `TLSConfigID`.

7. Save the file `webproxy.toml` and run the component.

4.3. Starting the component on Windows OS

To launch the component from the console, execute the following command:

```sh
<path_to_border_controller> run
```

where `<path_to_border_controller>` is the path to the executable file. For example:

```sh
c:\Program Files\TrueConf\webproxy_windows_amd64.exe run
```

Creating a service is similar to the instructions discussed for the **tc_bc** component, only you need to specify `c:\Program Files\TrueConf\webproxy_windows_amd64.exe run` as the file path (parameters `BinaryPathName` or `binPath`).

4.4. Starting the component on Linux OS

To launch the component from the terminal, execute the command:

```sh
<path_to_border_controller> run
```

where `<path_to_border_controller>` is the path to the executable file. For example:

```sh
/opt/trueconf/enterprise/etc/bc/webproxy_linux_amd64 run
```