

Configuration Templates

<https://campus.barracuda.com/doc/96771950/>

The Barracuda Firewall Control Center is a central administration unit designed to manage a large number of CloudGen Firewalls and Secure Connectors. After several years of successful operation, experience from various application fields has shown that the number of managed devices has grown steadily. This new situation demanded a new solution to configure, deploy and manage even larger and more complex configurations with increased comfort and less time effort.

Several configuration tools (Firewall Admin, ConfScript, REST, Web-UI) are in productive use for the configuration but have different technological approaches. Large setups often have the same configuration redundantly, the managed boxes only differ in a small set of parameters. For an administrator, this causes additional overhead for planning, deploying, and managing the setup of the affected devices.

Configuration Templates are a new tool for creating and maintaining configurations for firewalls and FSCs in a new way, with a special focus on scalability and automation. The tool is available for the Control Center and CG firewalls started with release 8.2.

Configuration templates cover the requirement of

- managing common configuration information for deploying and managing large-scale configurations while still
- giving maximum freedom for individually configuring parameters which makes similar configurations easily distinguishable from each other.

Configuration Units

The configuration template concept is based on functional configuration units. Each unit is a functional block of its own, which handles model and release-specific settings and describes a certain feature in an abstract way. This property allows configuration templates to be released and model-independent. Every configuration unit has a number of required and optional parameters, which are used to calculate the final configuration of a firewall or FSC. The configuration template is responsible for correctly parametrizing those inputs. This is done by wiring input parameters, calculated variables, and static default values to the configuration unit's input parameters.

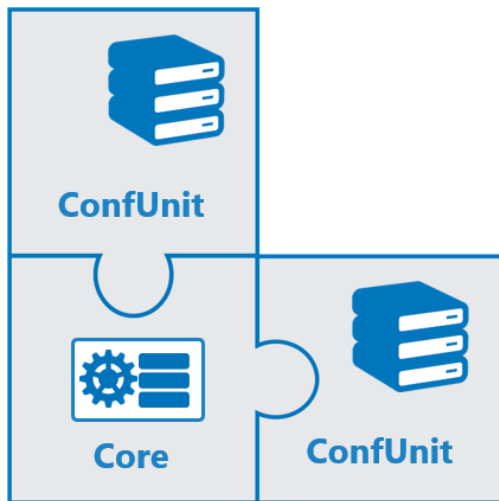
For more details on the properties of CGF related configuration units, see [CGF Configuration Units](#).

For more details on the properties of SC-related configuration units, see [SC Configuration Units](#).

Configuration Templates

Configuration templates are the blueprint to instantiate a large number of instances, which currently can be CG firewalls and FSCs. Such a template can be thought of as a choice of available configuration units.

Mixing units for different products (CGF vs. FSC) in the same configuration template is currently not possible.



The fundamental and most convenient way to create and edit templates is the graphical editor in Firewall Admin. TDL, and JSON are equivalent formats to define a configuration template.

For more information on how to manually work with `tdltool`, see [Template Definition Language - TDL](#).

References in Templates: Parameters, Variables, and Objects

Configuration Templates support parameters, variables, and objects.

Parameters can be defined in each template. Each parameter has a type and can have a default value. A reference to such a parameter can be used as input to a configuration unit. Each instance of a template has to provide values for each parameter used by the template, if a parameter has a default value the definition of a value per instance is optional. The main purpose of parameters is to provide a mechanism of defining instance-specific values for configuration unit input parameters. If Configuration unit input parameters are defined statically, the same value is used for every instance.

Variables are a mechanism to derive values from parameters by using simple functions. An integer parameter could be used in an IP variable as an offset in a network to calculate an IP address and could be used in a string variable to extend a string with the number.

In 9.0.0 it becomes possible to also use network objects in configuration templates. The network objects have to be defined in the usual way and then can be referenced as input to suitable configuration unit parameters, e.g. parameters of type IP. The new **Pool Object** provides the automated allocation of VIPs and MIPs if respective IP addresses must be allocated from predefined networks.

On the level of the configuration template, configuration mismatches are partially verified. Full verification is done on the instance level, where release- and model-specific checks can be taken into account, e.g., it can be determined on a configuration template level if a port 'P1' is valid in general, but it can only be verified on an instance level if port 'P1' exists for the actual model.

When the editing of a configuration template is done, all parameters, variables, and objects are logically checked for plausibility and completeness. If no errors are found, the template can be saved and later used in real-world applications.

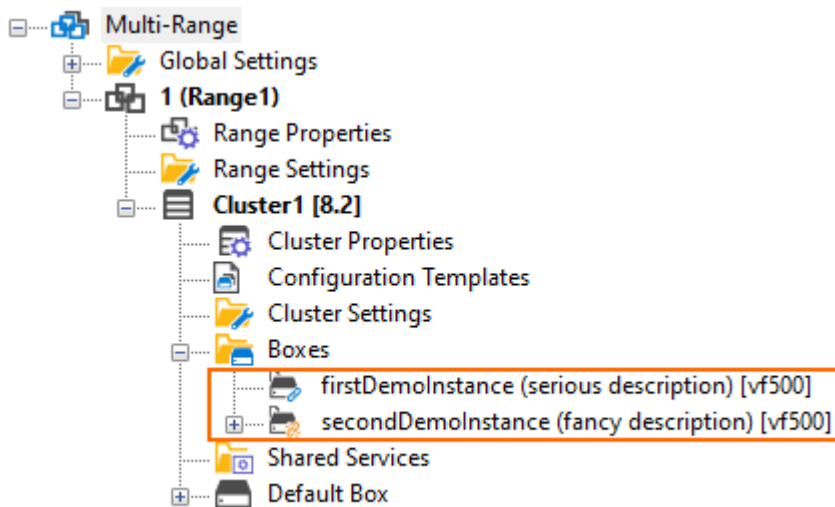
Instances, Bindings, and the Relationship between Functional Boxes and Instances

Instances

A **Configuration Template** instance represents the full configurations of a CGF or FSC.

Managed CGF configurations created in the configuration tree of the control center can be transformed into configuration template instances and vice versa. Each instance belongs to one specific configuration template and changes to the configuration template will be propagated to all of its instances. FSCs do not have a representation in the configuration tree and can only be managed by configuration templates. The legacy FSC editor should not be used for new configurations.

After an instance has been created from a configuration template, it will show up in the configuration tree of the Control Center as a node for a managed CGF.



These two device nodes look different from the standard device nodes indicating that they are configuration-template instances.

Instances can also be accessed in read-only mode in the configuration tree thus enabling users to inspect the content of a certain instance.

Converting an Instance into a Template

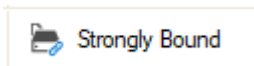
After creating an instance from a template, the instance can be modified to match your individual requirements. Your modifications can at a certain point become a new initial point where you want to use the state of the instance for creating further new instances.

You can achieve this by converting an instance into a template which you can then derive other new instances from. For this, you must right-click a certain instance in the list of instances and select **Create Box from Instance** from the list displayed in a pop-up menu. After naming the new template, it will be saved and will be ready to derive new instances.


Bindings

A binding is a virtual relationship between an instance and the configuration template it is based on. This relation determines how instances are managed after being derived from the template. The user can choose from two types of binding: strong or weak binding.

- **Strong binding** - Strong binding indicates that the instance is fully managed by the underlying configuration template framework. Direct changes of the configuration parameters by the user are not possible. This is the safest way of managing large amounts of managed devices in the Control Center.
Strongly bound relations are indicated by the following icon in the instance list view of the Configuration Template Manager:



Strongly bound boxes do not have a sub-tree in the Control Center's configuration tree in order to prevent manual configuration. This protects all subnodes from manual modification.


 firstDemoInstance (serious description) [vf500]

- **Weak binding** – Weak binding enables classical manual configuration in the configuration tree without Configuration Template Manager. However, the price for this option is a potential for configuration conflicts between the automated and the manual configuration.

Weakly bound relations are indicated by the following icon in the instance list view of the Configuration Template Manager:



Weakly bound boxes are displayed in the Control Center's configuration tree showing an expandable box node. This gives the user access to change all subnodes interactively.

 secondDemoInstance (fancy description) [vf500]

The default mode for Configuration Template Manager is strong binding.

Bringing it all together in the Configuration Template Manager Window

In order to see how to work with templates, instances, bindings, and the TDL, see [Configuration Template Manager](#).

Figures

1. puzzle_01.png
2. confTemplate_config_tree_with_device_nodes.png
3. icon_strongly_bound.png
4. config_tree_box_node_strong_binding.png
5. icon_weakly_bound.png
6. config_tree_box_node_weak_binding.png

© Barracuda Networks Inc., 2024 The information contained within this document is confidential and proprietary to Barracuda Networks Inc. No portion of this document may be copied, distributed, publicized or used for other than internal documentary purposes without the written consent of an official representative of Barracuda Networks Inc. All specifications are subject to change without notice. Barracuda Networks Inc. assumes no responsibility for any inaccuracies in this document. Barracuda Networks Inc. reserves the right to change, modify, transfer, or otherwise revise this publication without notice.