Multitenancy

It is a common scenario that a flexVDI platform is used to provide Desktop as a Service (DaaS) to the different customers of the platform's owner. In this scenario, resources, network connections and authentication domains must be isolated for each customer. flexVDI supports isolating these three factors independently for each customer. For instance, resources can be isolated per customer while using the same network configuration for all or many of them, or customers may have different authentication domains while sharing the resources of the platform.

Isolate resources

Resource isolation is implemented with *Pools*. As explained before, *Pools* aggregate the resources of a group of *Hosts* so that the resources needed by a *Guest* when it is started are allocated automatically to the most suitable *Host*. *Pools* isolate resources in the following ways:

- The resources that a *Pool* needs are reserved **statically**. Once they have been reserved, the *Pool* will have exclusive access to them, and only them, until a **rebalance** operation is performed. This also means that a *Pool* may get less resources than it needs when there are not enough available resources. Trying to get additional resources would mean stealing them from another *Pool*.
- The group of *Guests* that are assigned to a *Pool* will never use more resources than those reserved by the *Pool*. There is no way to lead another Pool to starvation.
- A *Pool* will get its resources only from the set of *Hosts* it is configured to. In this way, it can be limited to a subset of the *Hosts* of the platform. Some Guest Operating Systems' licensing for DaaS (e.g. Microsoft Windows) mandate that physical *Hosts* are assigned exclusively to a single client.
- *Pools* have a priority to decide in which order they get resources with a rebalance operation is performed. In case of a *Host* failure, this may result in the *Pools* with lower priority being assigned last and not getting all the resources they need. Use priorities that match your client's SLAs.

Use *Pools* to isolate resources dedicated to each customer. You can create one *Pool* per customer, or create many *Pools* per customer to assign different priorities to each group of *Guests* (e.g. higher priority to services, lower to desktops).

Isolate network connections

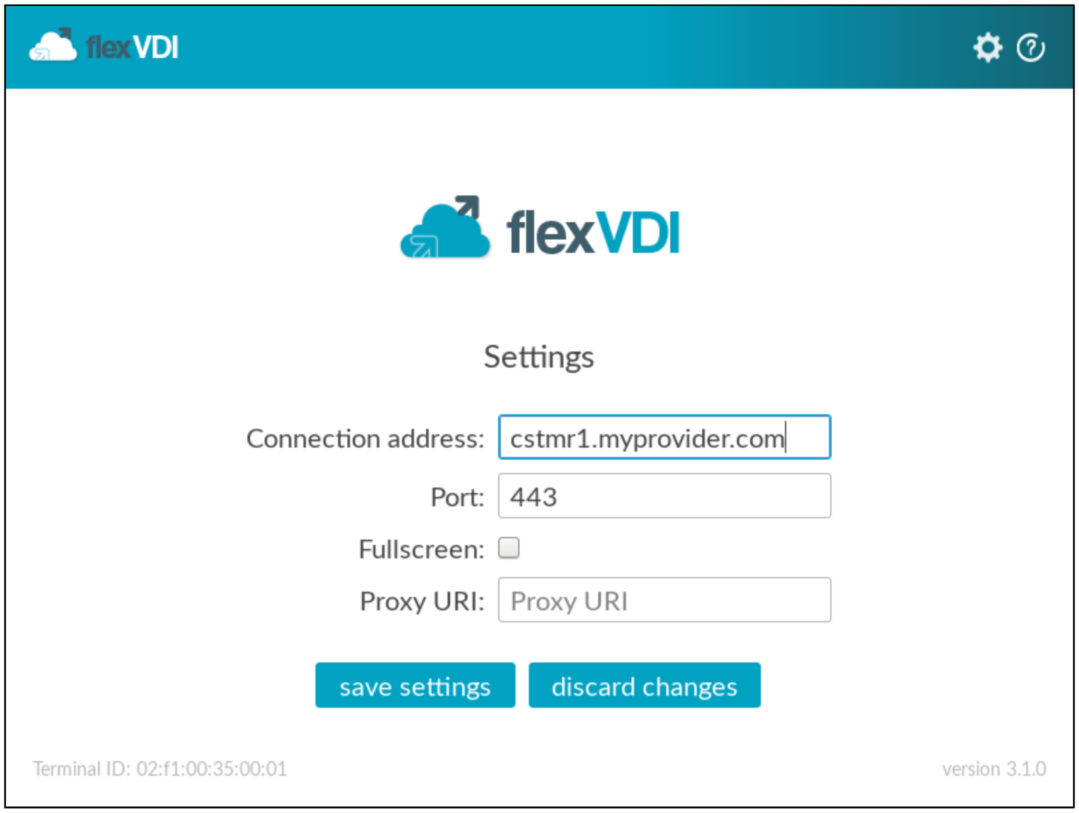
flexVDI networking model supports connecting *Guests* to multiple virtual bridges and VLANs, to match the level of network isolation you have in the rest of the data center. You can isolate the network connection of a group of *Guests* in the following ways:

- Create a virtual bridge for each customer. Create one or more Guests with a virtual interface in each bridge that will act as gateway/firewall to route traffic between each customer's subnet and the outside.
- Connect guests to the same virtual bridge but on different VLANs. Traffic is effectively separated, because the virtual bridge only sends *Guests* the traffic they are allowed to see. In this case, VLANs can be managed from outside the flexVDI platform, so no additional network services are needed.

Authentication domains

In a simple scenario, with only one customer, the recommended way to configure VDI authentication is to create an authenticated default Terminal Policy and additional non-authenticated Terminal Policies. In this way, only the terminals that require non-authenticated access to the platform are manually registered to a Terminal Policy. The rest will be automatically registered to the default Terminal Policy, and require authentication. However, in this case, there is only one authentication domain. If you add additional authenticated Terminal Policies for other domains, terminals must be manually registered to them.

To provide a solution for this problem and a more general case, flexVDI supports registering a terminal to a Terminal Policy automatically by connection name. It works in the same way as virtual web servers. First, create a different DNS entry for each authentication domain that points to your platform's IP address, e.g. `cstmr1.myprovider.com`, `cstmr2.myprovider.com`, ... Then, create an authenticated Terminal Policy for each of these names, e.g. `cstmr1.myprovider.com`, `cstmr2.myprovider.com`, ... Finally, configure them with the appropriate authentication parameters (LDAP server, branch name, bind user, ...). Whenever a new terminal connects to the platform and it is not registered yet to any Terminal Policy, flexVDI will first look for a Terminal Policy which name matches the host name that was used to establish the connection. So, provide each of your customers with the DNS name that represents their authentication domain:

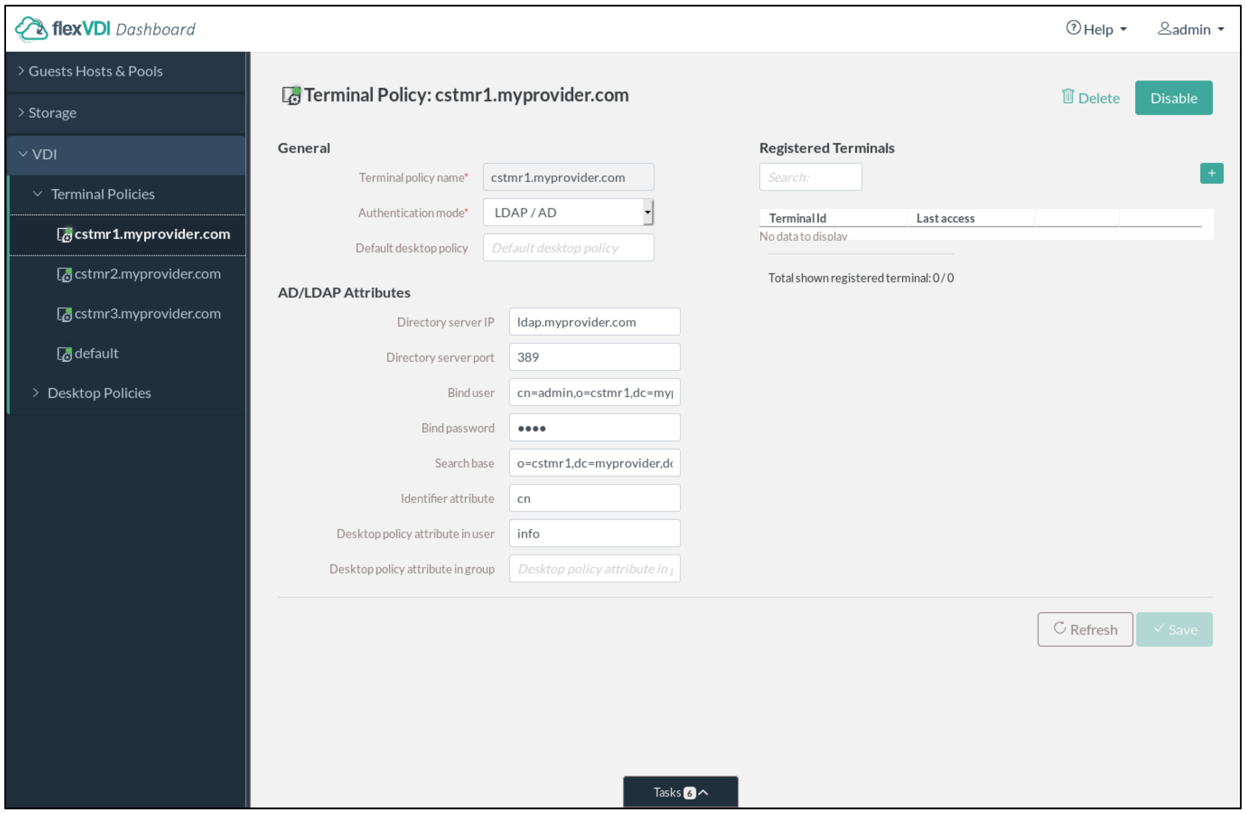


Settings

Connection address:	cstmr1.myprovider.com
Port:	443
Fullscreen:	☐
Proxy URI:	Proxy URI

Terminal ID: 02:f1:00:35:00:01

version 3.1.0



Terminal Policy: cstmr1.myprovider.com

General

Terminal policy name*	cstmr1.myprovider.com
Authentication mode*	LDAP / AD
Default desktop policy	Default desktop policy

AD/LDAP Attributes

Directory server IP	ldap.myprovider.com
Directory server port	389
Bind user	cn=admin,o=cstmr1,dc=myp
Bind password	••••
Search base	o=cstmr1,dc=myprovider,dc
Identifier attribute	cn
Desktop policy attribute in user	info
Desktop policy attribute in group	Desktop policy attribute in g

Registered Terminals

Search:

Terminal Id	Last access

No data to display

Total shown registered terminal: 0 / 0

This model works for non-authenticated Terminal Policies too, so you may have `vdikiosks.myprovider.com`, and have your kiosk devices connect to the platform using that address to be automatically registered with a non-authenticated Terminal Policy named `vdikiosks.myprovider.com`. However, take into account that, in this case, any user may access a non-authenticated desktop just knowing the right domain name.