The ultimate Citrix XenDesktop 7.x internals *cheat* sheet

Version 2.0

Table of Contents

Up from version 1.0

About two and a half years ago I published the first version of the ultimate Citrix XenDesktop 7.x internals cheat sheet and it turned out to be a big hit. In the meantime, it has been viewed over 80.000 times already. All the more reason to start working on version 2.0. Since I have been writing about Citrix technologies for the last couple of years I have built up a broad archive, which I can now partly (re) use and re-write to come up with an even more detailed edition, version 2.0 of the 7.x internals cheat sheet.

By the numbers

Just so you know what you'll be getting yourself into, here are some statistics on what's ahead. This document is 71 pages long, consists out of 16 (real) chapters, 40, so-called FMA facts and 24558 words in total, which goes beyond the concept of a cheat sheet, I know. I have reused and rewritten some material form my book: **Inside** Citrix: The FlexCast Management Architecture (*If you haven't picked up a copy, make sure to do so here)* as well as multiple blogposts that I have written throughout the last couple of months.

First things first

Building upon version 1.0 of the ultimate XenDesktop 7.x internals cheat sheet, version 2.0 includes much more information on the FMA's main components and core services, including detailed info on the desktop and server VDA's, StoreFront, Receiver, LHC, ICA/HDX and more. As an added bonus, I've also thrown in one of my most popular blogposts on the Citrix NetScaler.

As mentioned, I have included multiple sections of my book and reused and re-written a couple of blogposts that I published throughout the last 8 to 10 months or so, including some fresh new content here and there. Before we get to the user authentication, resource enumeration and launch section (the 'putting it all together' chapter) I'd first like to discuss, in detail the main components that make up the FMA, including a detailed look at each of the 13 (at the time of writing) FMA core services.

Because I will be handling multiple key components there will be a small overlap in the information shared. Though I guess you guys won't mind to much. And hey, it's by repetition that we learn, right?! Let's start by having a closer look at the main components and services that make up the FMA.

Delivery Controller

The Delivery Controller is the real workhorse and centerpiece of the FMA and, as such, it has a lot of responsibilities. To name a few, it brokers (VDA) sessions, verifies user credentials, and plays an important role during user login and resource enumeration as well as launch. It communicates with StoreFront and/or Web Interface, the underlying Host Connection (Hypervisor or cloud-based services), the Central Site database, and it also takes care of load-balancing hosted shared desktop connections. As of version 7.12 it includes and takes care of the Local Host Cache as well, which will be discussed in more detail later.

***FMA fact**: Your Delivery Controllers can be considered as the heart of your FMA deployment.*

You could say that the Delivery Controller is in fact the heart of XenDesktop / XenApp. It houses all thirteen primary (at the time of writing) FMA services including the well-known XML service. Here they are:

1. Analytics service.
2. Broker service.
3. The Citrix High Availability Service
4. Configuration Synchronisation Service (LHC)
5. Configuration service.
6. AD Identity service.
7. Configuration Logging service.
8. Delegated Administration service.
9. Machine Creation service.
10. Host service.
11. Environment Test service.
12. Monitor service.
13. StoreFront service.

Each service has its own specific responsibility. Note that the XML service isn't mentioned since it is not FMA-specific. As highlighted, throughout the 'The main FMA services and interactions' chapter we will have a closer look at each service individually, the XML service included, and how they all interact and communicate within the FMA.

Authentication and enumeration

When a user logs in, either internally through StoreFront or externally through NetScaler, for example, as mentioned the Delivery Controller plays an important role during the user authentication and verification process, as well as with enumerating and launching user resources. This process is also referred to as connection brokering.

A Delivery Controller has a direct and live connection with the Central Site database, which holds all static as well as dynamic (real-time) information within the Site. As opposed to the Data Collectors in XenApp 6.5 and earlier, none of this information will be stored locally (at least that's how it worked up till version 7.12) and all Delivery Controllers will fully depend on the Central Site database to provide this information when needed. Communication between a Delivery Controller and the Central Site database is constant (heartbeat messages are exchanged every 20 seconds with a TTL of 40 seconds).

The Delivery Controller also plays a key role in controlling all registered desktop and server machines (to which your users connect) regarding availability, load balancing and power management, which includes starting and stopping virtual machines when needed.

It brokers connections between users and their virtual and/or physical desktops and applications while maintaining and optimizing these connections wherever and whenever needed using technologies like Session Reliability (Common Gateway Protocol), Auto Client Reconnect, ICA Keep-Alive messages, and workspace control. Note that power management is not available for physical machines, only virtual.

One is none

As an IT specialist, you are probably familiar with the saying 'one is none', as this applies to almost all infrastructural components that we 'techies' must deal with. The same applies to the Delivery Controller. If the server hosting the Delivery Controller role is unavailable, your users will not be able to be authenticated or verified; as a result they will also not be able to access and/or launch any of their virtual desktops or published applications. Therefore at least two Delivery Controller servers per Site should be deployed on different physical hosts (when virtualized) to prevent a single point of failure.

All online Delivery Controllers within your Site will actively participate in handling user / session requests at any time (amongst other tasks), if one of the Controllers for whatever reason goes offline, one of the other Controllers will take over its tasks automatically and instantly. All Controllers within a Site have access to the same Central Site database and therefore are equally configured.

***FMA fact**: Your environment is as strong as its weakest link. Make sure to apply the 'one is none' rule wherever and whenever it makes sense.*

A Delivery Controller is different from a Data Collector in many ways. Besides the absence of the Local Host Cache (again, when dealing with versions prior to 7.12) Delivery Controllers do not communicate with each other (this also changed as of version 7.12, see the LHC chapter for more detailed information on this), they cannot host any user sessions like a Data Collector can, and as such they also do not have to run the same Operating System as the VDAs they manage.

Citrix Studio is the main management tool and console used to set up and configure XenDesktop and XenApp Sites. It can be installed on a Delivery Controller or separately on a management machine, for example. Since the Delivery Controller plays such an important role, Studio will also have a direct connection with one or multiple of your Delivery Controllers.

The main FMA services and interactions

The release of XenApp/XenDesktop version 7.12 introduced couple of new FMA services (primarily used by LHC). As you might be aware, I have written multiple articles on the FlexCast Management Architecture in the past, talking about its core services, their responsibilities, capabilities, communication channels/interfaces and so on.

Throughout the past two years I also came up with a nice graphical overview (at least I like to think so) representing a Delivery Controller including all main FMA services. This chapter is meant to provide you with an update on the FMA and its primary services, graphical overview included.

Although FMA services run completely isolated from each other, internal communications between the different services takes place using so-called WCF (Windows Communication Foundation) end points (also referred to as service interfaces) over port 80 by default. Though, here I would like to note that port 80 can be changed into any port number you might prefer and that encryption is supported as well.

The endpoint address is represented by the EndpointAddress class, which contains a Uniform Resource Identifier (URI) that represents the address of the service (a secure identity including a collection of optional Headers). Each service has the ability to interact with all other services in order to carry out important tasks and actions throughout the FMA. As a best practice, you will deploy two or more Delivery Controllers, avoiding a single point of failure and providing scalability where and when needed.

As we will shortly see, there are four services that have a special place within the FMA and take on a more prominent role than the others (a.k.a. core services). These are the Broker, Configuration, Delegated Administration and the Configuration Logging service.

***FMA fact**: While all services closely interact with and depend on each other, at the same time they are also completely separated from each other. Each service is configured to communicate to the Central Site database using its own individual DB connection string. This way, if one service fails it will not affect any the other services.*

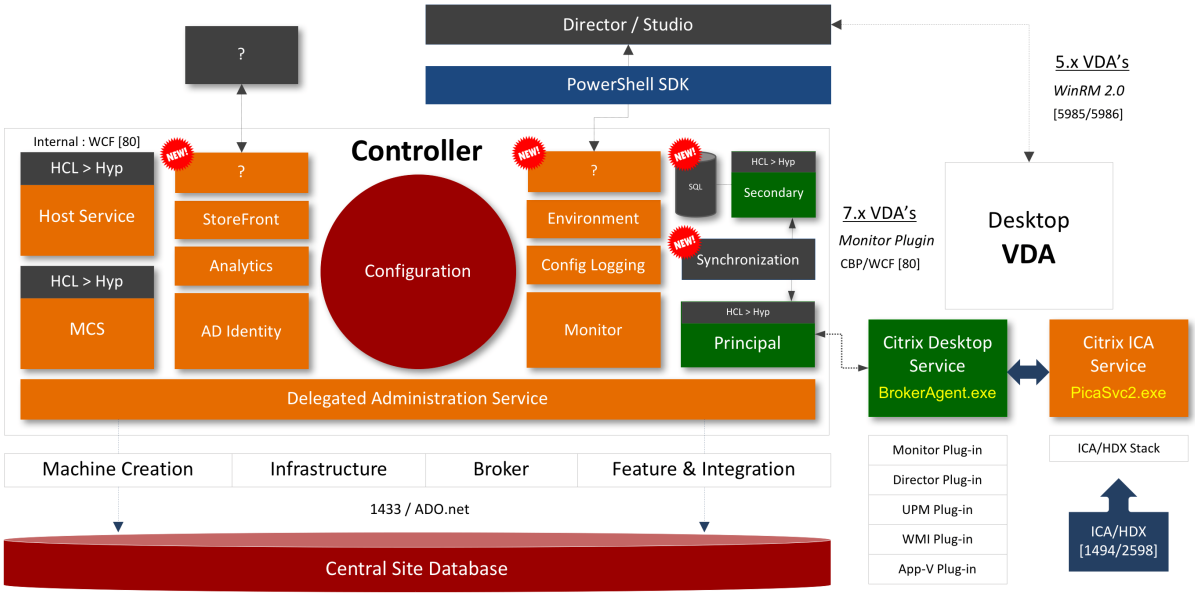
One of the biggest differences between the IMA and FMA is that with the FMA the Delivery Controller is only responsible for managing and brokering connections to managed as well as unmanaged VDAs installed on your server and desktop machines. It doesn't host any sessions of its own.

Also, all the ICA / HDX bits, bytes and related services (controlling the HDX functionality throughout a session) reside on the VDA itself, again relieving the Delivery Controller from any additional tasks. Because of this approach, it is also easier to maintain the code and different operating systems (server and desktop) can be deployed within the same site.

***FMA fact**: Keep in mind that if you change something for one specific service, like the DB connection string, for example you must do this for all the other FMA services as well.*

The FMA's thirteen, and then some

Let's go over each of the services one at the time. Where applicable I'll elaborate a bit more on the inner workings, any special considerations and/or relations to other services like the XML / STA Service, for example.



***FMA fact**: All FMA services run under the NT AUTHORITY\Network service account. Also, when authenticating to the Central Site database (this is where the Configuration Service plays an important role as well) all services use the local computer account of the machine that they are currently running on.*

1. Broker Service

The broker (XML) Service is probably the best-known one. By the way, it's indicated as Principal and Secondary (in green) on the image above, with the Synchronisation Service in between. From a Delivery Controller point of view, it brokers new and manages existing sessions, handles resource enumeration, the creation and verification of STA tickets, user validation, disconnected sessions, and more (see the list under 1.1 as well). From a VDA point of view it takes care of all communication to and from the Delivery Controller. It does this by communicating with the VDA Citrix Desktop Service, a.k.a. BrokerAgent.exe, which is part of the desktop as well as server VDA.

Note that the STA (Service) is also part of the Broker Service, and has been as of Presentation Server 4.0. Before that it was written as an ISAPI extension for Microsoft Internet Information Services, or IIS. As of XenDesktop 4.x the XML service (ctxxmlss.exe) has been rewritten in .NET and became part of the Broker Service as well. T

he Broker Service is built up of three separate services (or four if you also count the newly introduced Principal Broker Service, see below) all handling different tasks: it brokers connections, it enumerates resources, it can take care of authentication (token) and it acts as the Secure Ticket Authority, generating and validating STA tickets; however, this only applies to resources launched through a NetScaler, and last but not least, as of version 7.12 it is also involved in managing the Local Host Cache.

As of XenApp/XenDesktop version 7.12 the Local Host Cache (LHC) got re-introduced (though it's new to the FMA). One of the main LHC components is the Broker Service, however, when the LHC is involved it is also referred to as the Principal Broker Service (as you will find out shortly there is also a High Availability Service a.k.a. the Secondary Broker Service).

Think of it as a sub-service, just like the XML Service, for example. The Principal Broker Service will accept connection requests from StoreFront and it communicates with the Central Site Database just like before — brokering connections, taking care of load balancing and so on, while it also (continuously) interacts with the Configuration Synchronisation Service as well as the High Availability Service when the LHC becomes active.

1.1 Broker Service main responsibilities

As mentioned, the Broker Service has some huge responsibilities within the FMA. Besides some of the tasks already highlighted, one of its most important tasks is the registration of all VDAs, including ongoing management from a Delivery Controller perspective. To give you a complete overview of the main tasks and responsibilities of the Broker Service, have a look below:

1. As already mentioned it takes care of VDA registration, resource allocation, connection brokering, licensing enforcement, amongst other tasks.
2. During the initial user logon process it will validate the end-user's identity, based on credentials received through StoreFront.
3. Based on your configuration, it can take care of XML based user authentication (token creation).
4. Functions as the Principal Broker Service when LHC is enabled.
5. It continuously interacts with Configuration Synchronisation Service.
6. It is involved in HDX policy management.
7. It manages the overall power state of desktops and server machines when run virtually, starting and stopping VMs based on usage and on administrator configuration, including idle pool management.
8. It temporarily stores user credentials to allow users to be logged into virtual desktops without having to re-enter credentials (single sign-on).
9. It keeps track of virtual desktop state, based on information received from virtual desktops. As such, it will take appropriate action when needed.
10. It will participate in the initial load-balancing process, deciding which desktop or server to connect to. This information will eventually end up in an ICA launch file.
11. Administrators will use the Broker Service, although they may not actually know it to log off sessions, define Machine Catalogs and publish virtual desktops based on computer identity.
12. It exposes the Hypervisor state and alert information.
13. It will also handle all power management features, including but not limited to: power policy rules, reboot schedules and cycles, pool/buffer size management, and remote PC wake on LAN.

1.2 The Broker Service Site services

The Broker Service is somewhat special in that it also houses multiple so-called Site Services, let me explain. Site services provide Site-wide maintenance and housekeeping functionalities within and a XenDesktop Site. They take care of things like managing connections to your Host Connections, checking up on session idle times, managing reboot schedules, cache maintenance (refresh) and more.

Before I go any further you need to know that there are eighteen Site services in total, with each having its own responsibility, or multiple in some cases and that they are part the Broker

Service. More specifically, each individual Site service will only run on one of your Delivery Controllers (within a Broker Service), creating a distributed model. As soon as a Delivery Controller misses a *heartbeat with the Central Site database, all Site services running on that Delivery Controller will be transferred to one of the other active and still considered healthy Delivery Controllers. Again, they will be moved from one Broker Service to another.

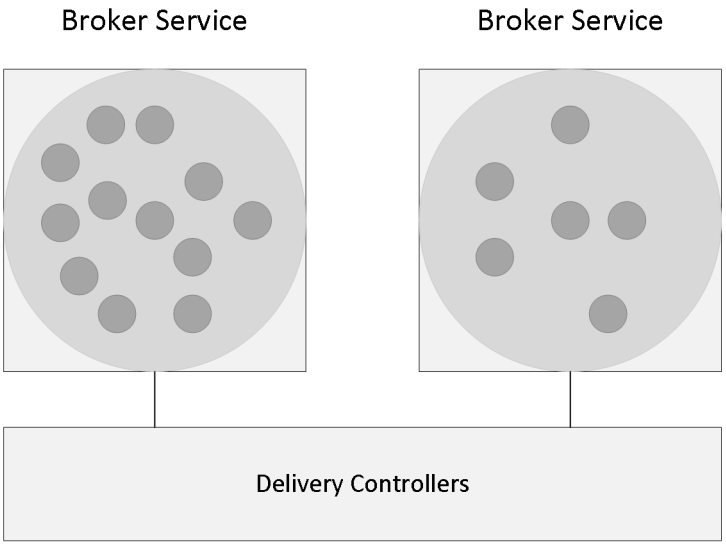
*A heartbeat message is exchanged between a Delivery Controller and the database every 20 seconds with a default timeout of 40 seconds.

***FMA fact**: While it is considered a best practice to keep all Delivery Controllers equally configured, Site services are the exception to the rule, so to speak.*

At runtime, when a Delivery Controller becomes active the Site services will automatically be divided between all active Delivery Controllers within your Site. Do note that although you as an Administrator can assign certain Site services to a Delivery Controller if you want or need to – this is not a recommended or supported approach. The election mechanism is controlled by the contents of the Central Site database, the FMA will take care of this for you.

The eighteen Site services are:

1. ControllerReaper.
2. ControllerNameCacheRefresh.
3. Licensing.
4. BrokerReaper.
5. RegistrationHardening.
6. WorkerNameCacheRefresh.
7. AccountNameCacheRefresh.
8. PowerPolicy.
9. GroupUsage.
10. AddressNameResolver.
11. RebootScheduleManager.
12. RebootCycleManager.
13. ScopeNamesRefresh.
14. FeatureChecks.
15. RemotePC.
16. IdleSessionManager.
17. LeaseReaper.
18. Hypervisor connection.



2. The Citrix High Availability Service A.k.a. Secondary Broker Service

I already briefly mentioned the Secondary Broker Service/High Availability Service while referring to the Principal Broker Service a couple of paragraphs back. Together with the Configuration Synchronisation Service (see below) all three services reside on every Delivery Controller in your environment — assuming you are running version 7.12 or later.

When an outage occurs, the Principal Broker Service will no longer be able to communicate with the Central Site Database, as a result it will stop listening for any incoming StoreFront and/or VDA information and it will instruct the High Availability Service/Secondary Broker Service to start listening for incoming connection requests and handle them accordingly.

As soon as a VDA communicates with the High Availability Service/Secondary Broker Service a VDA re-registration will be triggered. This way the High Availability Service/Secondary Broker Service will receive the most current session information related to that specific VDA (who is connected to which machine, for example). In the meantime, while the High Availability Service/Secondary Broker Service is handling new and existing connections/sessions, the (Principal) Broker Service will continue to monitor the connection to the Central Site Database.

As soon as it notices that the connection to the Central Site Database has been restored it will instruct the High Availability Service/Secondary Broker Service to stop listening for, and handle new and existing connections/sessions. From this point on it will resume brokering operations as before, basically repeating the abovementioned steps of VDA registration to get up to speed with the latest connection/session information.

Finally, the High Availability Service/Secondary Broker Service will remove any remaining VDA registrations and will again continue to update the local SQL Express database (together with the help of the CSS) with any configuration changes from that point on, as highlighted before.

3. Configuration Synchronisation Service (LHC)

Every two minutes the Principal Broker Service will be checked for configuration changes. If a configuration change has been detected it will be copied over, or synchronised to the High Availability Service/Secondary Broker Service. This is the primary task of the Configuration Synchronisation Service (CSS). These configuration changes include but are not limited to published icons, changes to Delivery Groups and Catalogs, certain Citrix policies and so on. It will not include information about who is connected to which server (Load Balancing), using what application (s) etc. this is referred to as the current state of the Site/Farm, which is not considered a configuration change.

All (synchronised) data is stored in a Microsoft SQL Server Express LocalDB database (notice the LocalDB part, this is different from SQL Express), which resides on the same Delivery Controller. In fact, each time new information is copied over, the database will be re-created entirely. This way the CSS can, and will ensure that all configuration data stored in the Central Site Database will match that of the data stored in the local SQL Express database keeping the LHC current.

As a second prime responsibility, the Configuration Synchroniser Service will provide the High Availability Service/Secondary Broker Service (s) with information on all other controllers within your Site (Primary Zone), including any additional Zones you might have configured. This way each High Availability Service/Secondary Broker Service will know about all the other available High Availability Services/Secondary Broker Services within your (entire) Site.

Communication between the various High Availability Services/Secondary Broker Services takes place over a separate channel based on an alphabetical list containing the FQDN's of the machines they (the services) currently run. This information is used to elect which High Availability Service/Secondary Broker Service (read: Delivery Controller) will take over within that specific Zone when the LHC becomes active because of a DB failure or another outage of some sort.

4. Configuration service

All FMA services need to register with the Configuration Service on start-up so that it knows they are all good to go. This is one of the main reasons why the Configuration Service has such a prominent role: it handles all inter-service communication within the FMA. The Configuration Service is the glue holding the FMA together.

Located at the centre of the FlexCast Management Architecture, it holds and manages a list of all FMA services, allowing them to advertise their WCF addresses, or end points (service interfaces), including the functionality that they provide. Only after a service successfully registers with the Configuration Service, when adding in more Controllers, after a reboot or

during Site creation, for example, will it become active and able to communicate with other FMA services.

Once all services have successfully registered themselves, the Configuration Service will share a listing of all active and registered services as being active Site members, including their main responsibilities, or capabilities and service (communication) interfaces.

As soon as an individual FMA service needs to communicate with one of the other FMA services it will first (need to) contact the Configuration Service to get a copy of the services listing mentioned earlier.

After an FMA service successfully queries the Configuration Service and the listing has been received, this information will be cached for five minutes. This is mainly to ensure that the system isn't being overwhelmed with service listing requests, preventing the Configuration Service from becoming a potential bottleneck. At this point the requesting service knows where to find the other services, what they are capable of (responsibilities) and how to communicate with them (end points/service interfaces).

As a side note, the Machine Creation Service and the Machine Identity Service both communicate through the Host Service to find out about the configuration and connections of the underlying Hypervisor/Cloud connection (Host Connections) if any, including the storage and network configurations needed for virtual machine provisioning. This information will be cached for one minute as opposed to the five minutes mentioned earlier.

FMA fact: *Each FMA service can query the Configuration Service to look up other services using the listing mentioned earlier. In short, service registration and communication are both reliant on the Configuration Service. It will also store configuration metadata for all services, relieving Active Directory.*

4.1 Permissions

The Configuration Service directory stores the Active Directory machine account identifier (SID) for each service that has successfully registered with it. At the same time, this information will also be stored in the Central Site database where it will be accessible to all Delivery Controllers including the services that they host. Only when the machine SID of the accompanying FMA service is listed, and known by the Configuration Service will communication between FMA services be possible.

When a service with an unregistered machine account contacts the Configuration Service for the services listing it will receive an access denied. The only exception to this is the 'Network service' account: it is always allowed. Viewing and validation the successful registration of FMA services can be done through the PowerShell SDK. Use the following syntax:

```
Get-ConfigRegisteredServiceInstance -InterfaceType sdk | select serviceaccount,
interfacetype, servicetype | format-table
```

FMA fact: *If you would like to refresh the cache of one of the FMA services (remember the five minutes), all you have to do is restart the accompanying Windows Service. The cache (services listing) is retrieved during service start-up.*

5. Configuration Logging Service

Monitors and logs all configuration changes made within a XenDesktop Site, including all Administrator activity. Depending on its configuration, no Site changes are possible when its database is unreachable, making it one of the four core services as mentioned earlier. The data itself can be stored within the Central Site Database or a separate database can be created, which would be the recommended approach. As of XenDesktop version 7.7 a separate location / database can be selected during the initial installation configuration process.

6. Delegated Administration Service

The Delegated Administration Service is also considered to be one of the FMA's more critical services. All other FMA services will need to communicate with the Delegated Administration Service to validate if they have all the proper permissions and/or rights needed to make the necessary changes to the Central Site Database. Next to this, it manages the configuration and administration of all delegated administrative permissions. Thus, if this service becomes unresponsive or unavailable site-wide configuration changes will not be possible.

7. AD Identity Service

Handles all Active Directory computer accounts (identities) related to XenApp / XenDesktop virtual and physical machines.

8. Machine Creation Services

Handles the creation of new virtual machines. When this service is unavailable no additional virtual machines can be created, at least not using MCS. Also note that MCS is only capable of provisioning/creating virtual machines, not physical. If you want to be able to 'service' physical machines you will need to use Provisioning Services.

FMA fact: *If you do not configure a Host Connection within Studio, when creating a new Device Catalog, the option to use MCS as a provisioning mechanism will not be available (greyed out).*

9. Host Service

Manages all connections between the physical hosts, the Delivery Controllers and the underlying Hypervisor (s) or Cloud platform (s), both referred to as Host Connections. As far as the Hypervisor goes this can be either vSphere, XenServer, Hyper-V or the Nutanix Acropolis Hypervisor. This is where your virtual server and/or desktop VMs live. Physical machines are still optional as well, but again you'll use PVS instead of MCS.

The following Cloud platforms are supported: Amazon Web Service, Microsoft Azure and CloudPlatform. As highlighted earlier, the Host Service is also responsible for discovering and managing the connections and configurations of the Hypervisor, network and storage that are required and used by machine provisioning operations. It does the same for any cloud platforms/connections you may have set up.

9.1 Hypervisor Communications Library

The HCL is used by several FMA services, the Broker, Host and MCS to be a bit more precise to provide an abstract Application Programming Interface, or API for interacting with the underlying Host Connection, or multiple. This will ensure a consistent and consequent representation of the configured Host Connection including network and storage resources.

As an example, while multiple Hypervisors are supported this also adds to the complexity of the code, this is where the HCL comes in, it functions as an abstraction layer. As a result, when a new version of a Hypervisor is released or an existing one needs to altered, Citrix can quickly add support without the need to replace the code in multiple places. Again, the same applies to Cloud platforms as well.

10. Environment Test Service

Takes care of all Site-wide tests, initiated from Studio. You can run tests on your Delivery Groups, Machine Catalogs or even on your entire Site configuration, and more. New tests are introduced with every new version of XenDesktop/XenApp going forward.

11. Monitor Service

Monitors the overall FMA architecture and produces alerts and warnings when it finds something potentially wrong. These will pop up in Studio or Director. Note, however, that, although these alerts will tell us that something is potentially wrong, they won't tell us what is wrong or where to look for answers. Therefore, FMA services are best checked/monitored using PowerShell code typed in directly from a PowerShell Command Prompt.

For example, try using the Get-BrokerServiceStatus and/or Get-ConfigServiceStatus cmdlets to view the status of the Broker and Configuration service. There is a 'Get-' command for each FMA service.

12. StoreFront Service

This takes care of your StoreFront deployment, which can be added as well as managed directly from Studio. Note that your StoreFront can and probably will appear twice within Studio. Once under the console root as an integral part of Studio (it will be there by default) which is needed to be able to configure the Citrix Receiver as part of your published hosted shared desktop (s) images directly from a Delivery Group. And a second time (as a separate node) enabling you to manage your StoreFront deployment as if you were using the separate MMC based StoreFront management console. The latter gives you the option (s) to change its look and feel, authentication methods, adding and removing Stores, and more.

13. Analytics Service

The Analytics Service does as the name implies: it collects analytical data used by Director/Studio (custom reports, to name one). It is also leveraged by the Citrix Customer Experience Improvement Program (CEIP), which will be enabled by default — this applies to the Citrix Call Home functionality as well by the way. All data will be shared anonymously, encrypted and, as always, will be used for the greater good.

More to come

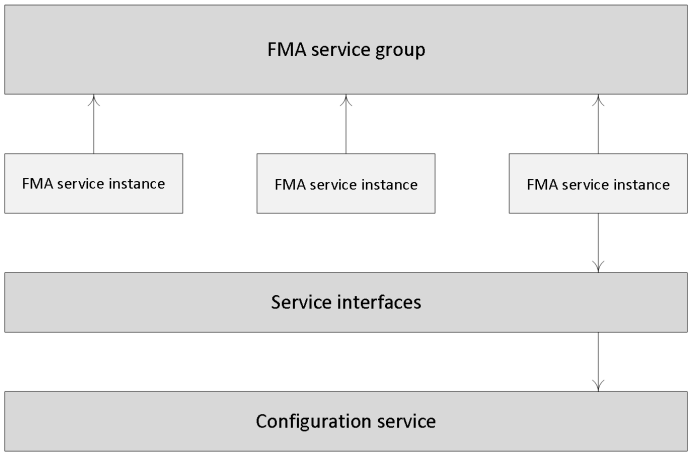
As you can see on the graphical overview, more services are on their way. Unfortunately, I can't share any additional information as all this is still under NDA. Just know that the services are already there, just not activated, usable and/or viewable.

Making the FMA services highly available

As we've established earlier, a XenDesktop Site consists of thirteen main services and each individual service, also referred to as a service instance, will run on every active Delivery Controller within your Site.

As a way to make these FMA services highly available, each service will need to register itself with a so-called peer service group, which contains all registered service instances of the same type. For example, if you have four Delivery Controllers you will also have four Broker services, of which only one will be active at a time.

All four Broker services will register themselves with the above mentioned service group (the Broker service, service group), and they will do so at startup. After registration is successful this information is written to, and stored in the Central Site Database so that all active Delivery Controllers including their services will have access to this information whenever needed. This way, if the active Broker service fails for some reason, one of the other Broker services can and will take over. This basically means that there will be a service group for every main FMA service, all thirteen of them, the same principle applies.



***FMA fact**: Each service group has a unique identifier, which can be queried using the PowerShell SDK if and when needed.*

The Desktop VDA

As mentioned in the 'The Virtual Delivery Agent' chapter, the VDA is a relatively small piece of software that gets installed on all virtual and physical machines running a Windows server and/or desktop operating system within a XenDesktop Site (note that there is also a Linux based VDA). It serves multiple purposes, as we will find out shortly, but only after a VDA is able to successfully register itself with one of your Delivery Controllers.

VDA registration

As soon as a Virtual Delivery Agent starts up, meaning the desktop or server Operating System boots, it will try and register itself with one of the Delivery Controllers known within the Site. For this to happen there needs to be a mechanism in place that tells the VDAs which Delivery Controllers are part of the same Site and how they can contact or reach them. For this, Citrix introduced the 'auto-update' feature, which will be enabled by default. It will keep all VDAs updated when Delivery Controllers are added or removed (go offline) from the Site. Each VDA maintains a persistent storage location to store this information. Also see the section named 'Desktop VDA ICA stack services' for some more detailed information regarding the VDA registration process.

When the auto-update feature is disabled, or does not supply the correct information, the VDA will check the following locations (in this order):

1. Through configured policies.
2. The ListOfDDCs Registry Key.
3. OU-based discovery (legacy).
4. The Personality.ini file created by MCS.

If a VDA is unable to register itself with a Delivery Controller or communication between the VDA and the Delivery Controller fails for any reason, you will not be able to connect to it.

Here I would also like to note that VDA registration is the Nr. one issue reported over at Citrix support, and as such deserves some extra attention.

This is also where the Citrix Desktop service, as part of installed VDA plays an important role. The Desktop service communicates directly with the Broker service over at the Delivery Controller and takes care of the initial VDA registration process through the Connection Brokering Protocol (CBP). The CBP is a collection of WCF (Windows Communication Foundation) end points defined to exchange information and handle the registration process.

***FMA fact**: Restarting the Citrix Desktop service on the VDA triggers the registration process and can be used to force re-registration when needed.*

Launching a VDI desktop

Let's have a closer look and see what happens when a VDI-based virtual machine is launched from a trusted, internal network. The XML service will again play an important role throughout the whole process.

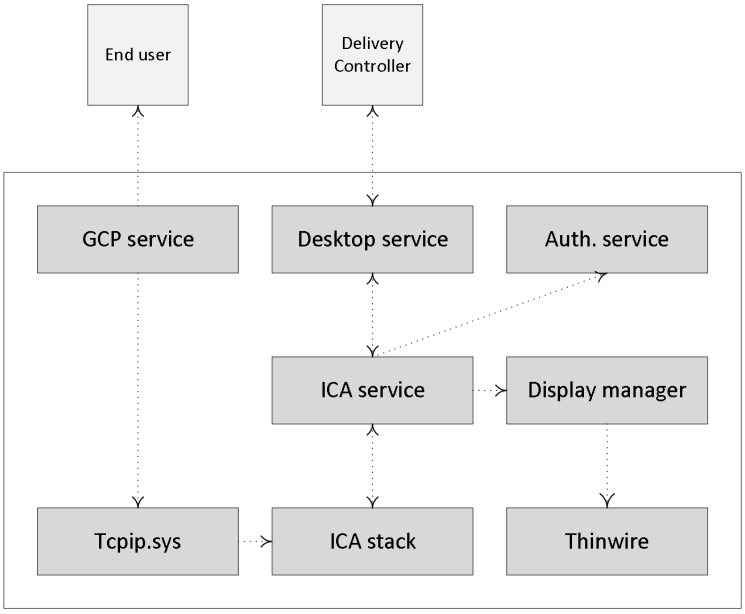
1. Let's assume that the VM is pre-subscribed and already present on the user's (StoreFront) home screen. Here it does not matter how we are connected: using a locally installed Receiver or using the Receiver for Web sites.
2. After the user clicks the desktop icon the StoreFront server will contact the Broker (XML/STA) service to check if any registered VDAs are available. It (the Broker service) does this by communicating with underlying Hypervisor platform through the Host service on the Delivery Controller.
3. If needed it will first start / boot a VM. It's not uncommon to pre-boot a few VMs, since, as you can probably imagine, this will positively influence the overall user experience. Understanding the usage patterns of your users allows you to boot enough machines before they're needed.
4. In between the VDA will register itself with the Delivery Controller handling the initial request. It will do so by leveraging the Connection Brokering Protocol (CBP) and communicating with the Broker service over at the Delivery Controller, as highlighted earlier.
5. Next the Delivery Controller, or Broker (XML/STA) service, will contact one of the VDAs and send a StartListening request. By default, the VDA isn't listening for any new connections on port 494 or 2595 until it gets notified that a user wants to connect.
6. As soon as the VDA is listening, the Broker (XML/STA) service will send this information back to the StoreFront server in the form of an XML formatted file (it's not an actual XML file).
7. Based on this information, the StoreFront server will generate a launch.ica file (it uses the default.ica file as a template) containing the IP address of the VDA and a whole bunch of other connection properties that are or might be needed. This is sent down to the user.

8. The locally installed Receiver (or HTML5-based Receiver) will read and autolaunch the launch.ica file, initiating a direct connection from the user's end point to the VDA.
9. The installed VDA will verify its license file with the Delivery Controller.
10. The Delivery Controller checks with the Citrix License Server to verify that the end-user has a valid ticket. A big change when compared to the IMA where every Session Host would communicate with the license server.
11. At this time, any applicable session policies will be passed on to the VDA and the session is launched.

What happens inside the VDA

Let's take it one step further and see what happens inside the VDA during launch time. The process below assumes that Session Reliability is enabled and that a desktop OS VDA gets launched as we've seen in the previous section. Remember that as of XenDesktop 7 there is also a server OS-based VDA, which we will have a closer look at in the next section.

1. The CGP service will receive the connection and sends this information on to the tcpip.sys, which will forward it to the ICA stack.
2. The ICA stack will notify the ICA Service a.k.a. the PortICA service (picaSvc) that a connection has been made after which the picaSvc will accept the connection.
3. Then the ICA Service will lock the workstation because the user needs to be authenticated to ensure that the user is allowed access to that particular machine.
4. As soon as the user logs onto the workstation, the PortICA service will communicate with the display manager to change the display mode to remote ICA, this request will be forwarded to the ThinWire driver.
5. In the meantime, the PortICA service will hand over the 'pre-logon' ticket data, which it received from the ICA stack, up to the Desktop service and from there back to the Delivery Controller in exchange for 'real' credentials.
6. The Desktop service receives the user's credentials, which are sent back to the PortICA service.
7. The PortICA service contacts the authentication service to log on the user.



Desktop VDA ICA stack services

If we have a closer at the Desktop VDA (check the main overview image) we can see that it consists of multiple FMA core services and plug-ins. The two main services are the Citrix Desktop Services a.k.a. BrokerAgent.exe and the Citrix ICA Service a.k.a. PicaSvc2.exe; together they control all HDX functionality for the duration of the session.

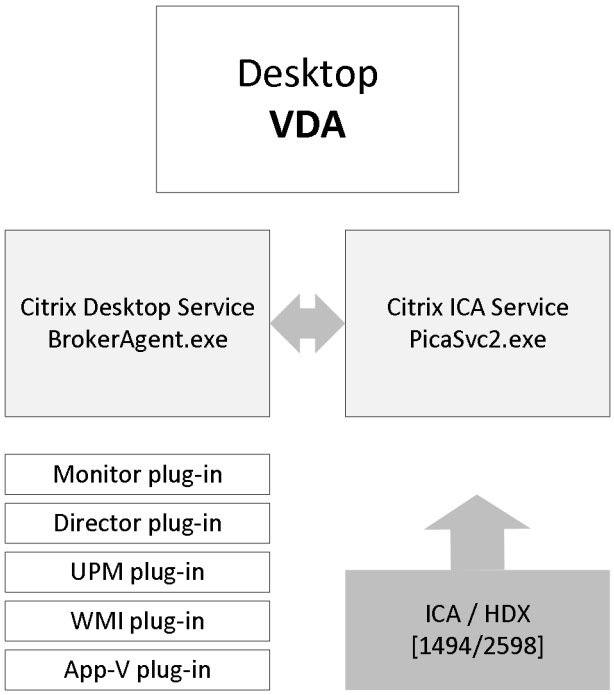
The Citrix Desktop service

The Citrix Desktop service communicates directly with the Broker service over at the Delivery Controller and, as highlighted earlier, takes care of the initial VDA registration process through the Connection Brokering Protocol (CBP), which is a collection of WCF (Windows Communication Foundation) end points defined to exchange information and handle the registration process. Once the VDA has been registered successfully the Citrix Desktop service will continue to communicate regularly with the Broker service on the Delivery Controller, also including a number of different feature plug-ins.

The Citrix ICA service

The Citrix ICA service implements the actual ICA protocol bits and bytes within the VDA, or the biggest part of it anyway. It will receive instructions through the Citrix Desktop service protocol coming from the Broker service over at the Delivery Controller. As soon as it is notified of a new connection request, and after AD authentication has taken place, and ticketing, licensing and HDX policy information has been successfully exchanged, the ICA stack will start listening for incoming connections allowing the launch request to complete.

Both Director and Studio also communicate with the underlying FMA services through WCF over port 80. Here it is also worth noting that when working with older VDAs (5.x and below) these will require WinRM listening to be enabled on port 5985 or 5986 to be able to communicate with Director and Studio. With 7.x VDAs this information is provided by the monitor plugin through the CBP protocol highlighted earlier.



The Server VDA

Ever since the introduction of XenDesktop 7, where the FMA took over and XenApp was integrated, a lot has been written with regards to its components, services, agents and so on. What surprises me though, is that the Server VDA is (almost) never mentioned, while this is, or at least was a brand-new component. Never before was it optional to install a (relatively) lightweight agent onto a XenApp server, it was basically all or nothing.

Although the (new(ish) FMA based Server VDA has been built from the ground up it still has a lot of similarities when compared to the 'old' ICA protocol stack deployed with XenApp 6.5 and earlier versions. However, unlike XenApp, the VDA (Virtual Delivery Agent) directly communicates with the Delivery Controller, it does this through the Broker Agent, basically the same way as we are used to with the desktop VDA (PortICA).

Before we move on…

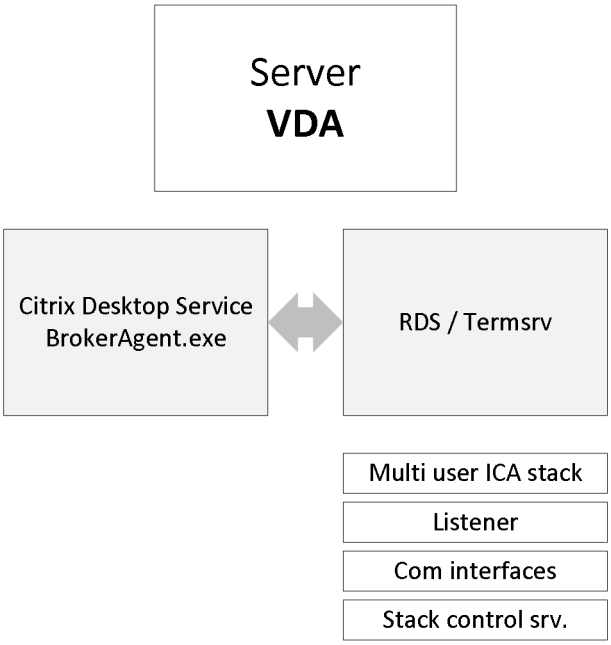
While not directly related, the PortICA service, now referred to as PicaSvc2.exe as of XenDesktop 7, does not represent the whole ICA stack within a Desktop VDA, it 'just' controls it. The ICA stack, and this goes for the RDP protocol stack as well, is made up out of a whole bunch of different components all running and interacting within kernel mode.

FMA fact: *As opposed to the Desktop VDA, which has been around for a couple of years now, there is no PortICA service within a Server VDA, it simply doesn't exist. Of course, similar functionality does exist but is handled by different components and services.*

VDA vs. VDA

One of the biggest differences between the Server and the Desktop VDA is its ability to accept and manage multiple users' sessions at once, hence the RDSH (XenApp) model. Where the Desktop VDA, also referred to as PortICA, can only handle one ICA session at a time. Server VDA's communicate directly, and exclusively with the Delivery Controller and as such they do not need access to the Central Site (SQL) Database or license server.

Also, the underlying OS of a RDSH / XenApp server does not have to be the same as that of the Delivery Controller. And of course, we can use multiple Operating Systems throughout our Site if needed or desired. As mentioned, for server machines Citrix now includes a multi-user ICA stack, which extends the Windows Remote Desktop Services with the HDX protocol. This is the same ICA protocol stack developed for Citrix XenApp, just with a different management interface to make it compatible with XenDesktop 7.x controllers.

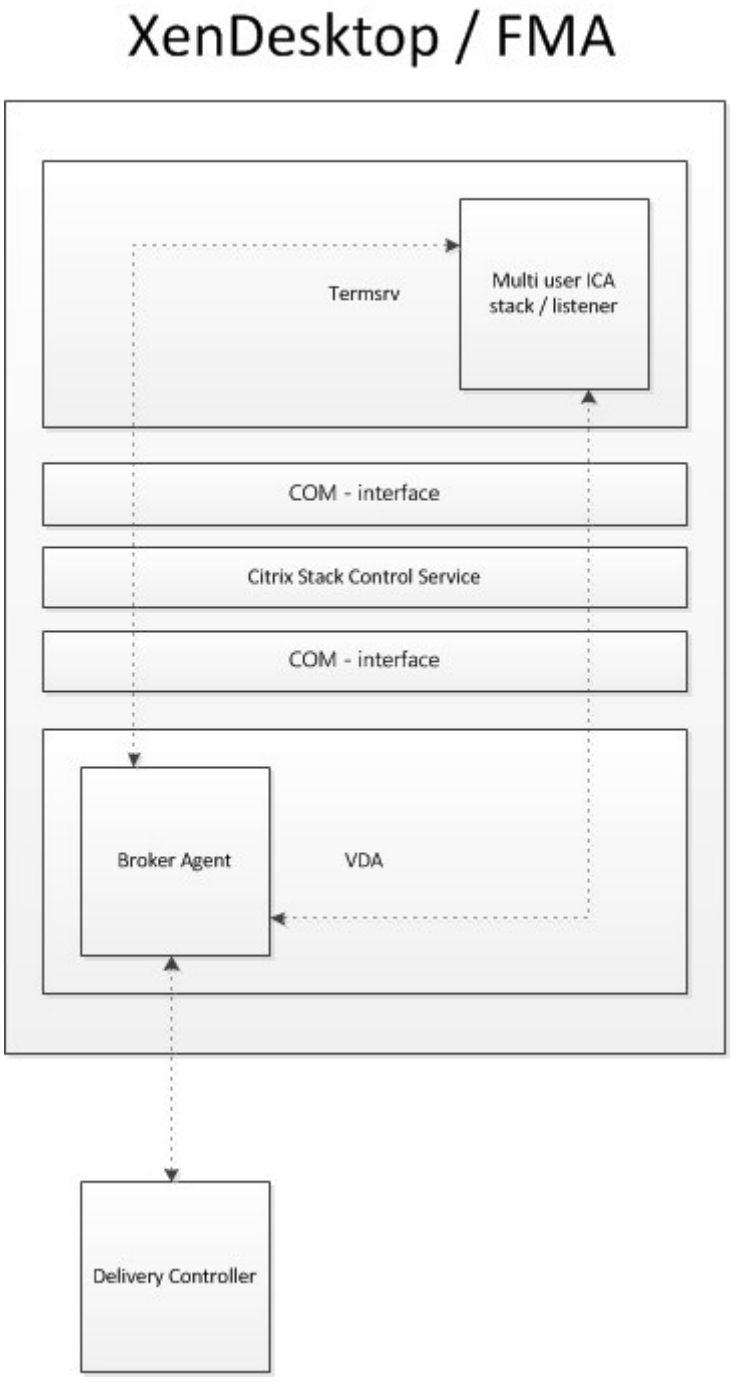


This is what happens during installation

During Server VDA installation one of the things it will do is register the Broker Agent Service (used for direct communication with the Delivery Controller), which is similar to the Desktop VDA process. Next it will install the multi-user ICA stack, as it does with earlier XenApp versions, which will then become part of Termsrv creating the so-called ICA stack listener waiting for new ICA connections (kernel mode).

The ICA stack itself has changed very little with the introduction of the FMA, one of its biggest changes is to be found in its communication interface, which is now better known as the earlier mentioned Broker Agent.

Last but not least, it will install and configure the Citrix Stack Control Service, this is its display name within the Windows services overview. As you will see in the graphical overview below, the above mentioned (Citrix) StackControlService, a.k.a. SCService64, will act as an interface between the Broker Agent (a.k.a. BrokerAgent.exe and part of the new FMA) and the ICA stack running in Termsrv, mapping a direct COM interface between the two.



To a certain extent, you could say that the SCService64.exe takes on a few of the responsibilities, which are similar to those of the PortICA service in a Desktop VDA. Obviously, this is new behaviour and was first introduced with XenDesktop 7 (FMA 2.0). The same can be said for the PortICAsvc.exe as part of XenDesktop 5.x, as of XenDesktop 7 it has been slightly altered and renamed to PicaSvc2.exe.

FMA fact: *Each Terminal Server protocol (like Citrix's ICA) will have a protocol stack instance loaded (a listener stack awaiting a connection request). When installed, the Server VDA basically extends Microsoft's RDS protocol with the HDX feature set / protocol.*

Although similar, still different

Although the above might show a lot of similarities to the way the ICA stack and communications functioned with earlier XA versions, the Server VDA is much more simplified and light weight when compared to earlier XA/ICA installations (although I believe it's still over 300 MB in total).

It now solely consists out of the components needed to host sessions and as such it doesn't share any of the other components and services installed on the Delivery Controllers, which wasn't the case with 6.5 and before.

Let's round things up with a Server to Desktop VDA comparison / recap overview:

Server VDA	**Desktop VDA**
Build from ground up	Existing / updated
Multiple sessions, multiple users	One session per user
Reconnect is allowed	Reconnect is allowed
Can publish desktops and applications	Can publish desktops and applications
CTX Stack Control Service / ICA Stack	PicaSvc2.exe a.k.a. PortICA
Desktop Service a.k.a. BrokerAgent.exe	Desktop Service a.k.a. BrokerAgent.exe
Server OS only	Desktop OS only
ICA to RDP and RDP to ICA session stealing is NOT supported	ICA to RDP and RDP to ICA session stealing IS supported
Non-brokered RDP and ICA connection allowed	Server OS can be hosted using a Server VDI configuration / setup
	Non-brokered RDP connections are allowed, non-brokered ICA connections are not allowed, except in HA mode.

Central Site Database

The XenDesktop Central Site Database holds all Site wide static (policies, configured Catalogs and Delivery Groups, Host Connections, Zones and so on) as well as dynamic (run-time) information (who is logged on to which VDA, what resources are currently in use etc.) needed during the user logon, authentication and resource enumeration process as well as the actual resource launch sequence (load balancing).

Needless to say, it is an important part of your infrastructure: prior to version 7.12 - when it is down your users won't be able to connect and/or launch resources and IT will not be able to make any configuration changes to the Site itself. Because of this you'll probably want to implement some kind of high-availability mechanism keeping your database up and running at all times, or at least to try and keep downtime to a minimum. Of course, the Local Host Cache helps in keeping certain resources available even when the Central Site database is offline, as does Connection Leasing.

It is not just the Central Site database

Besides the Central Site database XenDesktop also has a Configuration-Logging database and a Monitoring database. The Configuration-Logging database stores information about all Site configuration changes taking place, including other administrative activities. This database is only used when the Configuration-Logging feature is enabled, which it is by default. The Monitoring database stores all information used by Director, like session and connection information.

Database sizing

When properly sizing your FMA Central Site database you need to keep your eye on two files in particular: the database file itself, which will contain all Site information, the data and objects like stored procedures, tables, views and so on, and the so-called Transaction-Log file. The later contains a record of all transactions, including any database modifications that might have been made by a transaction. If there is a system failure and the current live Site database becomes corrupt and/or unusable in any other way, the Transaction Log can be used (replayed) to re-create the database and bring it back to a consistent state. However, this does depend on how you configure the Transaction-Log to handle data. You have the following options:

- Simple Recovery mode: In this mode, no log backups are required, meaning that no transaction log data will be saved. If the database fails, all changes made to the database since the last full back up must be redone.
- Full Recovery mode: This mode does require backup logs. If the database fails no work is lost. All data or any specific point in time can be recovered. Full Recovery mode is needed for database mirroring.

- Bulk-Logged Recovery mode: This model is an adjunct of the full recovery model that permits high-performance bulk copy operations. It is typically not used for Citrix databases.
- When backing up your Site database on a daily basis, or at least multiple times per week, simple recovery mode will probably be sufficient. However, it depends. If Site configuration changes are constant, multiple times per week or daily even, then Full Recovery mode might be desirable. Always make sure that high performance storage is used for your SQL infrastructure, SSDs preferably.

Site Database

Typically, the size of your Central Site database will depend on multiple factors. Since the information stored is both static as well as dynamic, its size can vary during the day. The following factors need to be taken into consideration: the number of configured and registered VDAs, connected and active sessions, the number of transactions taking place during logon and logoff, general logon and logoff behavior, the physical size of your Site, and a few more.

Local Host Cache

With XenApp and XenDesktop version 7.12 Citrix has re-introduced the Local Host Cache functionality. A long-awaited feature by many Citrix admins globally. Though many of us are familiar with the LHC feature within 6.5, LHC as part of the FlexCast Management Architecture, which is basically what we are talking about here, is architected differently, or built from the ground up even. This has resulted in a more robust solution, immune to corruption (at least that's the general idea) also needing less maintenance. Let's have a look and see what it is about.

As you probably know, the Local Host Cache ensures that Site-wide brokering operations will be able to continue (new and existing connections/sessions) even when the connection between your Delivery Controller (s) and the Central Site Database is down or fails altogether. The same applies to the Citrix Cloud (CC) by the way. If, for example your WAN link, which connects you to the CC control plain fails or goes offline for whatever reason, the LHC will ensure that new and existing connections/sessions will continue to be brokered/maintained. Even when your CC, or on-premises Database becomes unreachable, or goes offline for other reasons the LHC will have your back.

Remember that with CC the so-called cloud-connector basically replaces your on-premises Delivery Controller. Therefore, all the specific LHC services and/or components on your Delivery Controller (s) can be found on your Cloud Connector machines as well.

Before we have a closer look, you should know that with the re-introduction of the LHC Citrix will advise all her customers to make use of LHC over Connection Leasing (CL) as it is considered far more superior.

How does it work

One of the main LHC components is the Broker Service, which as you probably know also functions as the XML Service and the STA, or Secure Ticket Authority in full. When LHC is involved it is also referred to as the Principal Broker Service. The (Principal) Broker Service will accept connection requests from StoreFront and it communicates with the Central Site Database just like before — brokering connections, taking care of load balancing and so on.

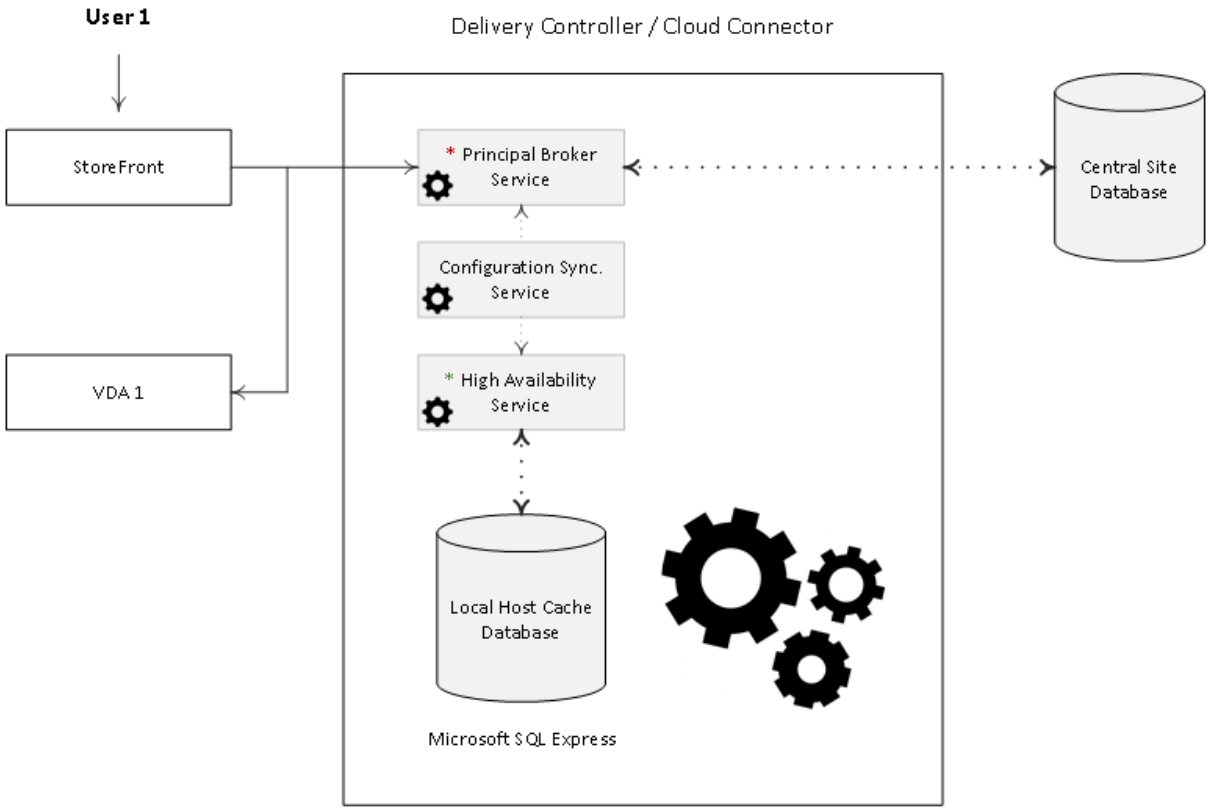
Next to the (Principal) Broker Service we have two new FMA services: the Configuration Synchroniser Service (CSS) and the High Availability Service, which is also referred to as the Secondary Broker Service (yes, both reside on the same Delivery Controller / Cloud Connector).

See the image below for a graphical overview on all this. Every two minutes the (Principal) Broker Service will be checked for configuration changes. If a configuration change has been detected it will be copied over, or synchronised to the High Availability Service/Secondary Broker Service.

FMA fact: *The above-mentioned configuration changes include but are not limited to published icons, changes to Delivery Groups and Catalogs, certain Citrix policies and so on. It will not include information about who is connected to which server (Load Balancing), using what application (s) etc. referred to as the current state of the Site/Farm.*

All (synchronised) data is stored in a Microsoft SQL Server Express (LocalDB) database, which resides on the same Delivery Controller as well. In fact, each time new information is copied over, the database will be re-created entirely. This way the CSS can, and will ensure that all configuration data stored in the Central Site Database will match that of the data stored in the local SQL Express database keeping the LHC current.

FMA fact: *The local SQL Express database has been part of the XenApp/XenDesktop installation as of version 7.9. It is installed automatically when you install a new controller or upgrade a controller prior to version 7.9.*



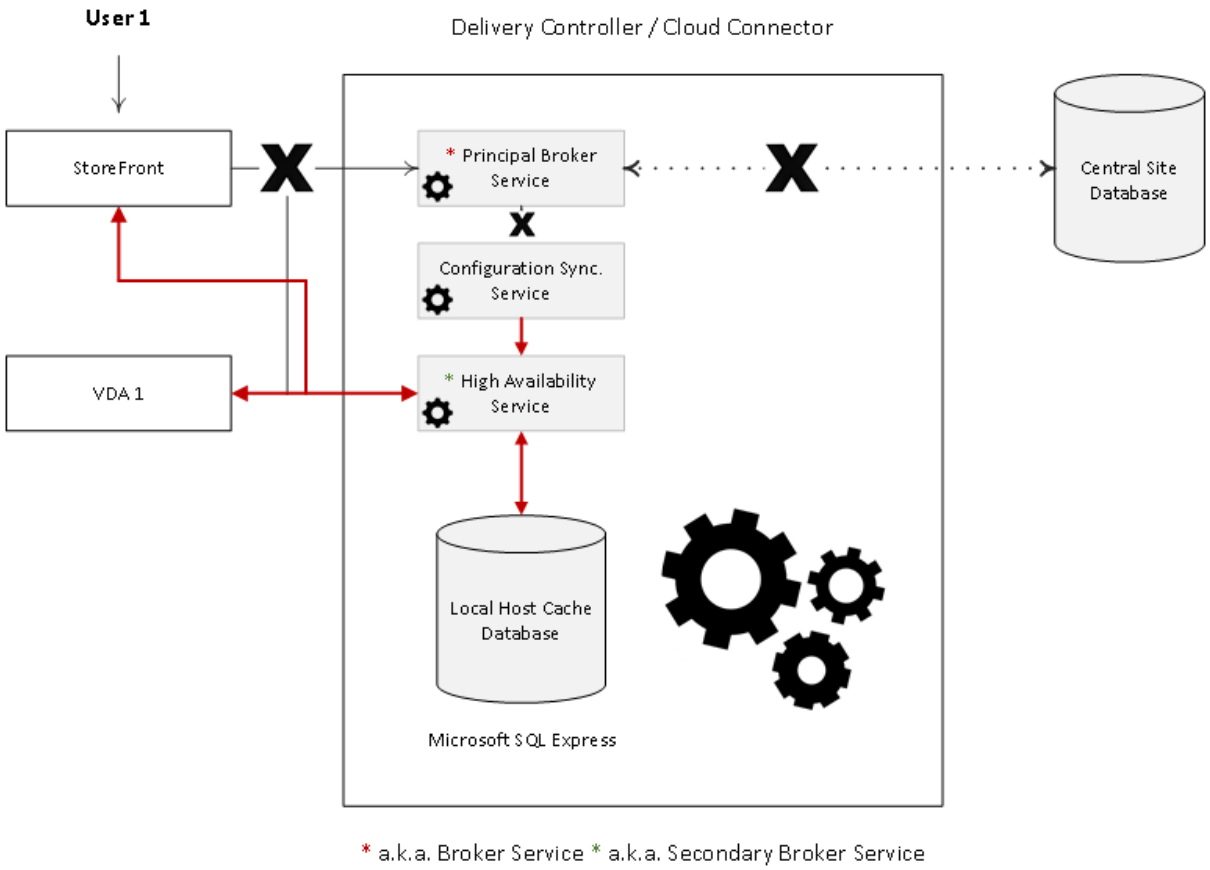
When an outage occurs, the (Principal) Broker Service will no longer be able to communicate with the Central Site Database, as a result it will stop listening for any incoming StoreFront and/or VDA information. It will instruct the High Availability Service/Secondary Broker Service to start listening for incoming connection requests and handle them accordingly.

See the image below (next page) for a graphical overview on all this.

As soon as a VDA communicates with the High Availability Service/Secondary Broker Service a VDA re-registration will be triggered. This way the High Availability Service/Secondary Broker Service will receive the most current session information related to that specific VDA (who is connected to which machine, for example). In the meantime, while the High Availability Service/Secondary Broker Service is handling new and existing connections/sessions, the (Principal) Broker Service will continue to monitor the connection to the Central Site Database.

As soon as it notices that the connection to the Central Site Database has been restored it will instruct the High Availability Service/Secondary Broker Service to stop listening for, and handle new and existing connections/sessions. From this point on it will resume brokering operations as before, basically repeating the abovementioned steps of VDA registration to get up to speed with the latest connection/session information.

Finally, the High Availability Service/Secondary Broker Service will remove any remaining VDA registrations and will again continue to update the local SQL Express database (together with the help of the CSS) with any configuration changes from that point on, as highlighted before.



FMA fact: *From the E-Docs – In the unlikely event that an outage begins during a synchronization, the current import is discarded and the last known configuration is used.*

Modular

To me this again is an excellent example of the FMA's modularity and flexibility. While all this still takes hard work and dedication from multiple product teams within Citrix, by altering/recoding a couple of existing services (instead of the whole 'package' when compared to 6.5) and adding in a few new ones, they were able to completely re-build the LHC from the ground up, and for the better I'd might add.

Zones and multiple controllers

The Configuration Synchronizer Service will provide the High Availability Service/Secondary Broker Service with information on all other controllers within your Site (Primary Zone), this will also include any additional Zones you might have configured. This way each High Availability Service/Secondary Broker Service will know about all of the

other available High Availability Service/Secondary Broker Services within your (entire) Site.

Communication between the various High Availability Services/Secondary Broker Services takes place over a separate channel based on an alphabetical list containing the FQDN's of the machines they (the services) currently run. This information is used to elect which High Availability Service/Secondary Broker Service (read: Delivery Controller) will take over within that specific Zone when the LHC becomes active because of a DB failure or another outage of some sort. Hmm… sounds familiar, right?

FMA fact: *We now have LHC for XenDesktop as well. However, do note that the Local Host Cache is supported for server-hosted applications and desktops, and static desktops; it is not supported for pooled VDI based desktops. In other words, resources need to be assigned.*

What else?

There is some stuff that you can't do when the LHC kicks in, like using Studio, for example, or making configuration changes using PowerShell to name another. Have a look at the accompanying LHC E-Docs pages for some more detailed information. There it will also tell you that the local SQL Express database / the LocalDB Service can use up to 1.2 GB's of RAM and that the High Availability Service can use up to 1 GB of RAM if the LHC stays active for a longer period of time. In short, you will have to account for some additional memory on top of the usual requirements when it comes to sizing your delivery Controllers.

As for storage, when the LHC is active and VDA's start re-registering the DB will grow. Citrix testing has shown that with a logon rate 10 logons per second the database will grow around one MB every 2-3 minutes. As soon as normal operations resume the local database will be re-created also reclaiming the earlier used space. It goes without saying that your Delivery Controllers will need to have sufficient (free) disk space to cope with this.

In a single-zone VDI deployment, up to 10,000 VDAs can be handled (up from 5000) during an outage. In a multi-zone VDI deployment, up to 10,000 VDAs in each zone can be handled during an outage, to a maximum of 40,000 VDAs in the site.

Since all connections/sessions will be handled by a single Delivery Controller (including all VDA re-registrations) when the LHC becomes active, the load on that machine will probably be higher than usual when, for example, connections/sessions are load balanced among all available Delivery Controllers within the Site/Zone. This will result in a higher than average CPU usage, something to keep in mind as well.

Don't forget that in theory every Delivery Controller can be elected as the main High Availability Service/Secondary Broker Service.

Turning it on and off

Using PowerShell, you have the ability to enable or disable the LHC functionality.

To enable LHC use: Set-BrokerSite -LocalHostCacheEnabled $true -ConnectionLeasingEnabled $false

As you can probably see in the above command, this enables LHC and disbales the CL functionality at the same time.

The same thing happens when the LHC is disabled, see below:

Set-BrokerSite -LocalHostCacheEnabled $false -ConnectionLeasingEnabled $true

Using the Get-BrokerSite cmdlet you will be able to check the current state of the LHC.

CDF Control

When it comes to troubleshooting you have a few options, for one you can start by checking the event logs (a couple of specific events might be generated, see E-Docs), after enablement the CSS can produce a trace report (you can force it to) and finally the Broker Service configuration can be exported for debugging purposes.

But that's not all. CDF Control can also be used to potentially identify any issues that you might be having.

CDF Control can be used as a stand-alone application (it's just an executable, nothing more) or as part of Citrix Scout, which will be installed on your Delivery Controllers by default.

CDF tracing works by reading so-called modules, which are built into the various Citrix components and services, like the Synchronization and Broker Services. These modules contain trace-messages, which when read will tell us their current status, this information is what gets logged as part of the actual CDF trace. These statuses could be telling us that everything is fine, for example or they can produce an error message of some sort pointing us in the right direction.

The Citrix Receiver

Here I've copied and pasted almost an entire chapter from my book: **Inside** Citrix The FlexCast Management Architecture: While first introduced by Citrix at Synergy back in 2009, the Citrix Receiver was formerly known as the 'ICA Client'. As Citrix added more products and capabilities to their portfolio (and thus more clients to install and manage) the name slowly evolved from ICA Client to Citrix Receiver (with a lot of steps in between), as we know it today.

The idea behind Receiver was, and still is, that it functions as a container, or placeholder to hold all other Citrix client software providing administrators with a central point of management. The Receiver itself doesn't really do anything. This way all the various Citrix clients that might be needed on a client device, like the ICA Client software to access online applications and desktops, Password Manager (RIP), SSL VPN client software, the Secure Access Gateway client and so on, can all be managed and maintained from a single location.

You might also recall that around the same time Receiver was introduced, Citrix started to rename some of their existing clients. For example, the ICA Client was renamed as the Online Plugin, and the Access Gateway client to Secure Access Plugin; they also introduced an Offline Plugin, Web Plugin etc.

The thought behind this was that these clients would 'plug in' to Receiver, which would then take over about management and maintenance. The only thing missing in this approach was a mechanism to deliver (and manage) the various plug-ins to the Receiver software installed on the client device. For this they introduced something called a Merchandising Server together with Citrix Receiver 1.0. The Merchandising Server provides the administrative interface to configure, deliver, and upgrade plug-ins for your users' computers.

From there Receiver has matured from version 1.0 back in 2009 to version 4.8 at the time of writing. The 4.8 client went GA in June 2017, and it was only as of version 3.0 that it was referred to as the Citrix Receiver (available as a complete package); before that it was still 'just' a placeholder for all the other plugin clients accompanied by the Merchandising Server mentioned earlier. As of Receiver version 3.2 and newer versions going forward, it slowly moved away from the Merchandising Server for updating and maintaining the plug-ins contained within Receiver, handing this functionality over to Citrix.com. Anyway, let me dig in a bit deeper.

Some history

The Citrix Receiver didn't just appear out of nowhere: it was quite a journey. During the next section, I'll try and take you through some of its history and highlight a couple of the most important changes along the way.

- It all started with the first release of the ICA Client software, version 4.0 back in September 1998. Note that I am not referring to the actual ICA protocol and earlier releases like Citrix Multiuser, WinView and WinFrame after that; these were all released (much) earlier.
- It contained the Program Neighborhood, Program Neighborhood Agent (PNAgent) and Web install packages.
- As of version 11.0 it got renamed as the Citrix Plug-in for hosted applications, which was also known as the Web Plug-in. This was over ten years later in June 2008. Although it was renamed, it still contained the same software packages.

- While this didn't directly lead to another name change, around version 11.2 they officially released Citrix Receiver (version 1.0) together with the Merchandising Server. This was in May 2009 during Citrix Synergy.
- As mentioned, initially Receiver was meant as a placeholder for all other Citrix clients, which as of Receiver 1.0 got renamed one by one (Plug-ins), or most anyway. It stayed this way all through 2010.
- Soon after, when version 12.0 came out (March 2010) it got renamed (again) to the Citrix Online Plug-in. Are you still with me?
- With the release of version 13.0 of the ICA Client software, or Online Plug-in it was now officially renamed as Citrix Receiver, version 3.0 at that time. This happened in August 2011.
- As highlighted earlier, as of version 1.0 Receiver was a self-service orchestrator and an updater, also referred to as the Receiver Infrastructure (Merchandising Server).
- At that time, Receiver 3.0 got split into two parts, Receiver Updater and Receiver Inside.
- Receiver Inside was integrated into the Online Plug-in to form a new package, which got named Citrix Receiver Enterprise.*
- It stayed this way up to Receiver Enterprise version 3.4.
- With version 4.0, which got released in June 2013, they changed its name to Citrix Receiver for Windows, the name it holds today.
- The Receiver Updater, as a separate component together with Merchandising Server, stayed with us until August 2015 (EOL). After a user installed Receiver Updater on his or her user device, Receiver Updater installed, updated, and restarted the Citrix Receiver without any user interaction needed.
- In between, from Receiver Enterprise version 3.2 to version 3.4 it still included the original PNAgent software, which was then referred to as Legacy PNA.
- As of version 4.0 the PNAgent functionality was no longer part of Citrix Receiver. This also meant the end of the Desktop Lock feature as it relied on the PNAgent functionality.
- Interestingly enough, Citrix reintroduced the Desktop Lock feature with the launch of Citrix Receiver version 4.2. Only this time they named it Receiver Desktop Lock.
- Citrix Receiver version 4.4 was launched in December 2015, which is the most recent version at this time.
- As of late-2012 they slowly moved away from the Receiver Infrastructure and Merchandising Server, which was EOL in August 2015.
- Today we install and configure Citrix Receiver either manually, using Group Policy (icaclient.adm GPO template), via a (start-up) script of some sort, as part of a base image, using our software distribution software of choice, or… through Receiver for Web sites. When a user accesses a Receiver for Web site from a computer running Windows or Mac OS X, the site will attempt to determine whether Citrix Receiver is installed on the user's device. If Citrix Receiver cannot be detected, the user is prompted to download and install the appropriate Citrix Receiver for their platform.

- When Citrix started to rename their Citrix client software (remember the Plug-ins?) they also introduced a couple of new clients, like the Offline Plug-in, the Desktop Receiver and the Citrix Self-Service Plug-in a.k.a. Dazzle. And I'm sure I left one or two out.
- The Self-Service Plug-in (Dazzle) was integrated with Citrix Receiver and your existing XenDesktop and XenApp infrastructure. It communicated with the Citrix Delivery Services (basically what we now call StoreFront) or Web Interface. Merchandising Server was used to control and manage it.
- The Offline Plug-in, version 5.1, was first introduced in May 2009 and disappeared in July of 2012. Earlier it was also referred to as the Citrix Streaming client, and used, as the name implies, with Citrix-streamed applications. For offline use, I might add. Unfortunately, Citrix Streaming never really took off.
- The Desktop Receiver was aimed at launching hosted shared desktops only (full screen), not published applications. Here they added the well-known pull-down menu, making it easier to switch between your hosted desktop session and your local desktop.

* While the standard Receiver for Windows is for general use, Receiver for Windows Enterprise is intended only to support:

- Repurposed PCs and Thin Clients configured with desktop lock
- Applications that require Fast Connect-enabled products
- Applications that require 508 compliance

Don't get confused by the different version numbers. There is a version numbering scheme for the ICA Client software versions (which has been renamed multiple times) and one for the Citrix Receiver versions.

It's not just Windows.

While the previous section primarily focused on the Citrix Receiver for Windows, there is a Citrix Receiver for almost every platform out there; I'll list them here:

- Receiver for Windows.
- Receiver for Mac.
- Receiver for iOS.
- Receiver for Linux.
- Receiver for Android.
- Receiver for Chrome.
- Receiver for HTML5.
- Receiver for Windows 8/RT.
- Receiver for Windows Phone.
- Receiver for BlackBerry 10.

Next to the various types of Receivers there are also multiple Receiver plug-ins like the HDX RealTime Media engine for Microsoft Skype and Lync (multiple versions), Receiver for Desktop lock (multiple versions) and the Offline plug-in.

Citrix Receiver communications

The Citrix Receiver is a client application enabling us to connect to various Citrix services like XenApp and XenDesktop, but also the XenMobile App Controller, the NetScaler Access Gateway and StoreFront, to name a few more. While Citrix Receiver basically sits idle during the user authentication and application enumeration processes (you don't really need a Citrix Receiver for that), it plays a leading role when it comes to launching a published desktop and/or application and establishing a secure connection up to XenApp / XenDesktop. It literally channels the ICA / HDX traffic back and forth between the client and server.

***FMA fact**: HDX is not a replacement for the ICA protocol. It offers a set of capabilities or technologies that offer a high-definition user experience, which are built on top of the ICA remoting protocol.*

Once a resource is launched, after all background communications are done and load-balancing decisions have been made, one of the first things that will take place is something referred to as the ICA handshake.

During this phase, the client (Receiver) and the server (VDA) will exchange information on the virtual channels that the client supports: here the client basically tells the server which specific capabilities (virtual channels) it can or should use during the connection. This way the server knows what to expect and what virtual channels it can or should use.

Virtual channels, you say?

I know, virtual channels? Yes. A big part of the communication between the client and the server takes place over and through what Citrix refers to as virtual channels (and not just Citrix by the way). Each virtual channel consists of a client-side virtual driver (part of Receiver) that communicates with a server-side application (part of the VDA).

Virtual channels (there can be 32 channels in total) are mainly used/applied for some of the bigger, well-known features like client drive mapping, smart cards, the clipboard, printing, audio, video and so on. However, the Citrix Receiver also supports a whole bunch of additional features and functionalities that do not involve or need a virtual channel.

And of course, from time to time, new virtual channels are released with a new version or Feature Pack of XenDesktop / XenApp and Receiver products to provide additional functionality. Like Framehawk and ThinWire Plus, for example. Those were released as part of Feature Pack 3 for XenDesktop 7.6, including a new Receiver (4.3) if I am not mistaken) on the client side.

Each virtual channel represents a specific feature or functionality on its own. But don't worry, I will elaborate a bit more on all this during one of the upcoming chapters where we will talk about the ICA protocol and the HDX additions in more detail.

***FMA fact**: If you want to make use of e-mail-based discovery you will need to use StoreFront, it does not work with Webinterface.*

So, you see, most features and functionalities configured at server level (mainly through policies) will need to be supported by the client as well; there is a strong dependency between the two (think about the ICA handshake mentioned earlier). And while most features apply to XenDesktop / XenApp VDAs, some do apply to StoreFront, NetScaler and/or Web Interface as well, most are related to security and communication.

Receiver for web access, which is built into Receiver based on HTML5 technology, or secure remote access via the NetScaler Gateway, NetScaler full VPN capabilities, RSA soft token support, Pass Through authentication, Smart Access policies and filters, IPv6 and more. While these types of features and functionalities mainly apply to components like StoreFront, NetScaler etc. your locally installed Citrix Receiver must be able to support these as well.

As mentioned, some of the bigger and better-known features that get built into XenDesktop come with their own virtual channel; however, the Citrix Receiver also has an extensive and impressive list of features that it is capable of without the need for a virtual channel. Check out the link below to find out more:

https://www.citrix.com/content/dam/citrix/en_us/documents/products-solutions/citrix-receiver-feature-matrix.pdf

Although not mentioned on the Receiver Feature Matrix, the Desktop lock is another example. It is dependent on the type and version of Receiver installed and probably best described as an add-on on top of Receiver. The Desktop Lock feature is often used when thin clients are not optional and you do not want to get rid of your older desktop machines just yet. When installed and configured properly (on a domain-joined machine) it will pass on the user's AD / domain credentials, logging them directly into their VDI session, without the user being able to interact with the local physical desktop.

But there is more. Features like Session Sharing, Session Reliability, Auto Client Reconnect and ICA Keep-Alive etc. are all Receiver-dependent as well, although these have been around for some time now. Again, more details will be discussed in the 'ICA / HDX protocol' chapter.

Connection information

During the StoreFront section, earlier I already mentioned a couple of ways how users can connect to StoreFront using Citrix Receiver. However, this doesn't just happen magically, somehow you will need to tell Receiver how and where to connect to: there are a couple of ways to achieve this.

First, and this is a popular approach, you can configure something called e-mail-based account discovery. This way, all your users will have to do is fill in their email address. After that, once they hit OK, Citrix Receiver will automatically determine the NetScaler Gateway or StoreFront Server associated with the email address. This method is based on Domain Name System (DNS) Service records. The user will then be prompted to log on using their domain credentials.

***FMA fact**: If you want to make use of e-mail-based discovery you will need to use StoreFront, it does not work with Webinterface.*

Using StoreFront, you can also create your own provisioning files, which contain connection details needed for your users to connect. Once Receiver is installed all they have to do is double-click the provisioning file and Receiver will be automatically configured. When using Receiver for Web sites, and many do, you can also offer the provisioning file on there. Just like with the Citrix Receiver installation file itself.

As a third option, you can provide your users with the information needed to connect up to StoreFront for them to enter manually. Here you can use the XenApp services site if you are still on 6.5, or you can provide your users with the address of your StoreFront server for them to be able to access the Store(s) on there.

After the information has been entered Receiver will first try and verify the connection: once done and successful your users will be prompted to fill in their user credentials.

Self Service Mode

By adding StoreFront to Receiver, as we've just talked about, you can configure something called Self Service Mode (will be enabled by default). It enables the user to subscribe to resources directly from the locally installed Receiver (right-click on the system tray icon) and, just like with the Receiver for Web sites approach, Keywords can also be used to pre-subscribe certain resources to your users.

Another potentially advantage when using this approach (opposed to the more limited Web Access Mode (although preferred by many) where Receiver is not configured and users access a Receiver for Web site) is the ability to (almost) fully manage and customise the application short cut location (or you let your users decide for themselves).

This way, published applications can appear in your users' Start menu and/or desktop without them being able to uninstall. Your users will not have to manually subscribe to their resources before being able to launch them. Of course, these two modes, Web Access and Self Service, can be configured and used side-by-side.

***FMA fact**: All, or at least most, of these resource short cut management options were already available with Citrix Receiver Enterprise up to version 3.4, when they killed it. It took up to Citrix Receiver version 4.2 to get this functionality back.*

Just be careful and think about when to implement which solution. Not all your users will be happy with a preconfigured Start menu or desktop, for example, especially when dozens or more applications are involved. Unfortunately, there will always be some pros and cons no matter which route you choose.

You have options

Luckily, when needed you do have some options to play with. Applications can be configured on an individual basis to be placed in the Start menu or on the desktop, or you can let your users decide. Put them all on the desktop or in the Start menu by default or a combination of all options mentioned: it is up to you. This is accomplished by manually configuring the Citrix Receiver. If you want…

- Your users to be able to choose the resources they want in their Start menu, then you configure the Receiver in Self Service Mode.
- Your users to be able to choose the resources they want in their Start menu and you also want to force a few specific resources onto their desktops, then you will need to apply those specific settings on a per application basis.
- By configuring the Receiver with PutShortcutsInStartMenu=False you prevent the Receiver from putting application short cuts into the Start menu automatically.
- PutShortcutsOnDesktop=true will put all application short cuts on the user's desktop.

These are just a few examples to give you an idea of the possibilities: it's very flexible. There are a couple of more options available so make sure to check out the E-docs pages on Citrix Receiver.

***FMA fact**: By disabling the SelfServiceMode (it is enabled by default) subscribed-to applications can only be accessed through the Start menu and desktop short cuts. This is also referred to as short cut-only mode.*

To conclude

Now, of course, there is still a lot more to tell and share when it comes to the Citrix Receiver, but hopefully this section provided you with enough information on some of its inner workings and more specifically why it is such an important piece of the puzzle.

StoreFront and Webinterface

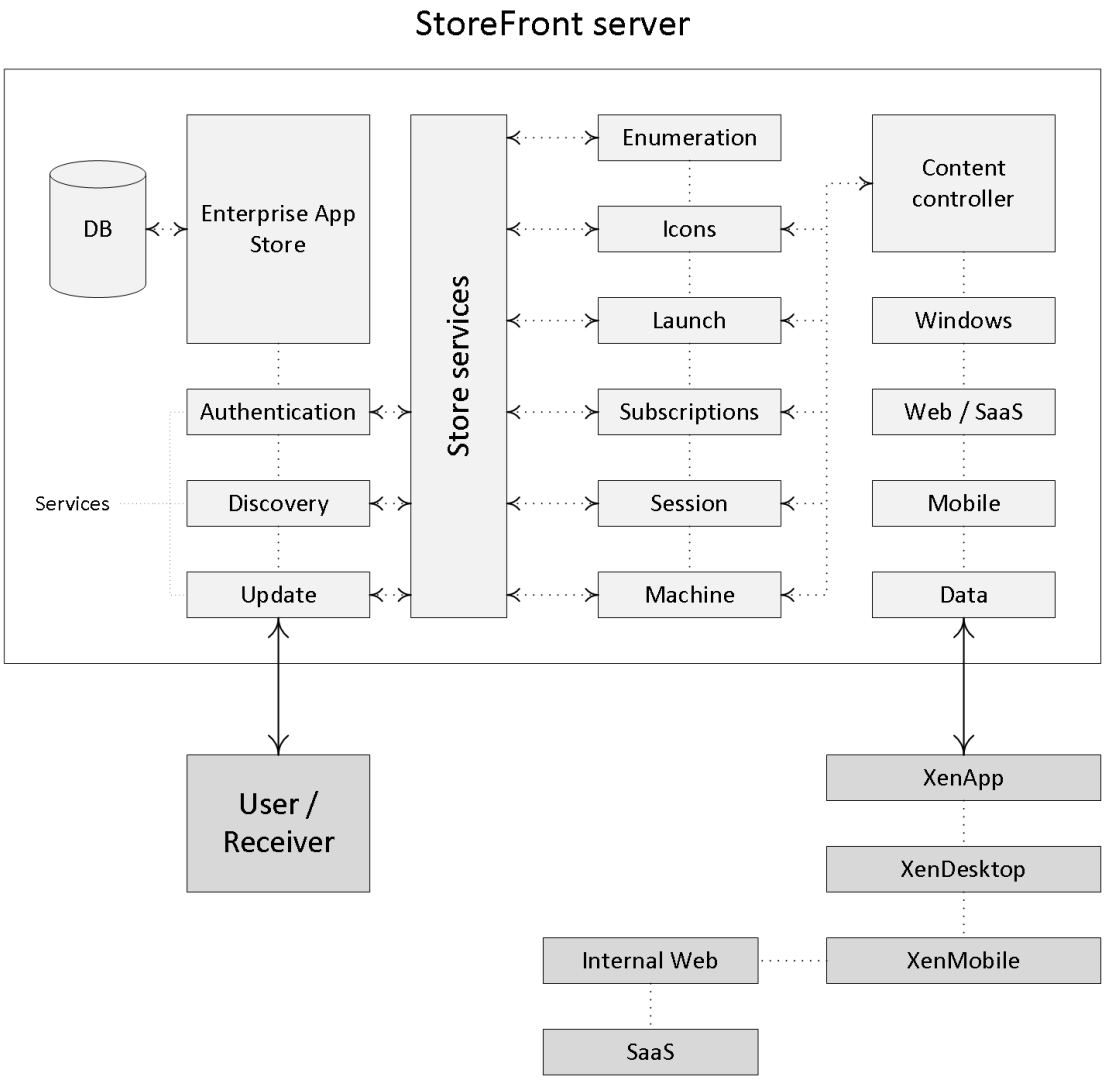
Initially, with the introduction of StoreFront it relied solely on its authentication service for user authentication purposes. This, as you might be aware is different from Web Interface, which will directly contact one of the configured Delivery Controllers where the Broker/XML service will take over. Since Web Interface is still widely deployed and used in (large) production environments (and StoreFront now also supports XML based user authentication) I would like to talk, in a bit more detail about both authentication methods available.

FMA fact: *Web Interface will be 'End of Life' in June 2018, however, Citrix advices to deploy StoreFront for new as well as existing deployments.*

Authentication in general

Within a XenApp/XenDesktop Site you basically have two (main) points of authentication, one of which is StoreFront, the other one being the NetScaler Gateway when authenticating externally, for example, though it could be used for internal authentication purposes as well. The StoreFront server will communicate with the Citrix Receiver, your Delivery Controllers and the NetScaler Gateway (call-back and STA) when users are authenticated and resources are launched externally.

Next to the above, StoreFront can also be configured to communicate with App Controller as part of a XenMobile deployment, and/or VDI-in-a-Box is also (still) optional. Like the Delivery Controller, StoreFront plays an important role in the resource enumeration and launch processes and it functions as the main Store (there can be multiple) from where users (can) subscribe to their desktops, applications and other resources.



StoreFront (internal) authentication/enumeration

With StoreFront, users are authenticated by the authentication service (it communicates with Active Directory) which is an integral part of StoreFront. Note that this is default StoreFront behaviour. As of StoreFront version 3.5 it (the authentication service) now also includes the Self-Service Password Reset feature (SSPR).

Users can authenticate to StoreFront using different methods like usernames and passwords, Domain pass-through, NetScaler pass-through, using smart cards or by enabling unauthenticated user access. And of course, we also have technologies like Kerberos constrained delegation for XenApp 6.5, and newer features like the Federated Authentication Service, or FAS, which are both out of scope for now.

As soon as a user logs in by filling in his or her username and password (on the StoreFront web page using the Receiver for website configuration, or using a locally installed Citrix Receiver) the StoreFront authentication service will pick up the user credentials and contact a domain controller where the actual user authentication will take place.

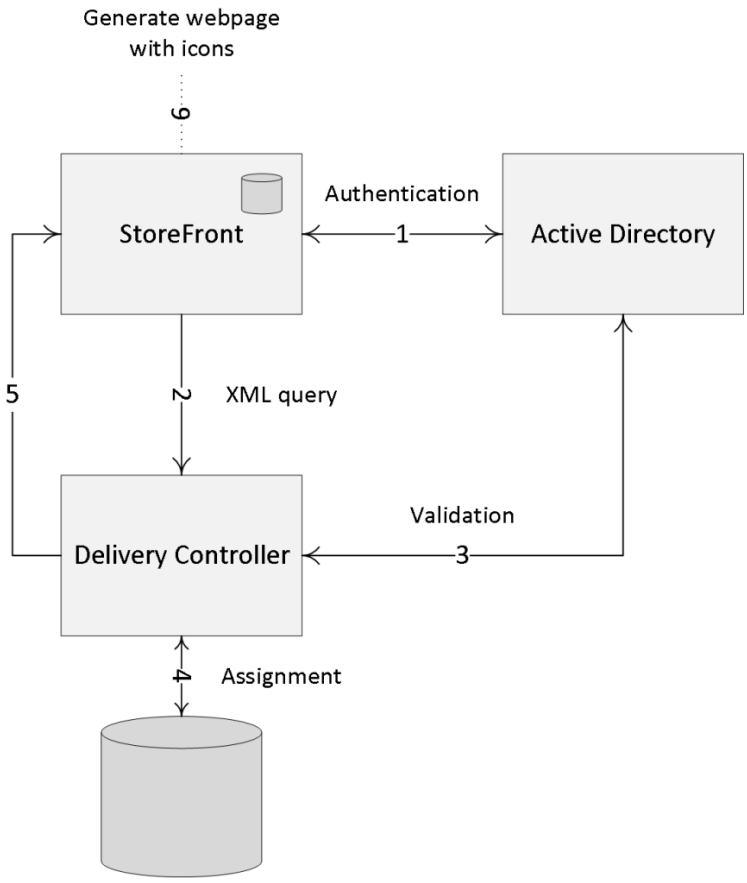
Once authenticated (1), StoreFront will forward the user credentials, as part of an XML query to one of the configured Delivery Controllers (2), assuming you configured at least two of course. In between, StoreFront will check its local datastore for any existing user subscriptions and stores them in memory.

The Delivery Controller receiving the credentials will again contact a Domain Controller, this time to * validate the user's credentials (3) before responding to the StoreFront server. During the next step (4) the Delivery Controller will check with the Central Site database to see which resources have been assigned to the user before sending them over to the StoreFront server (5).

Finally, StoreFront will generate a webpage displaying all the resource icons (published applications and desktops) to the user (6). Here I assume that authentication is taking place internally and directly to StoreFront.

FMA fact: ** Note how I mention user authentication and user validation. There is a difference. Authentication is to make sure that somebody is who he or she claims to be. Verification is done to find out which resources a user is allowed to launch, primarily based on Active Directory group memberships.*

When I discuss the resource launch sequences, you'll see that the above plays an important role in the grand scheme of things.



Web Interface XML based authentication

If we look at Web Interface, user authentication works a bit differently, it has no internal authentication service. When a user logs in by filling in his or her username and password, Web Interface will immediately forward these credentials, as part of a so-called XML query to a Delivery Controller, which will then contact a domain controller to authenticate the user before responding back to the Web Interface server – also referred to as XML based user authentication. As before, Web Interface can authenticate users to, and enumerate and aggregate resources from, multiple XenApp Farms and XenDesktop Sites, not App Controller as part of XenMobile though.

XML based authentication for StoreFront

As of StoreFront version 3.0, Citrix re-introduced XML-based user authentication. By simply running a few PowerShell scripts user authentication falls back to the XenDesktop / XenApp XML service, like with Web Interface as highlighted earlier. Particularly useful when StoreFront is not in the same domain as XenDesktop / XenApp and when it is not possible to set up an Active Directory trust, or multiple. Again, this method will be/is disabled by default: at least now you have options.

FMA fact: *As of StoreFront Version 3.5 and upwards PowerShell is no longer needed to enable XML based user authentication, it can now be enabled and disabled directly from StoreFront management console (GUI). Here's the accompanying E-Docs page showing you how it's done.*

They come in pairs

The StoreFront server plays a vital role when it comes to user authentication, resource enumeration and launch. If there is no StoreFront server available your users will be unable to launch any resources (as an exception, although not recommended, a direct ICA connection would work and doesn't need StoreFront). That is why you will always deploy at least two StoreFront servers per Site. By implementing a load-balancing solution, like a NetScaler or Windows NLB, for example your users won't notice a thing when one or multiple StoreFront servers become unavailable.

To be able to provide your users with desktops and applications, StoreFront must be configured with at least one Delivery Controller (FQDN or IP address). Since 'one is none' we will always make sure to configure at least two Delivery Controllers for HA purposes. In the case of a Delivery Controller failure, StoreFront will automatically fail over to the next Delivery Controller in line; resulting in an active/passive configuration.

Within large (r) organizations, where the logon load is higher than average, an active/active approach might be a better fit. This can be accomplished by implementing a load-balancing

device like the Citrix NetScaler, or you can choose to let the StoreFront server load balance the connections to the various Delivery Controllers instead.

Up to StoreFront 3.5 you will have to manually edit the web.config file for this, locate the following line:

<farm name=”XenApp” xmlPort=”80″ transport=”HTTP” sslRelayPort=”443″ loadBalance=”on” farmType=”XenApp”>

Change ‘loadBalance’ to either “on” or “off”.

As of StoreFront 3.5 the above can be configured by simply placing a checkmark directly from the GUI, you’ll find it under ‘Manage Delivery Controllers’. As far as I know, the built-in LB mechanism used for this is based on RRDNS technology, or at least the result is similar.

Citrix Studio

Citrix Studio is THE management console that allows us to administrate, configure and manage our XenDesktop and/or XenApp Sites from a single pane of glass. It also provides us with access to real-time data collected through the Broker service running on the Delivery Controller.

***FMA fact**: By default, Studio communicates with the Controller on TCP port 80.*

After you have set up your primary Site (something you will always have to do first, assuming there isn’t one in place already) you would normally continue with a Machine Catalog closely followed by a Delivery Group and then on to configuring and publishing applications and/or desktops to your users.

But it doesn’t stop there. From Studio, you have full control over your entire Site and everything in it, including but not limited to Zones, LHC, Application Layering, Machine Catalogs, Delivery Groups, Delivery Controllers, XenDesktop and XenApp (Session Hosts) machines and a lot more. Studio also allows us to integrate StoreFront and App-V, and it is also the place where we add and configure our Host Connection (underlying Hypervisor / cloud platform, or multiple), manage and initiate Machine Creation Services, set up delegated administration, and so on. And as you might know, most features are configured through policies, which are also available from Studio. So, you see there is quite a lot going on from a management perspective.

Basic troubleshooting

Next to configuring and maintaining our Site, Studio also allows us to execute some basic troubleshooting tasks. From Studio you can run several self-diagnostics tests on your Delivery Groups and Machine Catalogs, for example including a Site-wide test. Depending

on the size of your Site, the number of machines, Delivery Groups, policies etc. this will take (at least) a couple of minutes. When done, it will let you know about any issues and/or errors that it might have encountered.

We also have the option to launch PowerShell directly from Studio, which of course is only helpful if you know your way around PowerShell, potentially helpful none the less. And while this isn't directly related to troubleshooting, you can also use the built-in PowerShell functionality to auto-generate PowerShell scripts on almost everything you can do from Citrix Studio.

As mentioned, Studio is the place from where we configure and manage all, some, or most of our Citrix-related Site-wide policies (something that can also be done using the Group Policy Management Console, or GPMC if installed on your Delivery Controller, or on a separate management server for that matter. Note that for this to work you will also have to install Citrix Studio and the Citrix Group Policy extensions on the same 'management' machine). From Studio, we have a couple of options we can use to help in the troubleshooting process, so have a look below:

- An overview of all settings available within all policies combined.
- We can configure filters to apply specific policies to specific objects.
- Policy comparison: by selecting one or multiple configured policies and/or templates you can compare the configured settings within the selected objects.
- Group Policy Modelling allows the user to simulate a policy deployment that would be applied to users and computers before applying the policies.
- Citrix Group Policy Modelling, specific to Citrix-related policies only.

Resultant Set Of Policies is only available when using the GPMC approach mentioned above.

We can see the current license usage and overall statistics; how many users are logged onto which machines and Machine Catalogs; we can restart and/or shutdown machines; send out messages to logged in users; and of course, we can log off users if needed. Studio is also used to put machines, or Machine Catalogs, in maintenance mode.

Host connection

When you install XenDesktop and start Studio, you will first need to create a new Site or join an existing one, no other options are there. If you don't join an existing Site but create a new one, and let's call this step one, then step two and three will walk you through the Machine Catalog and Delivery Group creation processes. If your XenDesktop / XenApp Session Hosts will include virtual machines, and you would like to be able to auto-provision and manage new virtual machines, then you will also have to configure a Host Connection.

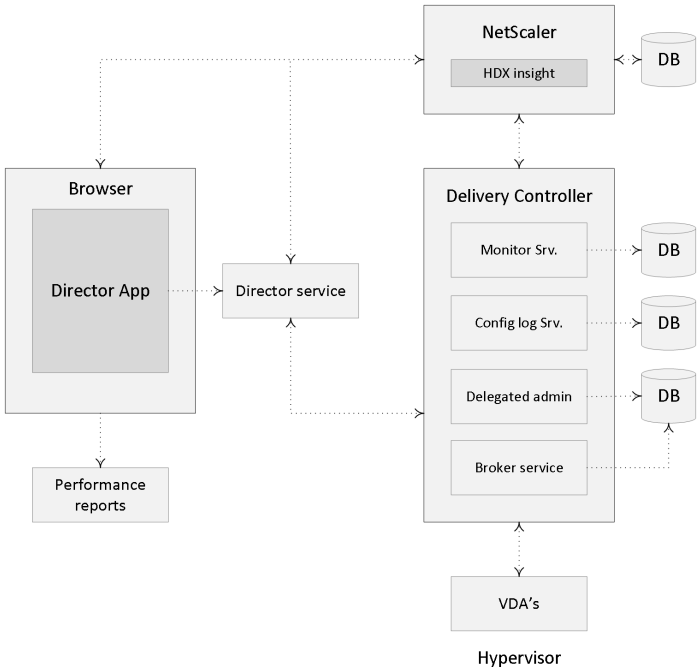
Your Host Connection is basically nothing more than a bridge to your underlying Hypervisor and/or cloud platform of choice where your VMs (will) reside. From a Hypervisor's

perspective, this can be either: Microsoft's Hyper-V, VMware's vSphere, Citrix's XenServer or Nutanix Acropolis. Since you will need to be able to start, stop, create, delete and monitor your VMs from Studio, some sort of management software will be needed, meaning SCVMM in the case of Hyper-V, Virtual Center if you go with VMware, or XenCenter if XenServer is used. Although with XenServer, technically you do not necessarily need a separate management infrastructure (advisable though), a stand-alone XenServer, or Poolmaster would be sufficient to connect to.

At least one of these needs to be in place for this to work; Studio will then use the Hypervisor's APIs to communicate and manage the virtual machines. It goes without saying that it will do the same for Azure or AWS, for example.

Director

Director is a real-time web-based tool (can be installed as a website on your Delivery Controllers) that allows administrators to monitor, troubleshoot (real-time as well as historical) and perform support tasks for their end-users. It is Citrix's first line of defence and it is, by default, included with all editions of XenDesktop and/or XenApp, although higher editions do offer some additional functionality. As of XenDesktop 7.x it has some of the former EdgeSight functionality built in (historical reporting mainly) and if you own the proper licenses (Platinum) for both XenDesktop and NetScaler you can monitor your NetScaler's (HDX Insight) using the same console as well.



***FMA fact**: Make sure to check out CTX139382 for a whole bunch of best practices around Director.*

Citrix EdgeSight (reporting)

As already briefly highlighted, Director makes use of Citrix's EdgeSight technology, which is now an integral part of Director. However, not that long ago, it was still available as a separate product offering the ability to monitor applications, devices, sessions, license usage, and the network in real time, allowing users to quickly analyse, resolve, and proactively prevent problems. The latest version that was released was 5.4.

While EdgeSight (reporting) was very powerful it was also quite complex to operate and work with. Today Director probably covers up to 80 – 90% of all tasks needed by first as well as second line support engineers and at the same time it is very easy and straightforward to navigate. As you might recall Citrix Director started out as a fairly simple monitoring and troubleshooting tool only capable of offering real-time support.

As time progressed people started asking for historical reporting features as well as trend analysis capabilities, and rightfully so. This is where the earlier mentioned EdgeSight functionality comes in. They reused and re-written part of the original code and primarily focussed on the reporting functionality now available in Director, as we know it today.

***FMA fact**: As it stands today, the EOL for EdgeSight has been set to 30-June-18, or 24-Aug-2016, depending on if you have a valid software maintenance and/or Subscription Advantage. In that case, the EOM is set to 31-Dec-17 or 24-Feb-2016.*

While Director will be shipped with every XenDesktop / XenApp version, offering real-time assessment and troubleshooting capabilities, the additional reporting and trends analysis functionality does require a Platinum license for both the XenDesktop and/or XenApp, depending on which product you are using. The same applies to the network analysis functionality; you will need at least a NetScaler Enterprise or Platinum license for this. With an Enterprise license you will be able to store historical data for 60 minutes, while with a Platinum license there is no limit.

OData

While using Director gives us access to a ton of historical information, it is impossible to include everything. Another option you have is the so-called Monitor Service OData API. This can be used to create custom reports for analysing purposes. It is all about getting the information out of your database and converting it to a, for us humans, readably format. And while this may sound complex, it doesn't have to be. In fact, all you need is Microsoft Excel. Using the Open Data (OData) protocol you basically let Excel directly communicate with your Delivery Controller. Have a look at the following Citrix blogpost for some more detailed information on how to set this up:

https://www.citrix.com/blogs/2015/02/12/citrix-director-analyzing-the-monitoring-data-by-means-of-custom-reports/

NetScaler

The NetScaler Gateway is primarily used (yes, it can do a lot more, out of scope for now) to provide our users with secure external access to our XenDesktop sites. As we will see shortly there are some key differences when it comes to external and internal user authentication and the resource enumeration process. In that respect, you could say that it is part of the FMA, though we do not need a NetScaler to set up and configure a XenDesktop Site. As mentioned in the beginning, I will include one of my most popular posts on the Citrix NetScaler near the end, so I will leave it at this for now.

ICA/HDX

In its most simple form, the ICA protocol transports keystrokes, mouse clicks and screen updates (using standard protocols like TCP/IP, IPX, NetBEUI, SPX) from the server to the client in a highly controlled and secure manner.

It is optimised for Wide Area Networks with high latency and offers various levels of Quality of Service. These types of protocols, since there are more besides ICA, are often referred to as remoting protocols.

***FMA fact**: By default, the ICA protocol uses TCP port 1494. If Session Reliability is enabled a.k.a. the Common Gateway Protocol, or CGP then ICA traffic will be encapsulated through TCP port 2598. Note that any network traces that you might run will also show 2598 instead of 1494.*

When Citrix released WinFrame back in 1995 it also introduced ICA version 3.0, which included ThinWire 1.0. Back then, ICA functionality was still limited to ThinWire (screen updates), printing, client drive mapping and audio. Before that there was the ICA version 1.0 in 1992 and the ICA version 2.0, which was also released in 1992 as part of the Citrix Multiuser launch discussed earlier.

Virtual Channels

This is where the client-server set-up becomes even more apparent. Let's do a quick résumé since the virtual channels are an essential part of the ICA protocol stack.

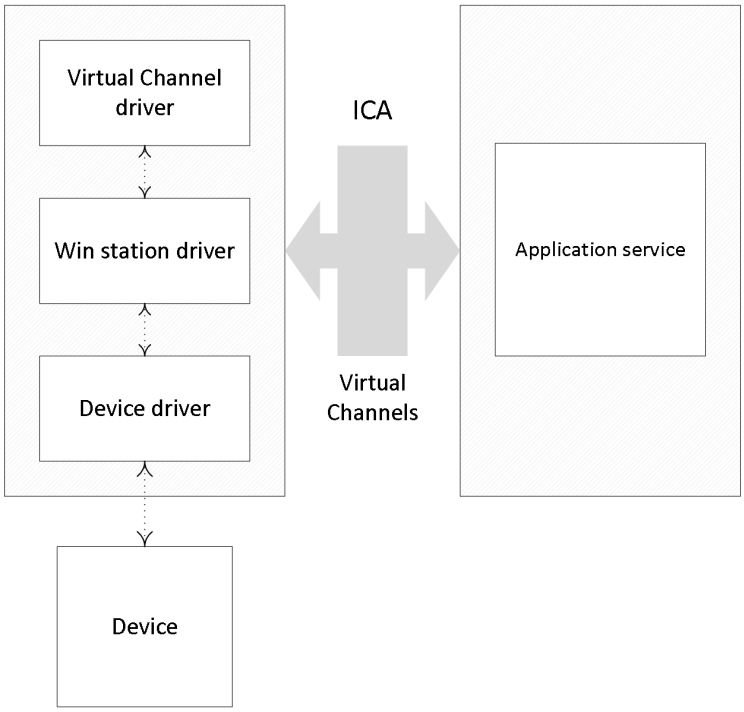
A big part of the communication between the client and server takes place over and through what Citrix refers to as virtual channels. This is where most ICA / HDX features live. Each virtual channel consists of a client-side virtual driver (Receiver) that communicates with a server-side application (the VDA). I say 'most', because Receiver also offers and supports a whole bunch of additional features and functionalities that do not involve or need a virtual channel.

Virtual channels (there can be 32 channels in total) are mainly used for some of the bigger well-known features where a bigger than average and direct communication path between the

client and server is needed, like client drive mapping, smart cards, clipboard, printing, audio, video and so on.

***FMA fact**: As a (security) best practice Citrix recommends disabling any virtual channels that are not in use.*

And of course, from time to time, new virtual channels are released with new versions of XenDesktop and Receiver to provide additional functionality. Take Framehawk and ThinWire Plus, for example. Those were released as part of Feature Pack 2 and 3, respectively, for XenDesktop 7.6, including a new Receiver on the client side. Each virtual channel represents a specific feature or functionality on its own.



What happens in a nutshell

When a new session is established, at client load time, first the client (Receiver) connects to the XenDesktop / XenApp Server (VDA). The client passes information about the virtual channels it supports to the server. This is where the version of the Receiver combined with VDA installed matters. The server-side application starts, obtains a handle to the virtual channel, and optionally queries the client for any additional information about the channel.

As soon as additional data has been sent and received, the server virtual channel application is completed and it closes the virtual channel to free up any resources that may have been allocated.

The above is also referred to as the client server handshake.

The earlier mentioned client-side virtual drivers can be found in the following Registry Key:

HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\ICA Client\Engine\Configuration\Advanced\Modules\ICA 3.0

If you would like to disable certain client functionality, you can do so by editing the Registry Key mentioned above. Simply remove the functionality you would like to disable, like clipboard (clipboard mapping) or ClientDrive (client drive mapping).

***FMA fact**: As mentioned, there are 32 virtual channels in total; however, Citrix reserves 17 of those. Third-party companies and customers who want to design and implement their own virtual channels are free to use the other ones. These are also referred to as dynamic virtual channels or DVCs.*

Most features and functionalities configured at server level (mainly through policies) will need to be supported at the client side as well; there is a strong dependency between the two (think about the ICA handshake mentioned earlier).

Creating your own

Although XenDesktop / XenApp products ship with various virtual channels included, supported by both the VDA and Receiver, they are also designed to allow customers and third-party vendors to create their own virtual channels by using one of the provided SDKs, or Software Development Kits. When you want to create a virtual channel of your own you have two choices. One: you can use the Virtual Channel SDK; or two: you can use the ICA Client Object (ICO) SDK. Citrix offers several resources for you to leverage when it comes to creating your own VC. Here is a shot excerpt from the Virtual Channel SDK page:

The Citrix Virtual Channel Software Development Kit (VCSDK) allows software engineers to write both host-side applications and receiver-side drivers to support additional virtual channels using the Citrix ICA protocol. The host-side virtual channel applications run on XenApp or XenDesktop, and the client-side portion of the virtual channel runs on the local device where Citrix Receiver resides. This SDK provides support for writing new virtual channels for the Citrix Receiver.

Citrix offers the following online resources:

1. The Citrix Virtual Channel Software Development Kit (login with My Citrix needed).
2. CTX113279 – How to Allow Custom Virtual Channels Created with ICO in Version 10.0 of the CTX Windows Client. Although somewhat outdated, it does provide some offer some interesting additional information.
3. Client Object API Specification Programmer's Guide.
4. The Citrix Developer Network. Home of all technical resources and discussions involving the use of Citrix SDKs.

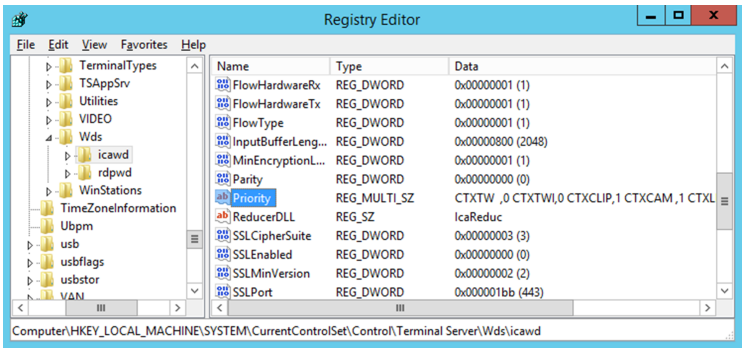
Life is all about priorities

As we have seen, the ICA protocol consists of various virtual channels each offering its own functionality. By default, each of these virtual channels is given a priority, ranging from 0 to 3: the lower the number, the higher the priority.

For example, printing has a default priority of 3, which means that it has the lowest priority and will therefore be allowed less bandwidth then virtual channels with a higher priority (lower number) like Audio and ThinWire (Windows screen updates).

While it is possible to manually change these priorities, it isn't that common to do so. You need to be aware that when giving a higher priority, and thus more bandwidth to one virtual channel it also means that you are taking away potential bandwidth from another virtual channel. However, in the rare occasion when you may want to assign a higher priority to one of the virtual channels, this is how it is done. In Registry locate the key:

HKLM\System\CurrentControlSet\Control\Terminal Server\Wds\icawd\Priority.



There you will find various abbreviations like: CTXCAM, CTXTWI, CTXFLASH and so on, accompanied by a number ranging from 0 to 3 (their current priorities). These two combined represent a single virtual channel.

By simply changing the number, you will change the priority of the associated virtual channel. These priorities, 0 to 3, are also commonly referred to as priority groups. And remember, always be careful when editing the Registry, make sure to back up the key, or keys beforehand.

FMA fact: *Other ways to accelerate ICA traffic would include Citrix policies, which can then be applied either per user or per server, or to the whole Site. Implementing a physical accelerator like the Citrix Cloud Bridge, formerly known as Branch Repeater, is always optional as well.*

Multi-Stream ICA

While this (see previous section) does offer us some level of control with regard to ICA traffic acceleration, it is still fairly limited. By implementing, or activating a feature named Multi-Stream, or Multi-Port ICA we can configure true Quality of Service (QoS) on all or parts of the ICA / HDX traffic sent throughout our network.

Note that I am referring to network-based QoS, which is different from prioritising a single virtual channel. Here we would like to be able to accelerate ICA traffic on a network (TCP/IP port) level rather than from within the ICA protocol itself. Without Multi-Stream ICA, we can only accelerate ICA traffic as a whole on TCP/IP port 1494 or 2598 (Session Reliability) as discussed previously.

With Multi-Stream ICA enabled we can assign separate TCP/IP ports to each of the earlier mentioned priority groups a.k.a. streams within Multi-Stream ICA. This means that we can configure and assign a TCP/IP port for all priority 0 virtual channels, and another separate TCP/IP port for all priority 1 virtual channels, and so on. Meaning that there can be four Multi-Stream ports in total.

Each virtual channel will by default already have a priority assigned to it, as we've seen, ranging from 0 to 3, which correlates to very high (real-time activities, such as webcam content), high (interactive elements, such as screen, keyboard, and mouse), medium (for bulk processes, such as client drive mapping) and low (background activities, such as printing). This is true for single-stream ICA traffic (without Multi-Stream ICA enabled and configured) as well as Multi-Stream ICA traffic (with Multi-Stream ICA enabled and configured). However, as highlighted earlier, these priorities can be manually changed if and when needed. The accompanying Multi-Stream ICA Registry Key, when enabled, can be found at:

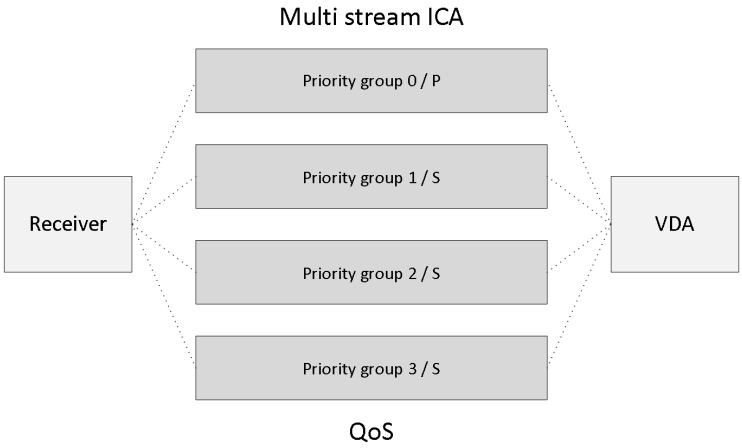
HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\Wds\icawd\MultiStreamIca

It will consist of two subkeys: Stream and VirtualChannels.

Within the Stream Registry Key, we can manually configure the various stream priorities and assign them to be either primary or secondary. Or we can use the default configuration instead. The format used is *Stream#, Stream type*. For example: 0,S will mean; all virtual channels with the priority 0 will be secondary, and 1,P means that all virtual channels with a priority of 1 will be primary and so on. Note that there can be only one primary stream, the rest will be secondary. The default configuration is: 0,S;1,P;2,S;3,S.

Within the VirtualChannel Registry Key we can manually configure the virtual channel stream pairs (binding a VC to a stream), which basically means that we assign a priority to a virtual channel, just like before. Or we can leave the default configuration in place. The format used is: *VirtualChannelName, Stream#.* CTXCAM,1 means that the virtual channel CTXCAM has been assigned to the stream 1.

As a result, it will be part of the stream pair 1,P, as highlighted earlier. The default configuration is: CTXCAM,0; CTXTW,1; CTXTWI,1; CTXLIC,1; CTXVFM,1; CTXPN,1; CTXSBR,1; CTXSCRD,1; CTXCTL,1; CTXEUEM,1; CTXMM,2; CTXFLSH,2; CTXGUSB,2; CTXCLIP,2; CTXCDM,2; CTXCCM,3; CTXCM,3; CTXLPT1,3; CTXLPT2,3; CTXCOM1,3; CTXCOM2,3; CTXCPM,3; OEMOEM,3; OEMOEM,2.



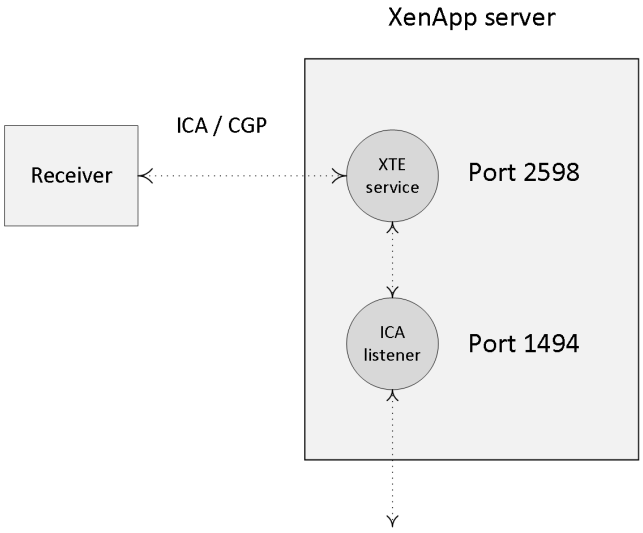
***FMA fact**: When not using a Cloud Bridge appliance, formerly known as Branch Repeater, Session Reliability must be enabled for Multi-Stream ICA to function.*

Once you have set up and configured the stream priorities and pairs you will have to configure a so-called multi-port Citrix policy where you configure separate TCP/IP ports to the primary (one) and secondary (three) streams as explained in the previous section.

Once that is out of the way you can go on and configure and apply QoS policies at network level on a per TCP/IP port level. This way you can apply QoS on grouped ICA virtual channels instead of a single or a couple of virtual channels within the ICA protocol.

Session Reliability

I already mentioned this feature once or twice throughout this chapter; here I will address it in a bit more detail. When Session Reliability is enabled, the ICA Client tunnels its ICA traffic inside the Common Gateway Protocol (CGP) and sends the traffic to port 2598 instead 1494. The XTE service acts as a relay; removing the Common Gateway Protocol layer and then forwarding traffic to the ICA listener on port 1494 internally as shown below:



Internally, all ICA traffic coming from the XenApp server destined for the end-user's client device will be sent through or via the XTE service as well; it basically works the same way, only vice versa. Session Reliability can buffer ICA traffic when the CGP connection between the client and the XTE service is somehow broken; it will then temporarily store all ICA data until the connection is restored. During that time, as long as the XTE service is buffering the ICA data, the user session will not go into a disconnected state; instead the session will remain active on the server.

From a client perspective, the session seems frozen while the client is attempting to reconnect with the XTE service over the Common Gateway Protocol. Once the session is restored, all buffered ICA data will be flushed and sent over to the client device and the session will continue as usual.

Configuration specifics

By default, Session Reliability (SR) is configured via policy and set to 180 seconds, or three minutes before the user session will be dropped and put into a disconnected state. However, this time-frame can be changed when needed. The default port used by SR is TCP/IP port Nr 2598 but can also be changed when desired.

***FMA fact**: When Session Reliability is enabled users will be automatically reconnected as soon as the network connection is reinstated, and they will do so without needing to re-authenticate. Configuring the 'Auto client reconnect authentication' policy to prompt users to re-authenticate can change this behaviour.*

Auto client reconnect is a feature used to detect unintended disconnected ICA sessions and will reconnect the user session automatically. As mentioned, users then do or do not have to re-authenticate, depending on how you configure the accompanying policy. If both Session Reliability and Auto client reconnect are used they will work in sequence, meaning that first

the Session Reliability policy will be applied, and as soon as the user session disconnects because the SR time-frame has elapsed, the Auto client reconnect policy will kick in.
As an alternative to Session Reliability you can also configure ICA Keep-Alive. This feature prevents a session from going into a disconnected state when a session seems broken. If configured, it will send a constant stream of ICA packages every few seconds (configurable) to detect if the user session is active. Only after the session has been marked as inactive will the session be put in a disconnected state. However, in practice Session Reliability is almost always preferred over ICA Keep-Alive.

Citrix HDX

I just wanted to briefly touch on HDX (High Definition Experience) since it is more than 'just' the ICA protocol and it is often misunderstood. In fact, HDX technologies are built on top of the ICA protocol: and they are not meant as a replacement at all. HDX technologies extend the ICA protocol. According to Citrix: HDX technologies offer a set of capabilities that deliver a 'high-definition' experience to users of centralised applications and desktops, on any device and over any network.

It does this by trying to optimise the user experience, decrease the overall bandwidth consumption, and increase the user density per server. The HDX portfolio offers several innovative and industry-first technologies further enhancing and extending ICA, still the Nr. 1 remoting protocol in the industry today. A couple of examples of HDX technologies are: Flash and Windows media redirection, 4K monitor support, HDX 3D Pro GPU acceleration and sharing support (separate VDA), acceleration of printing and scanning, optimisation of USB traffic and more.

***FMA fact**: Remember, Citrix HDX isn't a replacement for the ICA protocol. HDX technologies are meant as an extension and as such operate on top of the ICA protocol.*

Putting it all together

Now that we have looked at the various individual FMA components and services involved, let's have a closer look to see how everything works together. What happens when a user logs in, authenticates and so on.

Resource enumeration

When a user logs in for the first time, meaning that there are no active and/or disconnected sessions lingering around somewhere, right after the user is authenticated the resource enumeration process kicks in and will eventually show the user its assigned resources. That's why the user authentication process and resource enumeration basically go hand in hand. Let's put the two together and see what happens, I'll only use StoreFront in my example.

Resource subscription

Due note that when using StoreFront users will first have to select and subscribe to resources (applications and/or desktops) from the store before they show up on their main home screen and can be launched. When using so called Keywords, Administrators can pre-subscribe users to certain core resources so that their home screen won't be completely empty when logging in for the first time. This also works for assigned Desktops. Use Keywords: auto with the application and/or desktop of choice.

External user authentication through NetScaler

Let's take it step by step and see what happens when someone logs in externally. I'll assume that your NetScaler Gateway is set up and configured to integrate your StoreFront server(s), you have Receiver installed, SSL certificates are present, and that a STA / XML / Broker service address (Delivery Controller) and a domain controller for authentication purposes are also configured.

Perhaps you want to load balance your StoreFront and/or Delivery Controllers by creating and configuring virtual load balance servers on the NetScaler, adjust the theme of the NetScaler's Web Interface and finally, you'll probably have to configure your StoreFront deployment to accept pass-through authentication from NetScaler. Where port Nr. 443 (SSL) is used you can also use port 80, although 443 is recommended.

1. A user opens a web browser and connects to the external URL of the NetScaler Gateway (preferably using SSL over port Nr. 443). Here he or she will fill in his or her username and password. A locally installed Citrix Receiver can also be used to establish a direct connection to the NetScaler Gateway. Citrix Receiver uses so called Beacons to determine if a connection is internal or external and handles it accordingly. Check the (red) link for some more detailed information around Beacons and the discovery process.
2. During the login/authentication process an EPA (End Point Analyses) scan might be performed as part of a SmartAccess/SmartControl policy, for example, or NetScaler multi-Factor a.k.a. nFactor authentication could be configured (optional as of NetScaler 11.0 build 62.x and onwards).
3. Eventually the NetScaler will authenticate the user credentials (session ticket) against Active Directory, preferably using TCP port Nr. 636 (SSL) based upon the configured Authentication Policy. This could also involve two-factor/RADIUS authentication, which is basically considered a must have/minimum these days. Like StoreFront, the NetScaler has its own Authentication Service.
4. Once authenticated, the NetScaler will assign a session cookie (note that it does not built/assign the authentication token as part of the initial authentication process), which will be used for any potential subsequent client requests.
5. Next the user session and the user authentication credentials get redirected to StoreFront (based upon the configured Session Policy) where it will perform a call-

back to the NetScaler (Gateway Virtual Server) that handled authentication to validate the user in the first place. The authentication details will then be send to the StoreFront Authentication Service, which is similar to the Authentication Service of the NetScaler mentioned earlier.

6. This is where the earlier mentioned authentication token is built/generated — by default the StoreFront Authentication Service will take care of this. However, as of StoreFront version 3.0, Citrix re-introduced XML-based user authentication. By simply running a few PowerShell scripts user authentication falls back to the XenDesktop/XenApp XML service, which is equal to how Web Interface used to handle things. Particularly useful when StoreFront is not in the same domain as XenDesktop / XenApp and when it is not possible to set up an Active Directory trust, or multiple. Just be aware that this method will be disabled by default. As of StoreFront version 3.5 and upwards PowerShell is no longer needed to enable XML based user authentication, it can be enabled and disabled directly from the StoreFront management console.
7. From here the user credentials will be forwarded, as part of a XML query, to the configured Broker (XML) service on one of the available Delivery Controllers. Both these transactions will use port Nr. 80 by default, which of course can be changed to 443 (SSL).
8. In between, StoreFront will check its local data store for any existing recourse subscriptions and stores these in memory.
9. The Broker (XML) service will again contact a domain controller (using port Nr. 389 by default, change to 636 for SSL) to validate the user credentials, note that this is different to the user authentication process, as we've established earlier. During this process, it will find out to which security groups (SID's) the user belongs.
10. You basically authenticate/validate against LDAP three times:
11. Through NetScaler (session cookie) -> Active Directory, followed by a redirection of the authentication credentials over to the StoreFront server.
12. Through Storefront, either using the SF Authentication Service or via SF to the XML Service on one of your Delivery Controllers -> Active Directory, this will generate/built the authentication token.
13. Through the XML Service (validation) -> Active Directory, to find out the accompanying security group SID's used for resource enumeration.
14. With this information the Delivery Controller, or Broker (XML) service will contact the Central Site Database to find out which resources have been assigned to the user. It does this over port Nr. 1433 / 1434.
15. This data will then be gathered and send back to the StoreFront server in the form of an XML formatted file, through/using the Broker (XML) service.
16. Based on this information StoreFront will generate a web page containing all the assigned resources, which will be routed through the NetScaler Gateway and presented to the user. The users home screen will be populated with any pre-subscribed resources (Keywords). Depending on how you connected, your resources will be displayed either directly using a Receiver for web sites or you'll find them within the locally installed Citrix Receiver instead. The user will be able to browse its

own personal app store for any assigned resources to which he or she can subscribe and then launch.

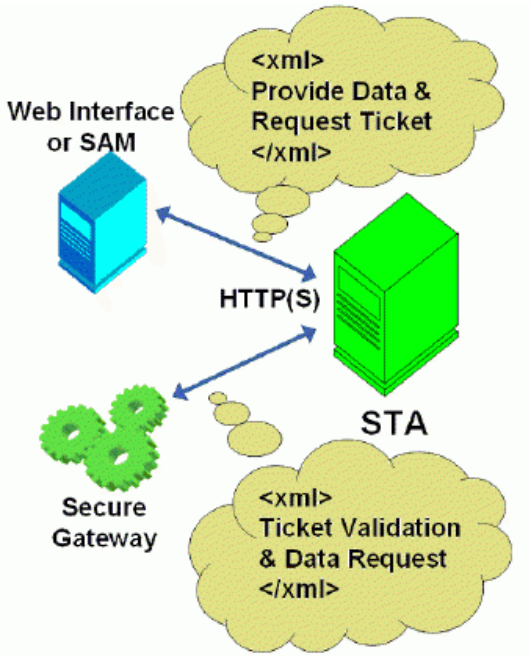
FMA fact: *If you don't enable authentication on the NetScaler's login page the NetScaler will contact StoreFront and the user will be presented (through the NetScaler) with the StoreFront login page (Receiver for web sites). The user fills in his or her credentials and authentication will be handled by StoreFront.*

The (external) launch process

Here we basically pick it up where we left off at the end of the resource enumeration process as explained above. Just as with the authentication process, there are some differences in how a recourse is launched with, and without a NetScaler in between. Also, when launching a Hosted Shared Desktop (XenApp) or a published application, as opposed to a VDI virtual machine (XenDesktop) there is an extra load balance step involved as well. Let's see what happens when we launch a published Hosted Shared Desktop trough NetScaler.

The Secure Ticket Authority

Before we continue… You might have heard about something called the STA, or the Secure Ticket Authority in full. It was first introduced with one of the earlier Secure Gateway editions over twelve years ago. It (the STA) runs as a service and is part of the Broker Service on the Delivery Controller just like the XML service. During the resource launch sequence the StoreFront server as well as the NetScaler will both need to be able to communicate with the STA. As such you need to configure the NetScaler and the StoreFront server(s) or Web-Interface server(s) to point to the exact same XML/STA service(s)/Deliver Controller(s).



FMA fact: *The NetScaler Gateway uses the STA to guarantee that each user is successfully authenticated. If users have valid STA tickets, the gateway assumes that they passed the authentication checks at the web server and should be permitted access. It prevents computers from the 'outside' to have knowledge about the network on the 'inside' of the datacenter and it authorises the NetScaler Gateway ICA Proxy to set up a connection from the 'outside' to the 'inside'. It basically specifies where an outbound connection can connect to on the 'inside'.*

Once a user launches a resource, externally (or internally for that matter) through NetScaler Gateway, at one point a secure ticket will be requested. As we will see shortly the STA ticket will eventually end up in the launch.ica file generated by StoreFront and/or Web-Interface.

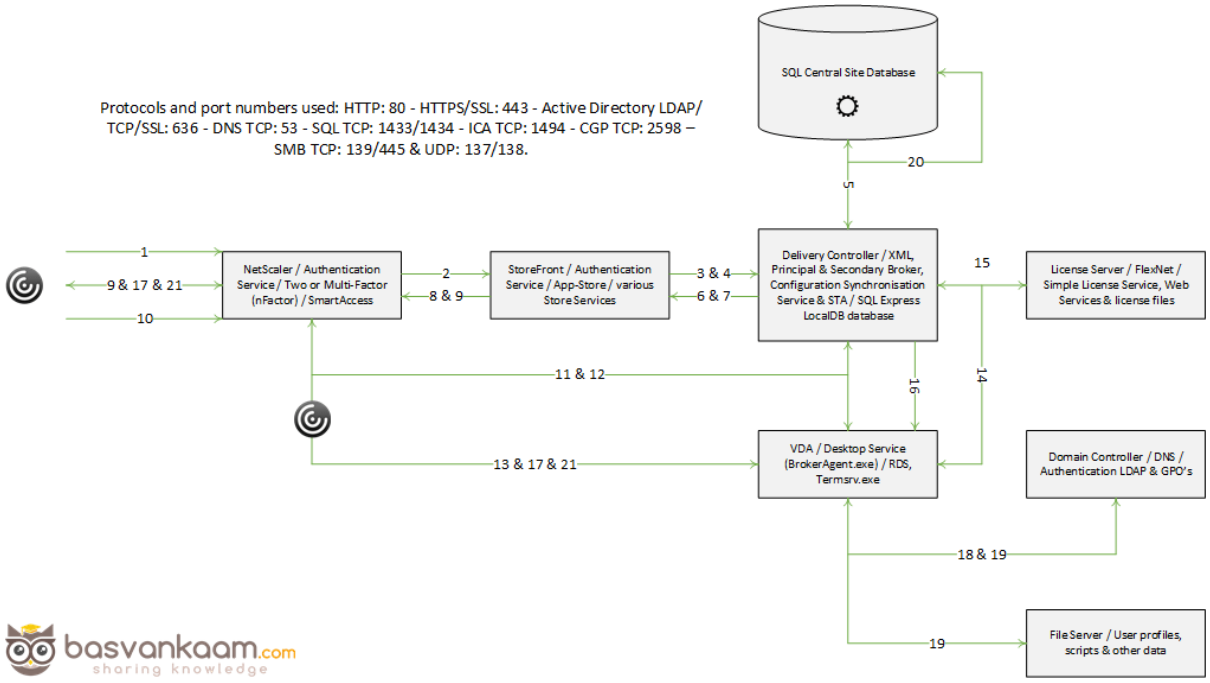
Once generated, the Delivery Controller hosting the STA service will hold the STA ticket information in memory for a configurable amount of time. As soon as a secure session is established the NetScaler Gateway responsible for handling the session only has to check the STA ticket (as part of the .ica launch file) with the STA service that originally generated the ticket. It (the STA service) does this from memory where the ticket was stored after it was created and send back to the StoreFront server as part of the XML formatted file mentioned earlier.

FMA fact: *The STA is only used when traffic traverses a NetScaler, so you don't have to worry about the STA service and its tickets when authentication takes place internally through StoreFront, for example.*

1. Assuming that the login, authentication and enumeration process finished without any issues (see above) the user is now free to subscribe to and launch any applications and/or desktops that might have been assigned to him or her. As an example, let's assume that the user wants to launch a (XenApp) Hosted Shared Desktop session a.k.a. a published desktop.
2. After the user clicks the icon the launch request is send to the NetScaler Gateway from where it will be forwarded to the StoreFront server (1 & 2) see image below.
3. The StoreFront server will contact the Broker (XML/STA) service, or Delivery Controller, to find out if and where the resource is available and where it can be best started (3). This is where the well-known XenApp load balancing mechanism comes into play. Which as of XenApp 7.x needs to be configured through policies (or use the defaults).
4. During this time, the StoreFront server will also request an STA ticket from the Broker (XML/STA) service (4). It will include the user, domain and resource name it wants to start. It will also request a 'least loaded' server as part of the load balancing process.
5. The Broker (XML/STA) service will query the Central Site Database (ports Nr. 1433 and 1434) to find out which server can offer the requested resource (5), which is also referred to as the current Farm state. The Delivery Controller will than use this

information together with its load balance algorithm to decide which server to connect to.

6. At this time, the Broker (XML/STA) service will create the STA ticket mentioned earlier. This will include information on the server and resource to connect to, amongst other information as discovered in the previous steps mentioned.
7. Next, the Broker (XML/STA) service will send this information back to the StoreFront server in the form of an XML formatted file (6 & 7).
8. Based on this information the StoreFront server will generate a launch.ica file (it uses the default.ica file as a template) containing the STA ticket and a whole bunch of other connection properties that are, or might, be needed (8). This will also include the FQDN/DNS name of the NetScaler Gateway itself.
9. StoreFront passes on this information down through the NetScaler Gateway onto the locally installed Receiver (9) which initiated the connection to begin with.
10. The locally installed Receiver will read and auto launch the launch.ica file to set up a connection to the NetScaler Gateway over 443 / SSL (10).
11. From here the NetScaler Gateway will first contact the Broker (XML/STA) service (this address is configured on the NetScaler as well) to verify if the earlier generated STA ticket, as part of the launch.ica file is still valid (11).
12. The Broker (STA) service will validate the STA ticket from memory. Once verified it will send back the IP address, port Nr. Resource name etc. of the machine and the resource it needs to connect to (12). Once done the STA ticket will be deleted.
13. The NetScaler Gateway will set up a new ICA connection using port 1494 (ICA) or 2598 (CGP – Common Gateway Protocol) depending on its configuration (13). Soon to include 'HDX Enlightened Data Transport' I'm sure.
14. The VDA will verify its license file with the, or a Delivery Controller (14).
15. The Delivery Controller checks with the Citrix License server to verify that the end user has a valid ticket (15). This will also be done for the Microsoft (CAL) licenses, regarding the Hosted Shared Desktop session and any published applications that might be involved.
16. Now, any applicable Citrix policies will be passed onto the VDA applying them to the session (16).
17. The Hosted Shared Desktop session is launched and the NetScaler Gateway acts as a proxy between the user and the XenDesktop resource in the data center (17).
18. User (Windows) authentication takes place between the domain controller and the Citrix Worker / Session Host (18).
19. The Citrix session will initialize; the Windows welcome screen appears. At this point the user profile is loaded, Group Policies (GPO's) are applied, scripts will be executed, drive and printer mappings are established and so on (19).
20. Somewhere in between the session/connection information will be passed on and registered in the Central Site Database where it will be used for future load balance purposes (20).
21. And finally, the Hosted Shared Desktop will be fully launched (21).



Internal user authentication through StoreFront

What happens when a user authenticates internally, directly to StoreFront? As you will see, besides the NetScaler in between both methods look very similar. Let's have a look. Same rules apply here, use port Nr. 443 where you can.

1. A user opens a web browser and connects to the internal StoreFront URL where he or she will fill in his or her username and password. This method is also referred to as Receiver for web sites as mentioned above (don't confuse this with the HTML 5 based Receiver for web, they're not the same). A locally installed Citrix Receiver can also be used to establish a direct connection to StoreFront, which is probably the preferred method whenever possible. The earlier mentioned (NetScaler) Beacon functionality applies here as well.
2. Next the StoreFront authentication service will pick up the user credentials and contact a domain controller to authenticate the user in Active Directory over TCP port Nr. 389. Here I'd like to note that if domain pass-through authentication is enabled on the StoreFront server, this step would automatically be skipped. This is where the authentication token is built/generated — by default the StoreFront Authentication Service will take care of this. However, as of StoreFront version 3.0, Citrix re-introduced XML-based user authentication. By simply running a few PowerShell scripts user authentication falls back to the XenDesktop/XenApp XML service, which is equal to how Web Interface used to handle things. Particularly useful when StoreFront is not in the same domain as XenDesktop / XenApp and when it is not possible to set up an Active Directory trust, or multiple. Just be aware that this method will be disabled by default. As of StoreFront version 3.5 and upwards PowerShell is no longer needed to enable XML based user authentication, it can be enabled and disabled directly from the StoreFront management console.

3. Once authenticated the user credentials will be forwarded, as part of a XML query, to the configured Broker (XML) service on one of the available Delivery Controllers. Both these transactions will use port Nr. 80 by default, which of course can be changed to 443 (SSL).
4. In between, StoreFront will check its local data store for any existing recourse subscriptions and stores these in memory.
5. During the next phase the Broker (XML) service will again contact a domain controller (default over port Nr. 389) to validate the user credentials, note that this is different to the user authentication process, as we've established earlier. During this process, it will find out to which security groups (SID's) the user belongs.
6. Here you basically authenticate/validate against LDAP two times:
7. Through Storefront, either using the SF Authentication Service or via SF to the XML Service on one of your Delivery Controllers -> Active Directory, this will, as mentioned earlier generate/built the authentication token.
8. Through the XML Service (validation) -> Active Directory, to find out the accompanying security group SID's used for resource enumeration.
9. With this information the Delivery Controller, or Broker (XML) service, will contact the central Site Database to find out which resources have been assigned to the user. It does this over port Nr. 1433 / 1434.
10. This data will then be gathered and send back to the StoreFront server in the form of an XML formatted file, through/using the Broker (XML) service.
11. Based on this information StoreFront will generate a web page containing all the assigned resources, which will be routed through the NetScaler Gateway and presented to the user. The users home screen will be populated with any pre-subscribed resources (Keywords). Depending on how you connected, your resources will be displayed either directly using a Receiver for web sites or you'll find them within the locally installed Citrix Receiver instead. The user will be able to browse its own personal app store for any assigned resources to which he or she can subscribe and then launch.

The (internal) launch process

Now that we've seen which steps are involved when launching a resource externally, a Hosted Shared Desktop in this case, let's have a look and see what happens when we launch a pooled VDI virtual machine internally. Since we do not have to 'deal with a NetScaler, we won't have to worry about the STA as well. After this we will have looked at an external and internal resource launch, a HSD, which is comparable to a published application, and a VDI virtual machine. Again, user authentication and resource enumeration has successfully completed, here we go (again).

1. Assuming the login, authentication and enumeration process finished without any issues (see the above sections) the user is now free to subscribe to and launch any applications and/or desktops that might have been assigned to him or her. As mentioned, this time we will launch a pooled VDI virtual machine. I'll assume that the

VM is pre-subscribed and already present on the user's home screen, never mind how we connected: locally installed Receiver or using the Receiver for web sites.

2. After the user clicks the icon the StoreFront server will contact the Broker (XML) service on the Delivery Controller, to check if any registered VDA's are available. It does this by communicating with underlying Hypervisor platform (your Host Connection) through the Host service on the Delivery Controller.
3. If needed it will first start / boot a VM. It's not uncommon to pre-boot a few VM's, since, as you can probably imagine, this will positively influence the overall user experience.
4. Next the Delivery Controller, or Broker (XML/STA) service, will contact one of the VDA's and sends a startlistening request. By default, the VDA isn't listening for any new connections on port Nr. 1494 or 2595 until it gets notified that a user wants to connect.
5. As soon as the VDA is listening, the Broker (XML/STA) service will send this information back to the StoreFront server in the form of an XML formatted file (it isn't an actual XML file).
6. Based on this information the StoreFront server will then generate a launch.ica file (it uses the default.ica file as a template) containing the IP address of the VDA and a whole bunch of other connection properties that are, or might, be needed. This is send down to the user, or better said, the Citrix Receiver.
7. The locally installed Receiver (or HTML 5 based Receiver) will read and autolaunch the launch.ica file initiating a direct connection from the users end-point to the VDA using the ICA protocol over port 1494.
8. At the same time, the installed VDA will verify its license file with the Delivery Controller.
9. The Delivery Controller checks with the Citrix License server to verify that the end user has a valid ticket.
10. Now, any applicable Citrix policies will be passed onto the VDA applying them to the session.
11. User (Windows) authentication takes place between the domain controller and the Citrix VDI VM.
12. The Citrix session will initialize; the Windows welcome screen appears. At this point the user profile is loaded, Group Policies (GPO's) are applied, scripts will be executed, drive and printer mappings are established and so on (19).
13. Somewhere in between the session/connection information will be passed on and registered in the Central Site Database.
14. And finally, the VDI session is fully launched.

To find out what happens inside the VDA during launch time, scroll up to the desktop and Server VDA sections.

HTML 5 receiver to the rescue

If for whatever reason you are unable or not allowed to install a Citrix Receiver locally, Citrix offers the Receiver for HTML 5. You will still be able to connect to StoreFront / NetScaler and launch your resources without any loss of functionality.

Although not enabled by default, StoreFront has a build-in HTML 5 based Receiver, which will kick in at launch time. It does this by fetching the HTML 5 engine from the StoreFront and making it part of the local browser.

Note that you must use a HTML 5 supported browser for this to work. Basically, your browser becomes your Receiver handling the launch.ica file. When you close the browser, you close the session. Even when your users will have Receiver installed you can enable it anyway as it will function as a fall-back mechanism.

Broker, XML, STA and Principal

Be aware that the STA (service) is also part of the Broker service, and has been as of Presentation Server 4.0. Before that it was written as an ISAPI extension for Microsoft Internet Information Services, or IIS. I also highlighted the so-called XML service multiple times. I put the XML and STA services between brackets because as of XenDesktop 4.x the XML service (ctxxmlss.exe) has been rewritten in .NET and became part of the Broker service.

The Broker service is build up out of three separate services, all handling different tasks, it brokers connections, it enumerates resources and it acts as the Secure Ticket Authority, generating and validating STA tickets. With the re-introduction of the LHC the Broker services is now also knows as the Principal Broker Service. Every two minutes the (Principal) Broker Service will be checked for configuration changes. If a configuration change has been detected it will be copied over, or synchronised to the High Availability Service/Secondary Broker Service.

FMA fact: *Make sure that the Broker (XML/STA) service on the NetScaler and the Storefront server is configured identically. The same applies to the load balance/fail over order in which you configure them.*

Citrix NetScaler Gateway, the basics

As promised, as a bonus here's my number one NetScaler blogpost (viewed over 55.000 times): I don't want to spend too much time talking about the different kinds of editions and or licenses available, since these keep changing as well. Throughout this chapter, I'd like to focus on some of the basic terminology and traffic flow that comes with the NetScaler Gateway edition providing our users with secure remote access. This (the Gateway edition) is probably one of the most popular NetScaler implementations today, although, and as you

might know, the NetScaler's ADC edition also has the Gateway functionality build-in and can provide us with a bunch of additional features as well.

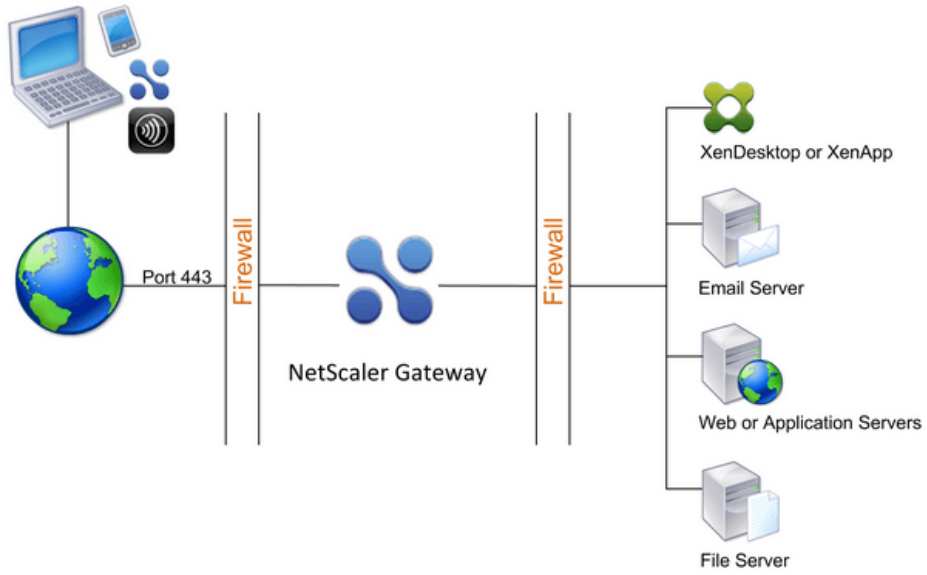
NetScaler ADC and Gateway

Before we continue, first a short word on the two NetScaler editions available today. Most of the confusion starts with the terms; Citrix NetScaler and Citrix NetScaler Gateway, although they sound very similar, and they do have an overlap, there are some distinct differences depending on the licenses used.

Citrix NetScaler refers to their Application Delivery Controller, or ADC, line of products, while the NetScaler Gateway, formerly known as the Citrix Access Gateway, or CAG, is primarily used for secure remote access. You basically buy a 'normal' NetScaler but with limited functionality due to the NetScaler Gateway License you upload. NetScaler ADC's can do much more than 'just' remote access, they can be used for load balancing and HA, content switching, application (SSL) offloading, application firewalling, cloud connectivity, hybrid cloud solutions and (a lot) more.

General use

As mentioned, the NetScaler Gateway is used and configured to provide our users with secure remote access into our secure corporate network. As such it is physically placed within our DMZ, or Demilitarized Zone in full, where it sits in between the Internet and our secure corporate LAN fronted by at least one or two firewalls as shown in the overview below (got this from Citrix.com).



Some terminology first

Before I show you how traffic flows and what actually happens when an external user connects up to the NetScaler Gateway, I first want to spend a few minutes explaining some of the terminology you need to know and understand before we continue.

The NetScaler uses vServers (virtual servers) to deliver different kinds of services, in this case the vServer will be configured as a gateway server. Just remember that you can configure multiple independent vServers on the same NetScaler serving different purposes, like a load balancing or SSL offload vServer for example.

The NSIP address (NetScaler IP Address) is the IP address which is used by the Administrator to manage and configure the NetScaler. It is mandatory when setting up and configuring the NetScaler for the first time, there can only be one NSIP address, it cannot be removed and when it's changed you will have to reboot the NetScaler.

A SNIP (Subnet IP Address) is used for server side connections, meaning that this address will be used to route traffic from, or through, the NetScaler to a subnet directly connected to the NetScaler. The NetScaler has a mode named USNIP (Use SNIP), which is enabled by default, this causes the SNIP address to be used as the source address when sending packets from the NetScaler to the internal network. When a SNIP address is configured, a corresponding route is added to the NetScaler's routing table, which is used to determine the optimal route from the NetScaler to the internal network.

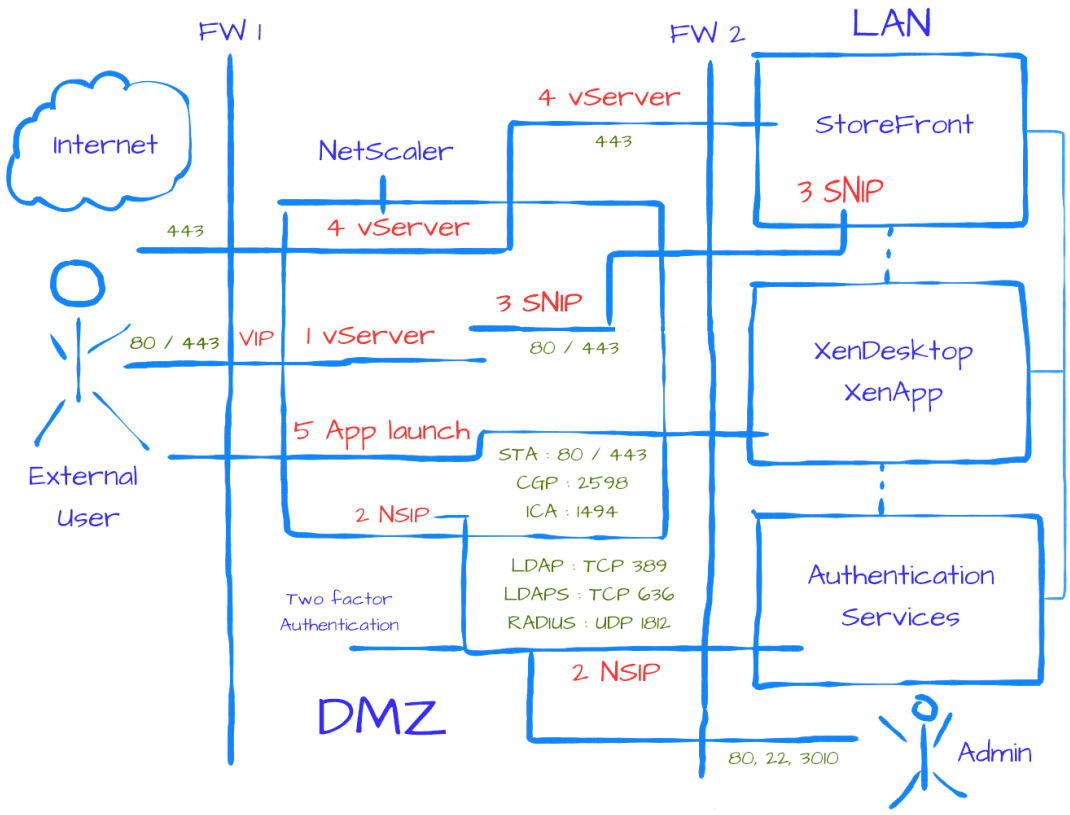
If it detects the SNIP address to be part of the route it will use it to pass-through the network traffic using the SNIP address as its source. A SNIP address is not mandatory. In a multiple subnet scenario, you will have to configure a SNIP (or MIP, I'll discuss this in a minute) address for each subnet separately. Also, when multiple SNIP addresses are configured on the same subnet, they will be used in a Round Robin fashion.

A MIP (Mapped IP Address) is similar to the SNIP address mentioned above. MIP addresses are used when a SNIP address isn't not available or when USNIP (Use SNIP) is disabled. In that case, it will also be used as the source IP address. Only when the configured MIP address is the first in the subnet it (the NetScaler) will add a route entry to its routing table.

A VIP address (Virtual IP Address) is the IP address of a vServer that the end users will connect to, and through which they will eventually be authenticated etc. For now, just remember that the VIP address is never used as the source IP and thus isn't involved in back-end server communication, instead this will always be handled by a SNIP and or MIP address, where often SNIP addresses are used over MIP's, but they can be mixed and used to connect to the same IP subnet even, again, Round Robin will then be used to determine the most optimal route.

Traffic flow

Now that we know what terminology is involved let's have a look and see how traffic and communications flow through the NetScaler and how users get authenticated. Hopefully the overview below will help in clarifying some of the concepts mentioned throughout this article. I hope you can appreciate my (free) drawing skills, I thought it was kind of cool to be honest – see below.



Up close and personal

Ok, so what happens? Let's take it step by step. Due note that I'm primarily focussing on the NetScaler interaction here, as such I left out both the application enumeration process (XML querying, XML and STA file generation etc.) as well as the application launch sequence (which system can provide the resource, least load, .ICA file generation etc.).

An external user will contact the NetScaler Gateway over port 80 or 443 (preferred). It will connect to the externally accessible virtual IP (VIP) address of the NetScaler's (Gateway) vServer. This is indicated as the VIP followed by the 1 vServer. Once a connection is established you have a few options, for example, using a SNIP address the (unauthenticated) user could be connected to the StoreFront server residing on the corporate (internal) network where authentication needs to take place (not displayed). A valid option but not as secure as we would like it to be right? Instead, we would like user authentication to take place on the NetScaler within our DMZ.

Let's assume authentication takes place on the NetScaler. The user's credentials are forwarded using the NetScaler's IP address, or NSIP, indicated as 2 NSIP, to your internal authentication services, Active Directory in most cases, where they will be validated (or not). But wait, let's do one better. Once validated, and still part of step 2 NSIP, we throw in something called two factor authentication, using SMS passcode tokens for example. This way every user will have to fill in his or her username and password plus an additional auto generated token code which will expire every few minutes (configurable), extremely secure.

Once the user is authenticated, the authentication services will pass through the user credentials to the StoreFront server. In step 3 SNIP, the already authenticated user will connect to our internal StoreFront server where it will enumerate the user's applications and or desktops.

Next, this information will travel back into the NetScaler and through the Gateway vServer onto the user's screen. As indicated in step 4 vServer.

Finally, when the user starts an application, I left this part out as well, the StoreFront server will eventually generate a so called .ICA file which is send back to the user's device and is used to connect the user directly to the requested resource on one of the XenDesktop / XenApp application servers. During the last phase of setting up this connection the Gateway server will check up on the earlier generated STA file to validate the session, after that the application or Desktop will be launched as indicated in step 5 app launch.

Wrap up

This concludes the ultimate Citrix XenDesktop 7.x cheat sheet, version 2.0. I hope you've found it of some use. If you haven't purchased a copy of my book this is probably the next best thing. I really enjoyed putting it together, and even though I could reuse some of my existing material I took caution in re-reading and re-writing almost all of it. I also realize it's far from complete, if that is at all possible.

As always, any feedback you have is greatly appreciated, and please make sure to pay it forward. I truly hope that version 2.0 will get the same attention as version 1.0 did before it – thank you for your support.

About the author

In case you might be wondering who initiated this cheat sheet, well, me. My name is Bas – or Sebastiaan van Kaam. I am 38 at the time of writing, Dutch, father of two beautiful daughters, engaged with Tineke and together we own a home in the Netherlands.
I started my IT career 17 years ago at company named Commit-Arbo, as a Helpdesk technician and worked my way up to system engineering and architecting mostly through self-study, attending various conferences, seminars and webinars all of which I enjoy doing still. I have applied Richard Branson's quote 'Screw it, let's do it' more than ones before stepping out of my comfort zone – the only way to truly challenge and improve yourself.

Currently I am the CTO for a Dutch company named Detron where my responsibilities include but are not limited to: Advising on relevant technology trends and products, strategic partnerships and market developments, including the necessary training and certification needs within Detron. Further development (and chairman) of the Detron Technology Board. Member of the Detron Solution Board. Support in the, ongoing formation of the Detron (technical) vision/strategy and potential new services in consultation with both the Solution as well as the Technology Board. Engaging with partners and customers regarding their technical as well as non-technical queries around current ICT issues and Detron propositions. Provide input for, write and coordinating blogs, reference cases and white papers. Present at internal as well as external IT orientated events and conferences.

From a technical perspective, I specialize in (or used to do) designing, building, maintaining, troubleshooting and optimizing SBC & VDI (Citrix, VMware & Microsoft) oriented infrastructures for mid-sized to larger enterprises with a special interest in Hyperconverged solutions and the EUC space in general. But above all, I just like to talk about technology, advice customers and try to stay updated with everything that is going on within the IT landscape, a challenge to say the least.

I am an active member of the community and as such I like to share what I know, learned and think, which is something I do on a regular basis over at basvankaam.com my own personal website / blog. Besides that, I am also a frequent presenter on national as well international events, like The Dutch, Polish and Norwegian Citrix User Group events, E2EVC, Citrix Synergy, VCNRW, the Virtual Expo and more. Throughout the year, I also organize and present multiple technical orientated (pizza) sessions for my colleagues over at Detron.

In June of 2016 I published my first book: **Inside** Citrix - The FlexCast Management architecture. Over 500 pages of FMA goodness. It is available in both paperback as well as Kindle, you'll find it on all major Amazon channels – the feedback I received up till now has been overwhelming to say the least.

As of February 2016, I am a proud member of the Citrix Technology Professionals (CTP) group (one of 50 in total) and have been named a Nutanix Technology Champion since 2014 – the founding year. I'm also part of the Atlantis ACE community and have been recognised as an iGel Tech Insider, Citrix Subject Matter Expert and Technology Advocate and I am part of the Inside Track for EUC/VDI champions vips.

The myCUGC (Citrix User Group Community) drew my attention shortly after Citrix Synergy in 2015, where it was founded. As a result, I am now a member of the myCUGC content working group, one of the myCUGC forum moderators and part of the so-called founding member group of the Citrix community program named the Citrix Technology Advocates (CTA).

As for recreation, besides writing about and researching technology I like to go out for a run at least three to four times per week and just staying fit in general. However, above all I love spending time with my kids, Julia and Sophie and girlfriend Tineke, in the end it is all about a healthy life/work balance.

Thank you and all the best, Bas.

basvankaam.com
sharing knowledge

- Web : www.basvankaam.com
- E-mail : basvankaam@gmail.com
- LinkedIn : LinkedIn.com/in/basvankaam
- Twitter : @BasvanKaam