

GSA Valve Security Framework – Kerberos

Google Enterprise EMEA

One of the most important benefits of deploying a complete search platform is the quick and accurate access to the information. As this is one of the main company's assets, the access to such information should be protected with the corporate authentication and authorization mechanisms the same way as the applications are already integrated.

Microsoft Windows has become one of the most common corporate network platforms. Windows 2000 started to support the use of Kerberos tickets as the native and silent authentication mechanism. Since then, the numbers of companies that use such a technology has increased dramatically and have extended Kerberos to non-Microsoft platforms as well.

Kerberos is today a common security requirement in most of the companies, as it offers big benefits, and that's why the GSA Valve Security Framework supports it exactly the same way as forms-based authentication. It's able not only to silently authenticate a user using his/her Kerberos ticket (TGT) but also to impersonate him/her forwarding such credentials to the Kerberos-compliant backend applications in a completely transparent way to the user.

In this document you'll learn how to successfully deploy a Kerberized scenario using the Valve. It mainly explains how to do it using Microsoft platform and its Kerberos implementation but it can be easily extended to other platforms like Linux as the main configuration steps remain the same. The main difference resides on the Kerberos tools and commands as they could be distinct.

This guide explains the basics about Kerberos and how to configure the Security Framework in this scenario but, as this authentication technology is quite complex, you would probably find some Kerberos errors that you need to investigate. This guide does contain a troubleshooting chapter with the basics about how to further analyze errors but cannot include all the cases. We strongly recommend you to be in contact with your Network Administrators as you'd need their expertise and probably some steps mention here requires having Administrator privileges they only have.

The Kerberos configuration explained here is so generic that can be used both with the Valve Security Framework's SAML interface or the Forms Based one. Have a look at the other technical documents about this framework to get more information on each interface.

Introduction to Kerberos

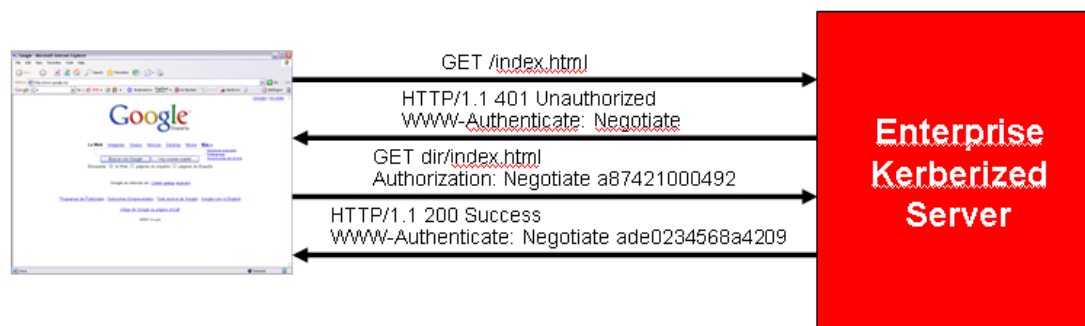
Kerberos is a network authentication protocol created by the Massachusetts Institute of Technology (MIT). It is designed to provide strong authentication for any kinds of applications by using secret-key cryptography, where the users/services have a pair of keys, one public known by everybody and one private that is only known by the owner.

The Kerberos protocol uses this strong cryptography so that a client can prove its identity to a server (and vice versa) across an insecure network connection. After a client and server have used Kerberos to prove their identity, they can also encrypt all of

their communications to assure privacy and data integrity as they go about their business.

Services and users trust on the corporate Kerberos server that manages all the communications that require this authentication technology. This server acts as the KDC (Key Distribution Center) that controls the authentication in all the layers throughout the company and sends the tickets valid to authenticate users and let them access to the services.

The integration between a web-enabled GSA Valve and Kerberos is done through a standard implementation, SPNEGO. It establishes the negotiation mechanisms both the server and the client must follow to successfully authenticate users on HTTP protocol. Below you just have a sample user-transparent negotiation using HTTP headers:



There are some requirements in order to have such a successful negotiation. The main one is both the client (browser) and the server have to be configured in such a way that permits the Kerberos authentication. In the case of the browser the majority of them can easily take Kerberos tickets from the desktop and use them whenever a server requests them. So, the server also needs to be able to negotiate using WWW-Authentication header but also it has to be kerberized using a service ticket for it as well.

Prepare your Environment

The first thing you need to do is checking your Kerberos environment is working smoothly and prepares it for running the Valve appropriately. Check the following steps before proceeding:

Test if Kerberos works

If you are using Windows, Kerberos should be working perfectly. You can do a basic Kerberos check using *kinit* tool. From one of the computers in your network that have access to the KDC (Key Distribution Center), in Windows is usually the Domain Controller, check the following using your user account (f.e: cesarlazaro@ENTERPRISE.COM):

```
kinit cesarlazaro@ENTERPRISE.COM
```

If everything is Ok, the command will ask you for your domain password and terminates without an error message.

This command will show you the initial ticket you got from the KDC if you execute it

without any argument.

Kerberos Configuration File

Your environment should have ready a standard Kerberos configuration file. This usually is present by default at the following locations:

- *Windows* (krb5.ini): c:\Windows or c:\Winnt
- *Linux/Unix* (krb5.conf): /etc/krb5.conf

Such a configuration file should contain the KDC location, realms, etc. The following example shows you how a Kerberos configuration file looks like. In this example we just take the domain *ENTERPRISE*, but in your case it should be something like *domain.corporate.com*, that completely maps to your network domain.

```
[libdefaults]
    default_realm = ENTERPRISE

[realms]
    ENTERPRISE = {
        kdc = ad.enterprise
        default_domain = ENTERPRISE
        admin_server = ad.enterprise
    }

[domain_realm]
    .enterprise = ENTERPRISE
    enterprise = ENTERPRISE
```

Store this file at the default Kerberos configuration location for the JDK. In Windows they are:

- JDK 5 and previous releases: c:\WINNT
- JDK 6 and upwards: C:\Windows

Java Developer Kit (JDK)

We strongly recommend you the use of JDK version 6 as it is bundle with specific SPNEGO features that will help you on debugging Kerberos. Make sure this is installed and Tomcat is making use of it.

GSA Valve Security Framework host

You have to choose where the Security Framework is going to be installed and it has to be ready to install on it the application server. Take into account it should be part of the default domain but cannot be installed on the domain controller, otherwise Spnego/Kerberos will not work as expected.

Configure your Browser for Kerberos

Browsers will just negotiate through user's Kerberos tickets only if they trust the server. It means you have to add some configuration information to any browser that users are going to connect from. The configuration differs depending on the browser. You can find some information on how to set up different browser at http://spnego.ocean.net.au/documentation/browser_configuration_for_spnego.html.

How to Setup the Valve for Kerberos

The GSA Valve Security Framework has been fully integrated with Kerberos in such a flexible way that it's possible to have different deployment scenarios. This is driven by the specific Kerberos configuration parameters you can find at the Valve configuration file.

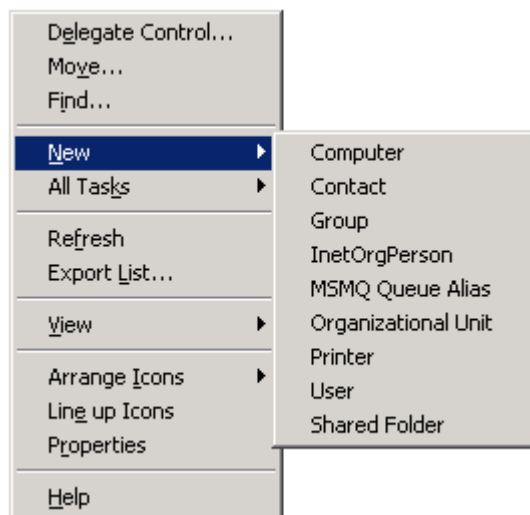
The first configuration step is preparing the environment to configure Kerberos on the Valve instance. This is done creating the Kerberos service ticket that maps with the host where the Valve is running on, as this is the one the browser will request. A valid service ticket is compiling of the HTTP protocol, as this is the one we're just using, the fully qualified server name and the Kerberos/Windows domain. For example:

```
HTTP/valve.google.com@ENTERPRISE.GOOGLE.COM
```

If you are using a Windows server, this service could be already configured in your environment (you can know it using `setspn` utility as it's explained afterwards). Take into account this service ticket is configured for IIS and might be in conflict with the Apache Tomcat instance that could cause some functional issues like for example using NTLM instead of Kerberos. We strongly recommend creating an alias server name just for the Valve service as it will not have any conflict with Windows services.

Create a Server Name Alias

You can create a server alias for the Valve instance if the server is on Windows. In order to do so, you can create just an Active Directory user account for the Valve using Windows administration tools [Start → Programs → Administrative Tools → Active Directory Users and Computers]. Select the option to add a new user that holds Valve configuration either from the menu or clicking the right mouse button:



Use an alternative account name for the Apache Tomcat server. For example, you can name it as *gsasecurity*, as follows:

The screenshot shows the 'gsasecurity Properties' dialog box with the 'Account' tab active. The 'User logon name' is 'gsasecurity@enterprise'. The 'User logon name (pre-Windows 2000)' is 'ENTERPRISE\gsasecurity'. Under 'Account options', 'Password never expires' is checked. Under 'Account expires', 'Never' is selected.

In this example, the user domain is *enterprise* and the full user login name is *gsasecurity@enterprise*. In your case your domain should be something like *domain.corp.com* that fits with your corporate Windows domain.

When you create it, take into account the following considerations:

- Do not select "User must change password at next logon."
- Select "Password never expires"
- Make sure that you do not have the computer name *gsasecurity* under Computers and Domain Controllers. If so, then you need to choose a different user account name.

The DNS and network settings should be configured properly, so that there has to be an alias server name that points to the real server (f.e: *gsasecurityserver.google.com*). We'll see this alias server name will be mapped with this new user. This configuration may vary depending on how servers are deployed in your environment, as for example the configuration is not the same if you have just a simple server or a bunch of them load balanced.

Create the Service Name

The account you have already created is meant to be used as an Kerberos HTTP service for the Valve Security Framework. This is done in Windows environments using the *setspn* command line tool that manages SPNs (Service Principal Name) in the Active Directory [More information on *Setspn*:

<<http://technet2.microsoft.com/windowsserver/en/library/b3a029a1-7ff0-4f6f-87d2-f2e70294a5761033.mspx?mfr=true>>].

You would need to add (-a) an SPN for such an account, associating it with the fully qualified server alias name. For example:

```
setspn -a HTTP/gsasecurity.google.com gsasecurity
```

You could see it has been successfully created listing (-l) the SPNs available for such account:

```
setspn -l gsasecurity
```

Note: this command line utility might not be available in your OS and you should have to download it from Microsoft site.

Create the Keytab file

A *keytab* is a file that contains a Kerberos ticket. This is specially meant for services that can just read the Kerberos ticket from it like this is the case. It also configures an SPN for an alias server on Windows Active Directory. The *keytab* file is created by the Kerberos *ktpass* command and it has to make it available to the Valve application to use it.

The *ktpass* command options you have to execute may vary between Windows versions and even depends on the Service Pack in place. For that reason check it first at Microsoft documentation [\[http://www.google.com/search?hl=en&q=ktpass+site%3Amicrosoft.com\]](http://www.google.com/search?hl=en&q=ktpass+site%3Amicrosoft.com). For example if your Domain Controller is in Windows 2003 SP1 this command looks like:

[NOTE: ONLY for Windows 2003 SP1]

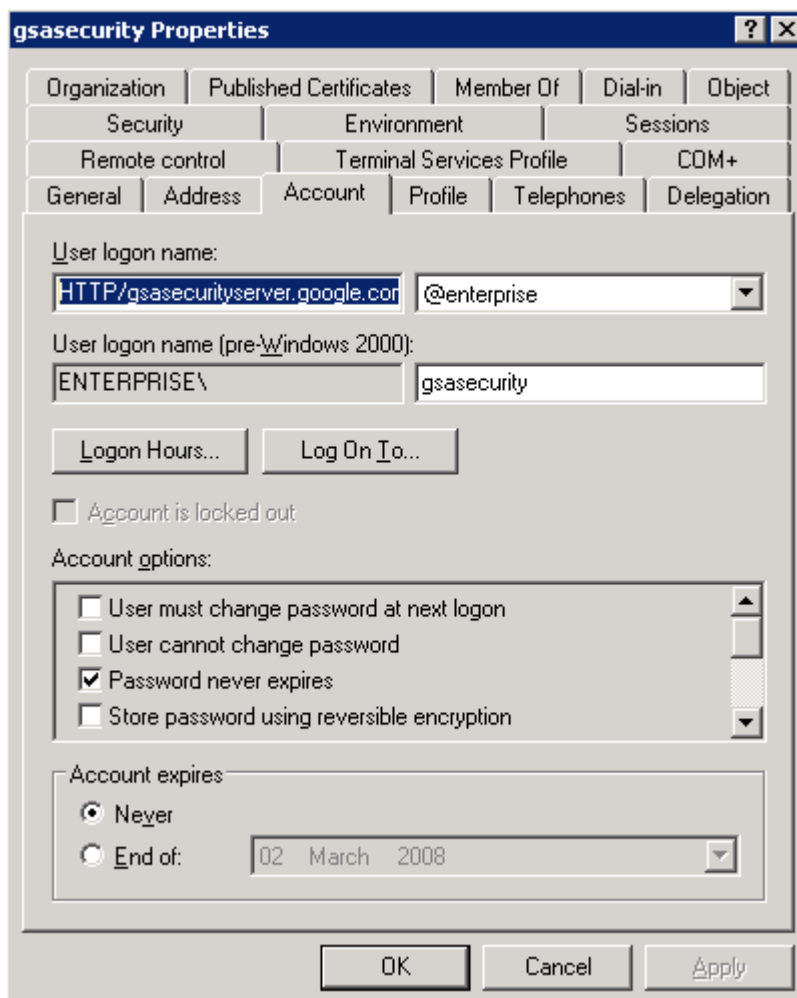
```
ktpass -princ HTTP/gsasecurityserver.google.com@ENTERPRISE
-mapuser gsasecurity -crypto rc4-hmac-nt -ptype KRB5_NT_SRV_HST
-mapop set -pass Password01
-out "c:\Documents and Settings\clazaro\krb5.keytab"
```

This step is extremely important to create the keytab file in the proper way as it'll be the one used by the Security Framework as the ticket storage. Otherwise you can find problems related to the access to such ticket or the encryption algorithms used. In previous versions to Windows 2003 SP1 you would have to use DES encryption instead of RC4.

This command should write out a successful confirmation message something like:

```
Key created.
Output keytab to c:\Documents and Settings\clazaro\krb5.keytab:
Keytab version: 0x502
keysize 99 HTTP/gsasecurityserver.google.com@ENTERPRISE ptype 3
(KRB5_NT_SRV_HST) vno 2 etype 0x17 (RC4-HMAC) keylength 16
(0xf3d2645787e66031e8d8803b83f9f0b2)
```

The user account entry at AD should have changed as well, including now the service account name:



If you are not using DES encryption then the option “Use DES encryption types for this account” should be unchecked in the service account. Otherwise it should be checked in.

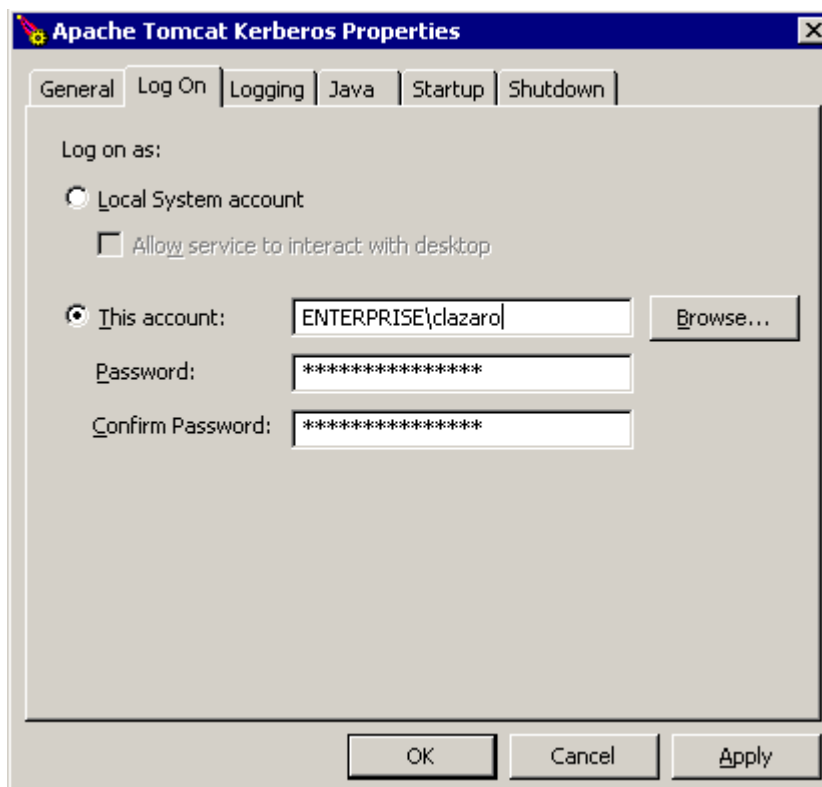
If you want to use the Kerberos cache, so it means the Valve configuration can make use of it directly, please store the ticket in the cache, including the keys, with the following command executed by the user that owns the Tomcat service:

```
kinit -k -t "c:\Documents and Settings\clazaro\krb5.keytab"
HTTP/gsasecurityserver.google.com
```

Pay attention to which *kinit* command you're using as it'll store the ticket in the default cache associated with this tool. Any JDK version has its own *kinit* tool, so you'd be storing the ticket into the cache of other JDK or Kerberos implementation and couldn't be reached by the JDK runtime the Valve is using.

In Windows the cache file should be located at `c:\Documents and Settings\`, where `username` is the one that runs the Tomcat service on (for example “clazaro”). There should be there a file name `krb5cc_<username>`. You should put there as well the keytab file created named as `krb5.keytab` as this is the default location for Windows.

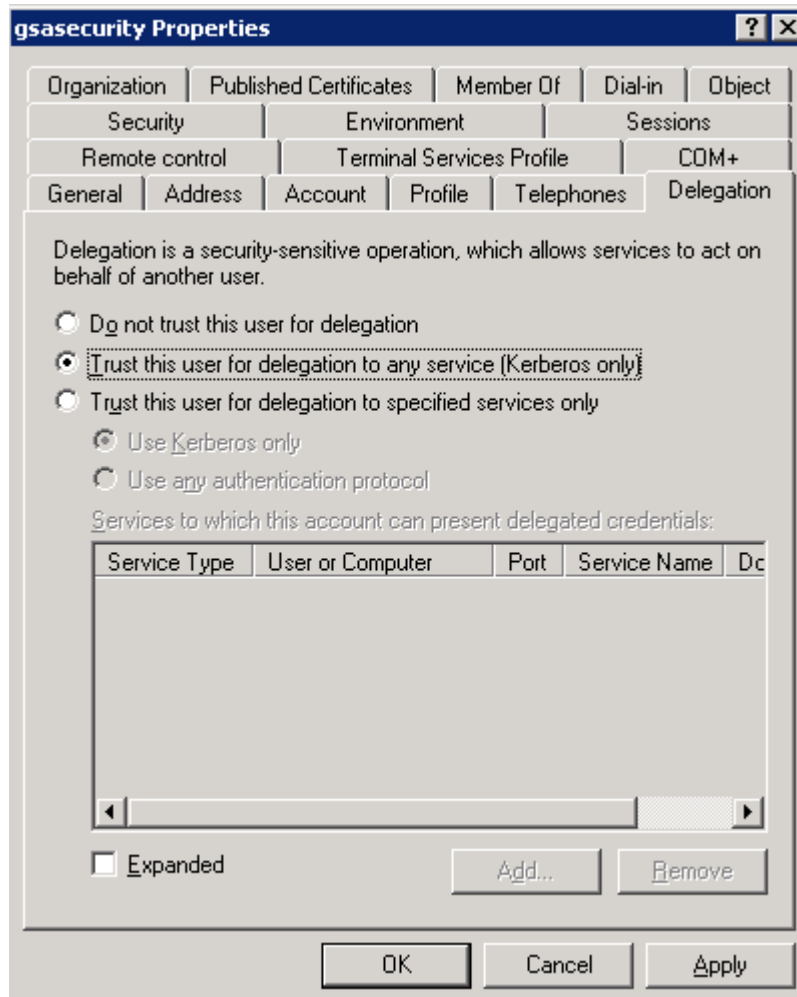
Make sure the proper user is running the Tomcat service on. In Windows you can specify this user in the Apache Tomcat service properties the following way:



Configure Delegation

The GSA Valve Security Framework impersonates the user and delegates his/her credentials to backend applications. In the case of Kerberos, this is doable in Windows thanks to a delegation feature. You should set this feature up for the Valve service account before proceeding as otherwise the user's Kerberos ticket will not be sent over from the Valve to any third party application. In Windows 2000 you just can enable or disable this feature (you can do it in the Account property "Account is Trusted for Delegation"), but since version 2003 this delegation process is much more restrictive as you're able to discriminate by protocol, port or host based on the Constraint Delegation feature.

In the *gsasecurity* account property window, go to Delegation tab and set user delegation up there, for example just trusting this user only for Kerberos delegation to any service as follows:

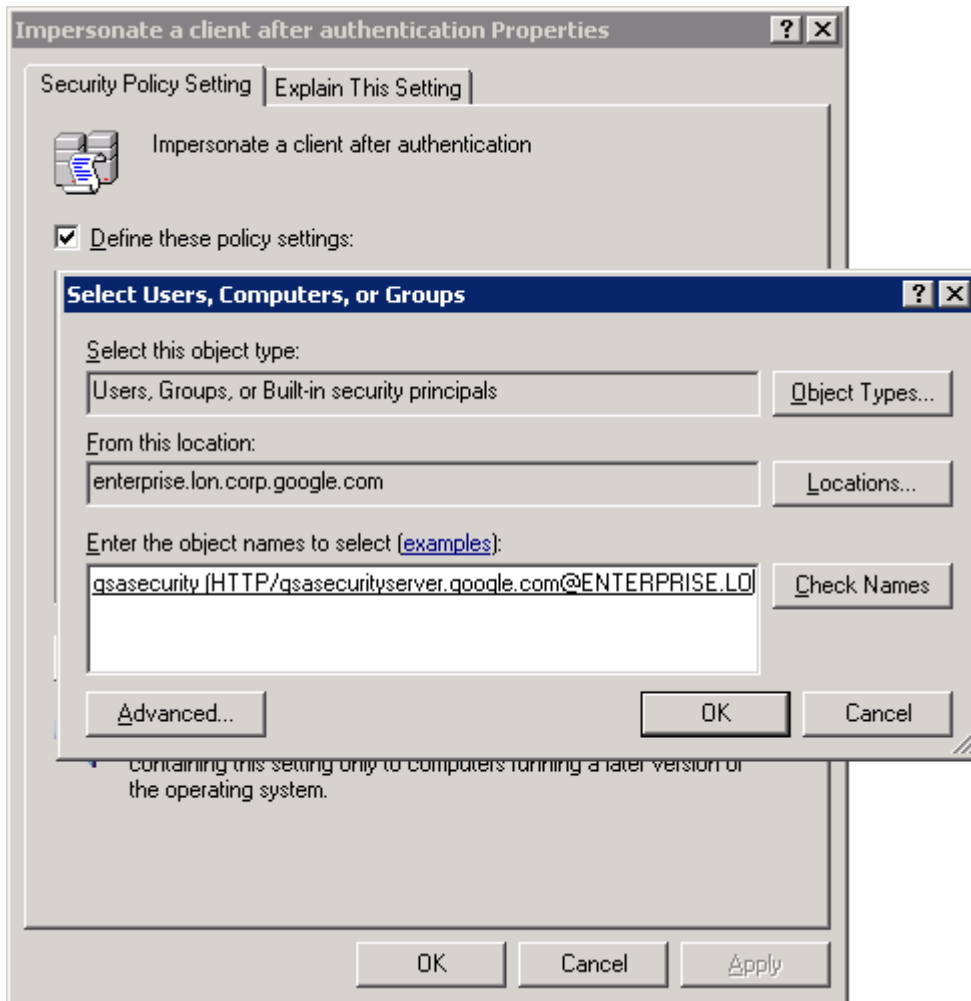


If you are configuring it on a production environment using Windows 2003, you should use constraint delegation (“Trust this user for delegation to specified services only” option) just selecting those computers (applications) you are going to impersonate users against it.

Configure Impersonation

The user should be able to take the user ticket and forward it to impersonate users. This has to be configured in Windows configuring the security settings on the same domain where the Valve is installed. Open Administrative tools, click then on Local Security Settings (or *Domain Security Policy* if you are at the Domain Controller). In the left panel, navigate the tree to security settings/ Local Policies/User rights assignment. A couple of things have to be configured there:

- *Impersonate a client after authentication*: on the right panel, look for “Impersonate a client after authentication. You should add the service account previously created for the Valve, as follows:



- *Logon as a service*: include as well the service account to be logged on as the application server service. This user is the owner of the service, in the case of the example shown here this user will be “clazaro”.

Create a Java Configuration File

The way the Valve has been defined is so flexible that makes use of a Java file for configuring the Kerberos implementation. This file is a standard one and will make use of the files previously generated.

Based on the same example, this file should look like:

```
com.sun.security.jgss.initiate {
    com.sun.security.auth.module.Krb5LoginModule required client=TRUE
    useTicketCache="true" debug=true;
};

com.sun.security.jgss.accept {
    com.sun.security.auth.module.Krb5LoginModule required
    principal="HTTP/gsasecurityserver.google.com@ENTERPRISE"
    keytab="c:\Documents and Settings\clazaro\krb5.keytab"
    useKeyTab=true
    storeKey=true
    doNotPrompt=true
    debug=true;
};
```

Change this file pointing to the exact locations of your files. Leave the flag *debug* to true until you check everything is working smoothly. The you should set it to false as it's very verbose.

Configure the Valve for Kerberos

Once the Kerberos environment has been properly set, the Valve has to been configured as well for Kerberos. Have a look at the Valve's Installation Guide before if it's not been installed yet. In this document we only talk about the Kerberos configuration, so if you are also interested on setting the whole Valve up, have a look at the same document. In the case you want to review advanced Kerberos deployment you can find some at the Deployment Scenario Guide.

The Valve configuration file contains all the parameters that drive how this security framework behaves. The default location of this file is at `$TOMCAT_HOME/common/classes/gsaValveConfig.xml`. Those configuration attributes related to Kerberos are the following:

Attributes	Description
isKerberos	Kerberos setting attribute. It just enable or disable any Kerberos treatment in the security framework [true false]
isNegotiate	If Kerberos is set, this parameter says if the user ticket authentication process is going to be silent (thru SPNEGO) or creating it on the fly with username and password [true false]
krbini	It points to the Kerberos network configuration file: krb5.ini (Win) or krb5.conf (Linux/Unix). This is the file mentioned in this document at Kerberos Configuration File chapter.
krbconfig	Java config file for Kerberos. It has to point to the Java configuration file created above.
krbAdditionalAuthN	If isNegotiate is set to true, this parameter says if there is an additional authentication process using a form apart from the silent authentication. It means we get a couple of credentials, one Kerberos and another one based on username and password to authenticate users in multiple repositories. [true false]
krbLoginUrl	Path to a login form when the previous parameter is "true". The default one in the Valve is loginkrb.jsp. The only difference between this JSP file and login.jsp is the form's action URL as they are completely different. Anyway, you can use the login page you want as long they keep the right action URL.
krbUsrPwdCrawler	The Kerberos crawling process is going to be done thru a login form that creates the crawler Kerberos tickets using his username and password [true false] (Only for Forms Based interface)
krbUsrPwdCrawlerUrl	It points to the crawling form set in the previous parameter. You can find in the Java application a default crawling login page for Kerberos: logincrawlerkrb.jsp (Only for Forms Based interface)

This attributes are included in the Valve's *kerberos* parameter. The example below shows you how to configure the Kerberos component. You should adapt it to your environment.

```
<kerberos isKerberos="true"
  isNegotiate="false"
  krbini="c:\\kerberos\\krb5.ini"
  krbconfig="c:\\kerberos\\bcsLogin.conf"
  krbAdditionalAuthN="false"
  krbLoginUrl=
    "http://gsasecurityserver.google.com/valve/loginkrb.jsp"
  krbUsrPwdCrawler="true"
  krbUsrPwdCrawlerUrl=
    "http://gsasecurityserver.google.com/valve/logincrawlerkrb.jsp"/>
```

In the above example, as `isNegotiate` parameter is set to `false`, both the `krbAdditionalAuthN` and `krbLoginUrl` are obviated. The last two ones about the Kerberos crawling process are independent from the others.

Other setting needed to be configured in here is the session. Session could be defined for example the following way:

```
<sessions isSessionEnabled="true"
  sessionTimeout="-1"
  maxSessionAge="400"
  sessionCleanup="30"
  sendCookies="true"/>
```

Adapt these session parameters to your scenario bearing in mind the following rules:

- The session could be enabled or not. If you enable it you will set it up for the whole Security Framework. If you don't want to have such scenario you can set the session as disabled (`isSessionEnabled="false"`) but setting the `maxSessionAge` parameter to the proper value (take into account Kerberos tickets are max aged and by default is set to 10 hours)
- If you set `maxSessionAge` attribute, then the `authMaxAge` parameter has to match with that value. One important difference is `maxSessionAge` is established in minutes whereas `authMaxAge` is done in seconds, so you should do the translation, for example:

```
...
<authMaxAge>24000</authMaxAge>
...
<sessions ...
  maxSessionAge="400"
/>
```

- Have a look at the Configuration guide and the Scenario guide as well to see how the other parameters should look like based on your requirements.

Test the Kerberized environment

You can now test if the new Kerberos settings are working as expected.

If you are just using the Valve's **Forms based interface**, open a new browser and access to any of your kerberized URLs (i.e. document secure using Kerberos) protected by the Security Framework the following way:

```
http://<valve_host>:<port>/valve/kerberos?returnPath=<kerberized_doc>
```

For example:

```
http://gsasecurityserver.google.com/valve/kerberos?returnPath=http://server.google.com
```

Take into account that you should access to this URL from a different machine than either the domain controller or the server where the security framework is running. Otherwise Kerberos tickets could not be exchanged.

In the case you were using the **SAML interface**, the best way to test this integration is working thru the GSA integration directly. Remember to set the debug level to a very verbose mode to clearly identify how the integration is being done.

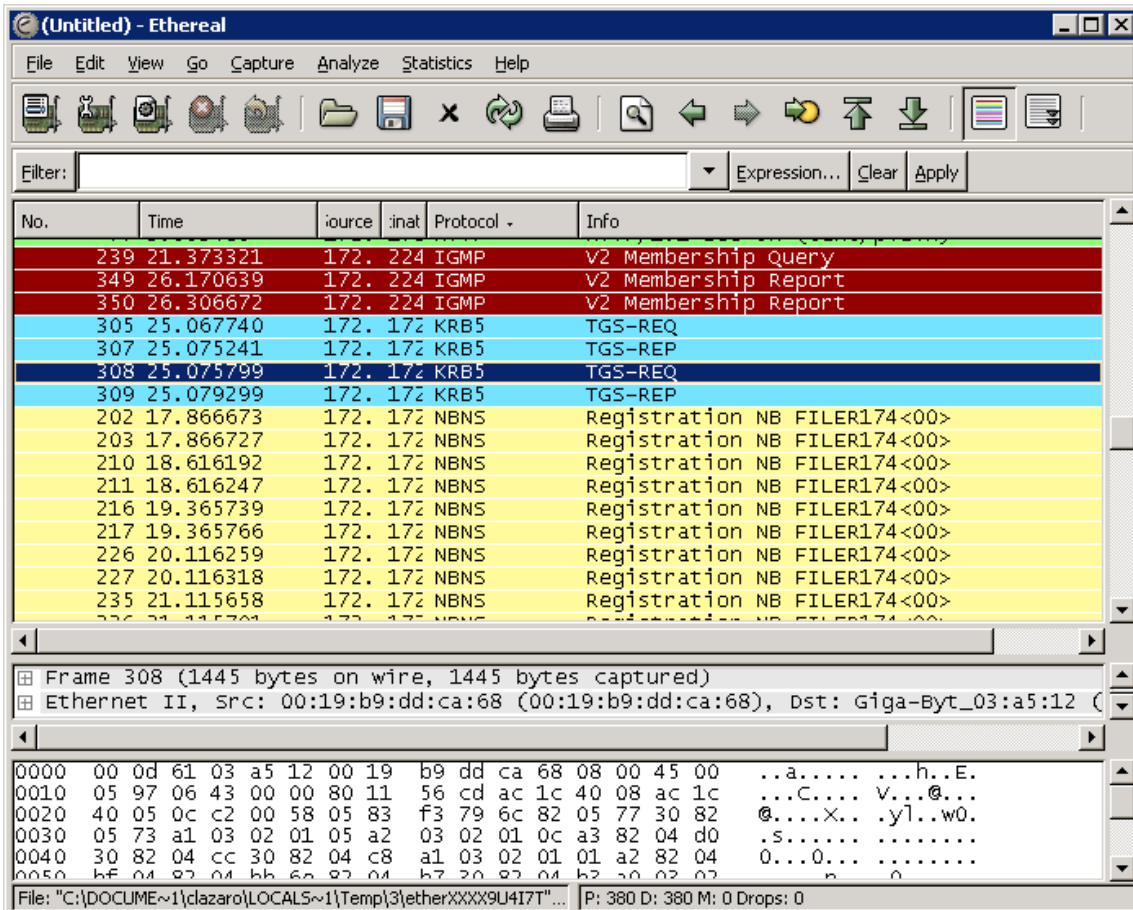
Troubleshooting

Kerberos is a complex protocol that sometimes is difficult to understand in case you'd have any issue. It would be almost impossible to include in this guide all the problems you'd find (it's more than possible that you'd face some Kerberos configuration issues) when setting up this environment and their solutions but you will get here some tips on how you can trace the environment in order to solve them. This guide has shown you how to configure the Valve Security Framework using Kerberos but as this protocol configuration might vary between OS versions and even Kerberos implementations, you should adapt it to the main configuration steps explained on this document.

Whenever a problem arises you should trace what it's happening in your environment and that's why it's important for you knowing how Kerberos and the lightweight Spnego protocol work. You can find on Internet multiple documents that explain them in detail, for example the following ones:

- MIT: Kerberos [<http://web.mit.edu/Kerberos/>]
- IETF: RFC 4559 – SPNEGO-based Kerberos Authentication in Microsoft Windows [<http://tools.ietf.org/html/rfc4559>]
- Microsoft HTTP-Based Cross-Platform Authentication via the Negotiate Protocol [<http://msdn2.microsoft.com/en-us/library/ms995329.aspx>]

There are multiple Kerberos errors that can happen in different stages as there are several interactions between the KDC, the server and the client. Sometimes it's useful to trace those network messages exchanged by these components to understand what is happening in this scenario. You can use a network protocol analyzer like Ethereal <<http://www.ethereal.com/>> where you can trace the KRB5 protocol as well.



If your network is based on Windows, the Microsoft's Kerberos Troubleshooting guide is a good document that could help you as you can find there what any error code means and how to solve it. You can download its Windows 2003 version at the following location <http://download.microsoft.com/download/5/9/c/59c349f5-f0c8-4b9e-9f70-dbc5f2a8c330/Troubleshooting_Kerberos_Errors.DOC>