

## White Paper

# AppGate and Single-Sign-On functionality

*Tomas Olovsson  
Gothenburg – May 19, 2004*

**There exist no universal single sign-on standard, but single sign-on, SSO, can still be introduced using different techniques. Depending on the underlying technology (client/server, legacy or web based applications), different approaches can be taken to create a SSO environment. This paper shows how an AppGate system can be used to implement single sign-on in a user environment.**

### The AppGate solution

An AppGate system is often used to protect internal resources from unauthorized access and as a remote-access solution for traveling users, partners and home users. It allows authorized users to securely access their information resources from almost any type of system. The AppGate system provides fast, easy access to multiple applications through a single log-on procedure while protecting user traffic through encryption and tunneling. It's built-in firewall functionality not only makes it possible to place it in very hostile environments but also makes it ideal to protect application servers from all types of unauthorized traffic without the need for an additional firewall.

### Single Sign-on

There are several reasons why single sign-on solutions are attractive to system owners, administrators and users:

- Users do not have to remember passwords to various systems but can after signing on to an AppGate server, access various applications without further authentication.
- Users can use strong authentication methods such as smart cards or a token devices regardless of whether their applications support it or not.
- When a user is authenticated by an AppGate server, it is not possible to switch identity, for example to mount a file share from a file server as another user if he/she happens to know another user's password.
- User passwords are less exposed which minimizes the risk that they are copied or stolen. If strong authentication mechanisms are used to the AppGate server, the use of passwords may be completely eliminated.
- System administrators have one central place where user identities and user rights are managed.
- Central log files can be used to monitor user activities instead of inspecting each application's private log, if there is one.

Although SSO potentially gives access to a large number of services, it is more convenient to use strong authentication since it's only required once. This minimizes the risk of unau-

thorized access due to a lost password. A token device or a smart card can immediately be revoked if it is missing.

## **AppGate and Single Sign-on**

The AppGate system can enable SSO functionality to various applications. The lack of a universal standard for single sign-on means that the method used to implement SSO must vary depending on the application. Since the AppGate server always authenticates its users and allows only authorized users to connect to application servers, the issue is how to transfer user credentials to the applications.

## **Unix/Linux and X-Windows**

Applications running on Unix systems are often X-Windows based and may therefore automatically be started by the AppGate server. When a user clicks on an icon representing a service in the AppGate client's portal-like interface, the AppGate server can automatically start the application for the user on the application server with the user's identity passed on. The user will just see a new window appearing on the workstation as the result on clicking an icon for the application.

## **Web-based applications**

There are two ways to implement SSO for web-based applications. First, the AppGate server always includes the user's true identity in all HTTP requests, thus all web servers and other application servers can always determine a user's identity from an object request. This solution works well if the application can be modified to use this information.

Another solution is to use the *web agents* included in version 6.2 of the server. The web agent is a small program that gets started on the AppGate server when a user accesses a certain web page. Thus, the web agent can invisibly handle the login phase to any web application without user interaction and then seamlessly hand over the established session to the client application or web browser.

As always, when using the AppGate system, the systems administrators can control under what circumstances the users should have access to the applications (for example based on time, IP addresses, authentication method, etc.)

## **Windows file sharing**

Windows file sharing, i.e. mapping of network drives, is handled through a NetBIOS proxy on the AppGate server. The AppGate server always replaces the users identity in the map request with the users true name to prevent users from accessing other users files. The NetBIOS proxy handles the users names but does not manage any passwords, but since the user's identity is known and can be trusted by the file server, a user with the same name and password at the local system as on the file server will not be asked for a password. It may also be possible to configure the file server to not ask for user passwords since the file server can trust his identity. Thus, a true SSO environment can be implemented, if desired.

In highly secure environments, the proxy can also be asked to log more information about user sessions, such as what files were uploaded and downloaded, possibly including all data being transferred to and from the user's workstation.

## **API and traffic interception for legacy applications**

SSO functionality can be enabled for other protocols, such as to legacy applications, using two different approaches: First, traffic can always be intercepted in the AppGate server and a small customer-designed program can intercept the login dialog and invisibly log in the user to the server. The AppGate server allows administrators to add such modules on the server that intercepts any traffic between the client and the server.

It is also necessary to submit information about the users passwords to the AppGate server to make it possible for the AppGate server to log in on behalf of the user. For most protocols, the effort needed to implement SSO is relatively small.

The second method is possible to use if it is possible to make small modifications to the application. The AppGate server can be configured to always answer questions to applications and application servers regarding who is connecting to it. Thus, applications that can be configured to ask such questions can get an authoritative answer about the users true identity (for example proved by using a smart card) instead of asking the user for a username/password.

## **Applications not requiring unique user identities**

Finally, all applications do not have to know the user's real identity. Many applications do not differentiate users once they are authenticated and never log "who is doing what". All these applications can rely on the AppGate server to do that authorization, possibly with a much better authentication system than the application would support.

Since the AppGate administrator can control what users can access a particular application and under what circumstances, it is possible to relax the checks in the application to allow all users being authorized by the AppGate server to run the application. In addition, all requests are logged by the AppGate server, which always makes it possible to go back to the logs and see who used the application and at what time.

At first sight, this might not seem to be a true SSO solution, but it will ease administration and increase security and there are many applications that could benefit from this solution. If the application only asks for a username/password for authorization, it is much easier to hand that task to the AppGate server. In addition, there will be a central repository where all user identities and user rights are managed instead of having to configure each application.

## **Summary**

Since there exist no established standard for single sign-on today, it is necessary to approach each application or group of applications with different solutions. For many people, this is a much better solution than to wait for a standard to emerge. And even if a SSO standard eventually appears, legacy applications will still have to be treated this way.

Single sign-on means not just that users don't have to remember many passwords but also that a common authentication system such as smart cards or token cards can be used together with many more applications than today. Central administration of user rights and log files makes it much easier to administer such systems. In many cases, application log files are never inspected unless they are sent to centrally managed log files.

An AppGate server also protects application servers from hostile traffic and ensures that all traffic between the client and the server is encrypted. Some applications include their own mechanism for encrypting network traffic, but with an AppGate solution all applications can benefit from encryption with approved ciphers (such as AES with long keys) and certified protocols.

A central authentication system is also more cost effective than having to administer many different authentication mechanisms and user accounts on application servers. Security is also increased since users cannot share identities to servers and if passwords are not used, many attacks against servers will not be possible such as password guessing against file servers using large dictionaries.