Study of HTTP authentications

Zhihao Cao

In this paper, we are going to explore how several kinds of web application authentications are accomplished between client and server over the Internet. When we use computer to surf the Internet, we always need to log on some websites to prove identity and get access to private or sensitive information. This is the most frequent authentication activity we do in our Internet applications. Therefore secure and reliable solutions are needed to prevent our credential from being exposed.

Normally, when we try to authenticate our identity over a web application, we input our user name and password into a login form on a web page. Then, the form will be sent by browser to an authentication server using HTTP protocol. There are several types of authentication schemes are offered by HTTP. The popular types of schemes are Anonymous, Basic, Digest, NTLM, Negotiate, and some others like proxy authentication, Windows Live ID and so on. Here are the details of schemes:

Anonymous:

It is hardly to say that Anonymous is one type of authentication scheme because it does not do any authentication. It just sends a request to server and wait response. In this way, we actually try to access some resources that have been granted privileges to everyone in the network.

Basic:

Basic is the simplest way for performing access controls to resources in the Internet. It is not a secure scheme because it does not need cookies or session identifiers. It basically sends a HTTP request to authentication server with the authorization header that contains a Base64-encoded string (client's user name and password). Since the Base 64bit encryption is reversible, this string is actually equivalent to plaintext of user's account and password. Even though later people added an md5 (so called irreversible hash) encryption to make the string irreversible, this scheme still could be easily broken by replay attack. If a resource owner requires the resource to be protected, a more secure scheme is needed or uses the Basic authentication over HTTPS.

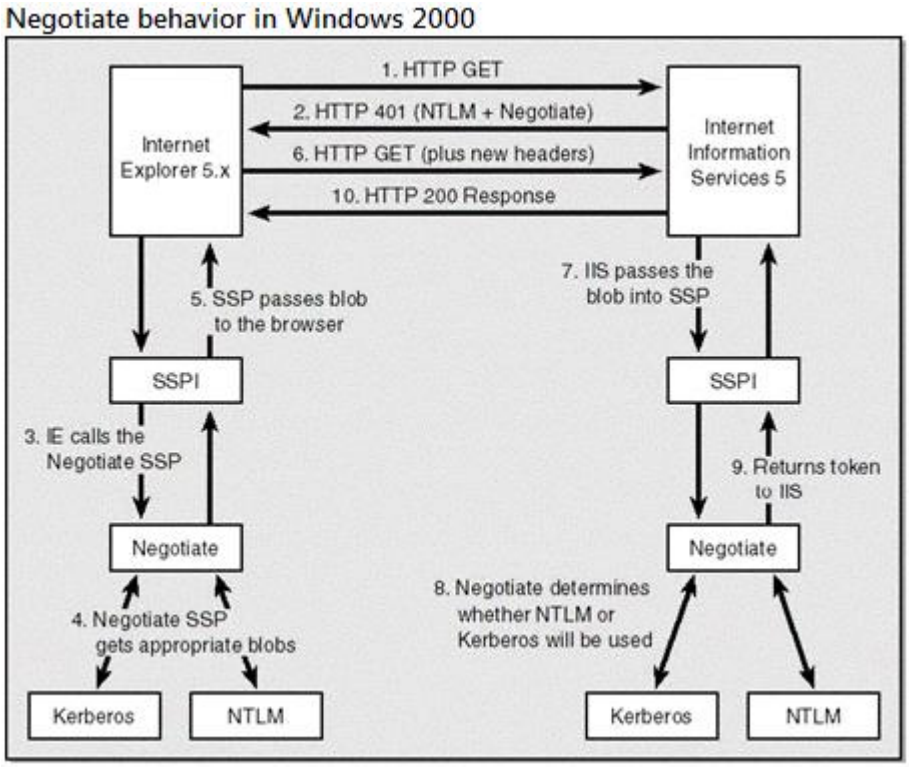
Digest:

Comparing to Basic authentication, Digest is more secure. It uses challenge-response way to accomplish the authentication. First, the server receives a HTTP request from client with Digest authentication. Then, the server sends a string which is called nonce back to client as a challenge. Then, the client encrypts the nonce with its credential in

MD5 algorithm, and sends the result within HTTP authorization header back to authentication server. The server verifies the credential by decrypting the result. In this way, the replay attack will not work because each time the encrypted value is different; server will not authorize the encrypted value the same as last time. Although it is more than Basic authentication, it still could be easily attacked by Man in the middle attack.

NTLM:

NTLM is also a challenge-response authentication scheme that mainly deployed by windows in the old time. Now this scheme is not recommended by Microsoft but still integrated inside Microsoft OS for compatibility with older system. This scheme needs multiple communications between client, server, and domain controller; and it also requires persistent connections to accomplish the authentication. First, client computer hash the password provided by user, and then sends the plaintext username to DC. The DC generates a nonce based on the username and sends back to client as a challenge. Then, client encrypts the challenge and the password's hash together, and then sends the user name, challenge and encrypted value to server as a response. The server receives the response from client and sends this response to domain controller. The domain controller retrieves the hash of user's password by looking up the database with the corresponding user name, and uses this hash of user's password to encrypt the challenge from server. In the last, the domain controller compares the encrypted challenge with the response that sent from client. The authentication is accomplished if these two values are the same. This process is complicated but not worthy compares to its costs. This scheme is vulnerable to some attacks like pass the hash attack and Squirtle toolkit. Since Windows 2000, Microsoft had already adopted Kerberos as its default authentication protocol, but still used NTLM as an alternative method for Negotiate scheme.



This figure was taken from ***Designing Secure Web-based Applications with Windows 2000*** by Michael Howard and published by Microsoft Press.

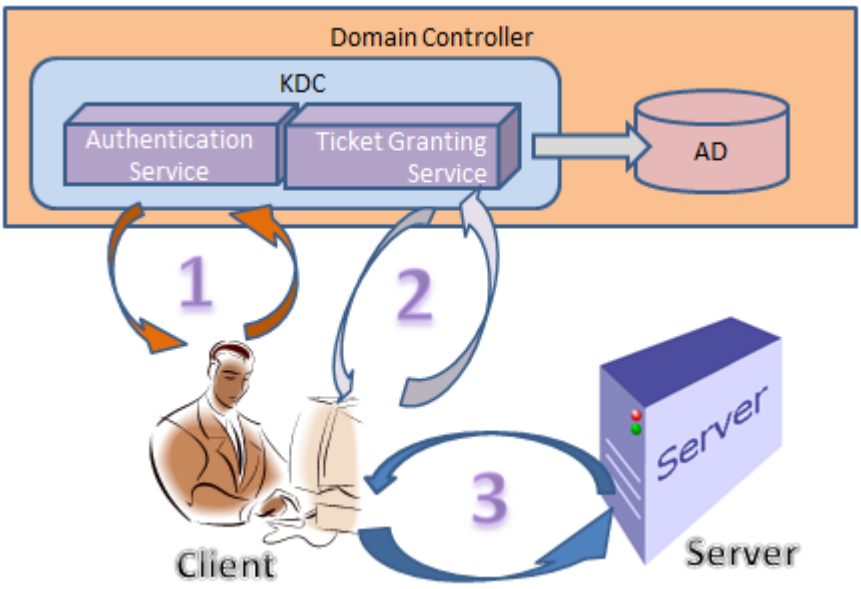
Negotiate:

Negotiate is a currently adopting main authentication scheme for Microsoft products. It is widely referred as Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO). It is one of the authentication protocols for Integrated Windows Authentication (IWA).

At first, the browser sends a HTTP request to authentication server. The authentication server will regard it as unauthenticated request for protected resources, and then responses a HTTP status 401 with headers that indicate the schemes that the authentication server would accept. Normally, the default options for Microsoft IIS are Negotiate and NTLM. Then, the browser will send a second request to negotiate what scheme should be chosen. The authentication server will respond a second HTTP status 200 with the information about authentication to finalize the decision that could be agreed by both sides. After the negotiation process, the browser starts requesting authentication through one of the two protocols: Kerberos and NTLM. If Kerberos is available both in browser and authentication server, it will be adopted by default; if not, the browser will try NTLM. The details of NTLM are explained in above subtopic; the details of Kerberos are going to be explained in below subtopic.

Kerberos:

Kerberos is one of the most widely deployed authentication protocols in the Network security field. It is a highly improved version of NTLM developed by MIT. It is designed to be more efficient, more secure and more reliable based on NTLM. In addition, it is a mutual authentication protocol and it supports Single sign-on (SSO) property. It became a default authentication method for Windows products since Windows 2000. It is widely adopted by UNIX and UNIX-like OS such as Apple's Mac OS X, Linux, Oracle's Solaris, and so on.



The principle:

Kerberos actually works on the basis of "tickets" to enforce authentication control. If a client requires the access to a protected resource, it needs to get an authenticated ticket Key Distribution Center (KDC) before doing that. In other words, client needs to buy a ticket and the ticker should be authenticated before accessing the target server. However, the client needs a Ticket Granting Ticket (TGT) to buy a ticket. That is, client should obtain the Ticket Granting Ticket before buying a ticket.

Getting Ticket Granting Ticket:

The Ticket Granting Ticket is a main mission of Kerberos Authentication service (KAS). When a user input the user name and password into a client machine and tries to get authentication from a server, the Kerberos protocol will first send a request to Key Distribution Center (KDC) for requesting a Ticket Granting Ticket (TGT). The request contains two parts: user name's plaintext and encrypted authenticator. After Key Distribution Center (KDC) receives the request, it retrieves the user information from Active Directory (DC), and uses this information to generate a key to decrypt the authenticator to verify client's authentication. If the decrypted authenticator value is identical to known value from AD, the client's identity is proved. KAS will generate a

Logon Session key (LSK) encrypted by the derivative key of user's password and generates Ticket Granting Ticket (TGT). The TGT mainly includes two things: user's information and Logon Session key (LSK). The TGT is also be encrypted by KDC's key. At last, two values (LSK and TGT) that encrypted by different keys are sent back to client.

Buying a ticket:

The client receives the Logon Session key (LSK) and Ticket Granting Ticket (TGT). Now, if client wants to access a server's resources, it needs to use the TGT to buy corresponding Service Ticket (ST) from Key Distribution Center (KDC). So now, the client sends a request to Ticket Granting Service (TGS) for requesting a Service Ticket (ST). The request contains four attributes: User name, Authenticator (encrypted by LSK), TGT and the requested server name. TGS receives the request, decrypts the TGT by KDC's key to retrieve the LSK, and uses this LSK to decrypt the authenticator to authenticate client's identity. If the identity is proved, TGS will generate a Service Session key (SSK) that encrypted by LSK to keep communication secure between client and server, and generates a Service ticket (ST) encrypted by the derivative key of server's password for the requested server. At last, TGS sends the Service Session key (SSK) and Service ticket (ST) back to client.

Accessing using ticket:

Client receives the response from Ticket Granting Service (TGS). It decrypts SSK by using its Logon Session key (LSK). It caches SSK and ST. Then client can use this ST and an Authenticator to access requested server. However, how dose target serer knows the ST is generated by TGS rather than fake by someone else? Actually, after target server receives client's request, it will use the derivative key of its own password to decrypts ST to retrieve the SSK. Then it uses this SSK to decrypt the Authenticator to retrieve user identity. So far, the requested server has finished the authentication for client, but we should remember that Kerberos is a Mutual Authentication protocol. Therefore, the client now needs to authenticate if the requested server is a phishing server. To do that, server need to re-encrypt the Authenticator using Service Session Key (SSK) and sends it back to client. Then, the client decrypts this Authenticator using the SSK in its cache to verify if the decrypted Authenticator is identical to it was. This can prove that the client and the requested server have the same SSK. Till this step, the client, KDC and requested have successfully finish the Mutual Authentication.

Conclusion

In this study, 6 kinds of HTTP authentication schemes are explored. The sequence of schemes we go through is from simple to complex, so we understand how these schemes are different from basic to sophisticated. In reality, the HTTP authentication

will automatically choose the most secure scheme to keep the Network away from being attacked, but the actual selection is up to what schemes are available on a server.

Reference

- HTTP-Based Cross-Platform Authentication by Using the Negotiate Protocol
  https://msdn.microsoft.com/en-us/library/ms995329.aspx

- How Kerberos authentication works
  http://www.aub.edu.lb/it/kb/win/Pages/Kerberos_With_Active_Directory.aspx

- How HTTP Authentication works and why load testers should care
  http://www.webperformance.com/load-testing/blog/2011/06/how-http-authentication-works-and-why-load-testers-should-care/

- Introduction to Kerberos Authentication
  https://software.intel.com/sites/manageability/AMT_Implementation_and_Reference_Guide/default.htm?turl=WordDocuments%2Fintroductiontokerberosauthentication.htm

- Windows authentication – how it work
  http://www.cnblogs.com/artech/archive/2011/01/24/kerberos.html

- Understanding HTTP Authentication
  https://msdn.microsoft.com/en-us/library/ms789031.aspx

- NTLM authentication process
  http://www.cnblogs.com/xwdreamer/archive/2012/08/23/2652541.html

- James F. Kurose and Keith W. Ross. Computer Networking 4th Edition a Top-Down Approach, 2008