Inside Blizzard: Battle.net

Part 1

Skywing
skywing_uninformed@valhallalegends.com

Contents

Chapter 1

Foreword

Abstract: This paper intends to describe a variety of the problems Blizzard Entertainment has encountered from a practical standpoint through their implementation of the large-scale online game matchmaking and chat service, Battle.net. The paper provides some background historical information into the design and purpose of Battle.net and continues on to discuss a variety of flaws that have been observed in the implementation of the system. Readers should come away with a better understanding of problems that can be easily introduced in designing a matchmaking/chat system to operate on such a large scale in addition to some of the serious security-related consequences of not performing proper parameter validation of untrusted clients.

Chapter 2

Introduction

First, a bit of historical and background information, leading up to the present day. Battle.net is an online matchmaking service that allows players to set up online games with other players. It is quite possibly the oldest and largest system of it's kind currently in existence (launched in 1997).

The basic services provided by Battle.net are game matchmaking and chat. The matchmaking system allows one to create and join games with little or no prior configuration required (other than picking game parameters, such as a map to play on, or so-forth). The chat system is similar to a stripped-down version of Internet Relay Chat. The primary differences between IRC and Battle.net (for the purposes of the chat system) are that Battle.net only allows a user to be present in one chat channel at once, and many of the channel parameters that IRC users might be familiar with (maximum number of users in the channel, who has channel operator privileges) are fixed to well-defined values by the server.

Battle.net supports a wide variety of Blizzard games, including Diablo, Starcraft, Warcraft II: Battle.net Edition, Diablo II, and Warcraft III. In addition, there are shareware versions of Diablo and Starcraft that are supported on Battle.net, as well as optional expansions for Diablo II, Starcraft, and Warcraft III. All of these games share a common binary communication protocol that has evolved over the past 8 years, although different games have differing capabilities with respect to the protocol.

In some cases, this is due to differing requirements for the game clients, but usually this is simply due to the older programs not being updated as frequently as newer versions. In short, there are a number of different dialects of the Battle.net binary protocol that are used by the various supported products, all at the same time. In addition to supporting an undocumented binary protocol, Battle.net has for some time now supported a text-based protocol (the “Chat

Gateway", as officialy documented). This protocol supports a limited subset of the features available to clients using the full game protocol. In particular, it lacks support for capabilities such as account creation and management.

Both of these protocols are now fairly well understood and documented certain persons outside of Blizzard. Although the text-based protocol is documented and fairly stable, the limitations inherent in it make it undesirable for many uses. Furthermore, in order to help stem the flood of spam on Battle.net, Blizzard changed their server software to prevent clients using the text-based protocol from entering all but a few pre-defined chat channels. As a result of this, many developers have reverse engineered (or more commonly, used the work of those who came before them) the Battle.net binary protocol and written their own "emulator" clients for various purposes (typically as a better alternative to the limited chat facilities provided by Blizzard's game clients). These clients emulate the behavior of a particular Blizzard game program in order to trick Battle.net into providing the services typically only offered to the game clients, hence the name "emulator client" [1]. In fact, there are also partially compliant server implementations that implement the server-side chat and matchmaking logic supported by Battle.net to varying degrees of accuracy. One can today download a third party server that emulates the Battle.net protocol, and a third party client that emulates a Blizzard client supporting the Battle.net protocol, and have the two inter-operate.

[1] Most of these clients area referred to as "emulator bots" or "emubots" by their developers, and the Battle.net community in general

Chapter 3

Battle.net issues

By virtue of supporting so many different game clients (at present, there are 11 distinct Blizzard-supported programs that connect to Battle.net), Blizzard has a sizable version-control problem. In fact, this problem is compounded by several issues.

First, many client game patches add or change the protocol in significant ways. For instance, the notion of password-protected, persistent player accounts was not originally even designed into Battle.net, and was added at a later date via a client patch (and server-side modifications).

On top of that, many clients also have very significant differences in feature support. To give an example, for many years Diablo and Diablo Shareware were both supported on Battle.net concurrently while Diablo supported user accounts and the shareware version did not. As one can imagine, this sort of thing can give rise to a great many problems. The version control and update mechanism is not separate from the rest of the protocol. Indeed, the same server, and the same connection, are used for version control, but a different connection to the same server is used for the transfer of client patches. As a result, any compliant Battle.net server is required to support not only the current Battle.net protocol version that is in use by the current patch level of every existing client, but it must also support the first few messages used by every single version of every single Battle.net client ever released, or at least until the version checking mechanism can be invoked to distribute a new version (which is not the first task that occurs in some older iterations of the protocol).

To make matters worse, there is now a proliferation of third party clients using the Battle.net protocol (to varying degrees of accuracy compared to the Blizzard game clients they attempt to emulate) in use on Battle.net today. This began sometime in mid-1999 when a program called "NBBot", authored by Andreas Hansson, who often goes by the handle "Adron", entered widespread

distribution, though this was not the intent of the author. NBBot was the first third party client to emulate the Battle.net protocol to an extent that allowed it to masquerade as a game client. Several years later, the source code for this program was inadvertently released to wide-spread public distribution, which kicked off large-scale development of third party Battle.net protocol clients by a number of authors.

Despite all of these challenges, Blizzard has managed to keep Battle.net up and running for nearly a decade now, and claims over a million active users. However, the road leading up to the present day has not been "clear sailing" for Blizzard. This leads us into some of the specific problems facing Battle.net leading up until the present day. One of the major classes of problems encountered by Blizzard as Battle.net has grown is that it was (in the author's opinion) simply not designed to support the circumstances in which it eventually ended up being used. This is evident in a variety of events that have occurred over the past few years:

- The addition of persistent player accounts to the system.
- The addition of the text-based chat protocol to the system.
- Significant changes to the backend architecture utilized by Battle.net.

Although it is difficult to provide exact details of these changes, having not worked at Blizzard, many of them can be inferred.

3.1 Network issues

Battle.net was originally setup as a small number of linked servers placed at various strategic geographical locations. They were "linked" in the sense that players on one server could interact with players on a different server as seamlessly as with players connected to the same server. This architecture eventually proved unsupportable, as increasing usage of Battle.net led to the common occurrence of "server splits", in which one or more servers would be unable to keep up with the rest of the network and become temporarily disconnected.

Eventually, the system was split into two separate networks (each starting with a copy of all account and player data present at the time of the division): The Asian network, and United States and European network. Each network was comprised of a number of different servers that players could connect to in an optimized fashion based on server response time.

Some time later, even this system proved untenable. The network was once again permanently fragmented, this time splitting the United States and European network into three subnetworks. This is the topology retained today, with the

networks designated “USEast”, “USWest”, “Europe”, “Asia”. It is believed that all servers in a server network (also referred to as a “cluster” or “gateway”) are, at present, located at the same physical hosting facility on a high-speed LAN.

As new game requirements came about, a new architecture for Diablo II and Warcraft III as required. In these cases, games are hosted on Blizzard-operated servers and not on client machines in order to make them more resilient from attempts to hack the game to gain an unfair advantage. There are significant differences to how this is implemented for Diablo II and Warcraft III, and it is not used for certain types of games in Warcraft III . This resulted in a significant change to the way the service performs it’s primary function, that is, game matchmaking.

3.2 Client/Server issues

Aside from the basic network design issues, other problems have arisen from the fact that Blizzard did not expect, or intend for, third party programs to use its Battle.net protocol. As a result, proper validation has not always been in place for certain conditions that would not be generated through the Blizzard client software.

As mentioned earlier, many developers eventually turned to the using the Battle.net protocol directly as opposed to the text-based protocol in order to circumvent certain limitations in the text-based protocol. There are a number of reasons for this. Historically, clients utilizing the Battle.net protocol have been able to enter channels that are already full (private channels on Battle.net have a limit of 40 users, normally), and have been able to perform various account management functions (such as creating accounts, changing passwords, managing user profile information, and so-forth) that are not doable through the text-based protocol.

In addition to having access to extended protocol-level functionality, clients using the Battle.net protocol are permitted to open up to eight connections to a single Battle.net network per IP address (as opposed to the text-based protocol, which only allows a single connection per IP address). This limit was originally four connections per IP address, and was raised after NATs, particularly in cyber cafes, gained popularity.

This was particularly attractive to a number of persons on Battle.net who used third-party chat clients for a variety of reasons. The primary reason was generally the same “channel war” phenomenon that has historically plagued IRC was also rather prevalent on Battle.net, and being able to field a large number of clients per IP address was seen as a significant advantage.

Due to the prevalence of “channel wars” on Battle.net, artificially large numbers

of third-party clients utilizing the Battle.net protocol came into use. Although it is difficult to estimate the exact number of users of such clients, the author has observed upwards of several thousand being logged on to the service at once.

The development and usage of said third party clients has resulted in the discovery of a number of other issues with Battle.net. While most of the issues covered here are either already fixed or relatively minor, there is still value in discussing them.

3.2.1 Client connection limits

Through the use of certain messages in the Battle.net protocol, it is possible to enter a channel beyond the normal 40 user limit. This was due to the fact that the method a game client would use to return to a chat channel after leaving a game would not properly check the user count. After miscreants exploited this vulnerability to put thousands of users into one channel, which subsequently lead to server crashes, Blizzard finally fixed this vulnerability.

3.2.2 Chat message server overflow

The server software often assumed that the client would only perform 'sane' actions, and one of these assumptions dealt with how long of a chat message a client could send. The server apparently copied a chat message indicated by a Battle.net protocol client into a fixed 512-byte buffer without proper length checking, such that a client could crash a server by sending a long enough message. Due to the fact that Blizzard's server binaries are not publicly available, it would not have been easy to exploit this flaw to run arbitrary code on the server. This serious vulnerability was fixed within a day of being reported.

3.2.3 Client authentication

Aside from general sanity checks, Blizzard also has had some issues relating to authentication. Blizzard currently has two systems in use for user account password authentication. In order to create a third party client, these systems had to be understood and third party implementations reduced. This has revealed several flaws in their implementation.

The first system Blizzard utilizes is challenge-response system that uses a SHA-1 hash of the client's password. The game client implementation of this system lowercases the entire password string before hashing it, significantly reducing password security. (A third party client could opt not to do this, and as such create an account that is impossible to log on to through the official Blizzard game clients or the text-based protocol. The text-based protocol sends a user's

password in cleartext, after which the server lowercases the password and internally compares a hash of it with the account in question's password in a database.) However, a more serious security problem remains: in SHA-1, there are a number of bit rotate left ("ROL") operations. The Blizzard programmer responsible for implementing this apparently switched the two parameters in every call to ROL. That is, if there was a "#define ROL(a, b) (...)" macro, the programmer swapped the two arguments. This drastically reduces the security of Battle.net password hashes, as most of the data being hashed ends up being zero bits. Because of the problem of incompatibility with previously created accounts, this system is still in use today.

The second system Blizzard utilizes is one based off of SRP (Secure Remote Password, see http://srp.stanford.edu). Only Warcraft III and it's expansion use this system for password authentication. This product has it's own account namespace on Battle.net, so that there are no backwards compatibility issues with the older "broken SHA-1" method. It is worth noting that Warcraft III clients and older clients can still communicate via chat, however - the server imposes a namespace decoration to client account names for communication between namespaces, such that a client logged on as Warcraft III would see a user "User" logged on as Starcraft on the USEast Battle.net network as "User@USEast". However, this system is also flawed, albeit less severely. In particular, the endian-ness of calculations is reversed, but this is not properly accounted for in some parts of the implementation, such that some operations expecting to remove trailing zero bits instead remove leading zero bits after converting a large integer to a flat binary buffer. There is a second flaw, as well, although it does not negatively impact the security of the client: In some of the conversions from big numbers to flat buffers, the server does not properly zero out bytes if the big number does not occupy 32 non-zero bytes, and instead leaves uninitialized data in them. The result is that some authentication attempts will randomly fail. As far as the author knows, this bug is still present in Battle.net.

3.2.4 Client namespace spoofing

With the release of Warcraft III, a separate account namespace was provided for users of that product, as mentioned above. The server internally keeps track of a user's account name as "x#username", where x is a digit specifying an alternate namespace (the only currently known namespace designation is 'w', for Warcraft III). This is known due to a message that exposes the internal unique name for a user to protocol clients. While the character '#' has never been permitted in account names, if a user logs on to the same account more than once, they are assigned a unique name of the format 'accountname#serial', where 'serial' is a number that is incremented according to how many duplicate logons of the same account there are. Due to a lack of parameter checking in the account creation process, it was at one time possible to create accounts,via a third party client,

that were one character long (all of the official game clients do not allow the user to do this). For some time, such accounts confused the server into thinking that a user was actually on a different (non-existent) namespace, and thus allowed a user who logged on to a single character account more than once to become impossible to 'target' via any of the user management functions. For example, such a user could not be sent a private message, ignored, banned or kicked from a channel, or otherwise affected by any other commands that operate on a specific user. This was, of course, frequently abused to spam individuals with the victims being unable to stop the spammer (or even ignore them!). This problem has been fixed in the current server version.

3.2.5 Username collisions

As referred to in the previuos sub-section, for some time the server allowed Diablo Shareware clients. These clients did not log on to accounts, and instead simply assigned themselves a username. Normal procedures were followed if the username was already in use, which involved appending a serial number to the end to make a unique name. Besides the obvious problem of being able to impersonate someone to a user who was not clever enough to check what game type one was logged on as, this creates an additional vulnerability that was heavily exploited in "channel wars". If a server became split from the rest of the network due to load, one could log on to that server using Diablo Shareware, and pick the same name as someone logged on to the rest of the network using a different game type. When the server split was resolved, the server would notice that there were now two users with the same unique name, and disconnect both of them with the "Duplicate username detected." message[1]. This could be used to force users offline any time a server split occurred. Being able to do so was desirable in the sense that there could normally only be one channel operator in a channel at a time (barring server splits, which could be used to create a second operator if the channel was entirely emptied and then recreated on the split server). When that operator left, the next person in line would be gifted with operator permissions (unless the operator had explicitly 'designated' a new heir for operator permissions). So, one could "take over" a channel by systematically disconnecting those "ahead of" one's client in a channel[2].

3.2.6 Server de-synchronization

At one time, a race condition such that if a malicious user were to log on to two connected (i.e. not-split) servers at the same time, the two servers would cease to communicate with another, causing a server split to occur. It is difficult to provide an exact explanation for why this would occur given the collision

[1] This is synonymous with the "colliding" exploits of old that used to plague IRC
[2] A channel is ordered by a user's age in the channel

elimination mechanism described above for users that are logged on with the same unique name, but it is assumed that in the process of synchronizing a new user between servers, there is a period of time where that a second server can also attempt to synchronize the same user and cause one of the servers to get into a invalid state. According to observations, this invalid state would eventually be resolved automatically, usually after 10-15 minutes.

3.2.7 Seeing invisible users

Battle.net administrators have the ability to become invisible to normal users. However, until recently, this was flawed in that the server would expose the existence of an invisible user to regular users during certain operations. In particular, if one ignores or unignores a user, the server will re-send the state of all users that are ignored or unignored in the current channel. Before this bug was fixed, this list included any invisible users[3].

3.2.8 Administrative command discovery

Originally, Battle.net would provide no acknowledgement if one issued an unrecognized chat command ("slash-command"). Blizzard later changed the server software to respond with an error message if a user sent an unknown command, but the server originally silently ignored the command if the user issued a privileged (administrator-only) command. This allowed end users to discover the names of various commands accessible to system administrators.

3.2.9 Gaining administrative privileges

Due to an oversight in the way administrator permissions are assigned to Battle.net accounts, it was at one time possible to overwrite the account of an administrator with a new account and keep the special permissions otherwise associated with the account. (An account can be overwritten like so if it has not been accessed in 90 days). This could have very nearly resulted in a disaster for Blizzard, had a more malicious user discovered this vulnerability and abused such privileges.

3.2.10 Obtaining passwords

Eventually, Blizzard implemented a password recovery mechanism whereby one could associate an e-mail address with an account, and request a password

[3]It is worth noting that the official game clients will ignore any unknown users returned in the state update message, so this vulnerability could only be utilized by a third party client

change through the Battle.net protocol for an account at logon time. This would result in an e-mail being dispatched to the registered address. If the user then replied to the mail as instructed, they would be automatically mailed back with a new account password. Unfortunately, as originally implemented, this system did not properly perform validation on the confirmation mail that the user was required to send. In particular, if a malicious user created an account "victim" on one Battle.net network, such as the Asian network, and then requested a password reset for that account, they could alter the return email slightly and actually reset the password for the account "victim" on a different Battle.net network, such as the USEast network. This exploit was actually publicly disclosed and saw over a day of heavy abuse before Blizzard managed to patch it.

Chapter 4

Battle.net server emulation

Blizzard 'declared war' on the programmers of servers that implement the Battle.net protocol some time ago when they took the developers of "bnetd" to court. As of Warcraft III, they have taken active measures to make life difficult for developers programming third party Battle.net-compatible servers. In particular, two actions are of note:

During the Warcraft III Expansion beta test, Blizzard implemented an encryption scheme for the Battle.net protocol (this was only used during the beta test and not on production Battle.net). This consisted of using the RC4 cipher to encrypt messages send and received from the server. The tricky part was that Blizzard had hardcoded constants that were encrypted using the cipher state, but never actually sent on the wire (these constants were different for each message). This made implementing a server difficult, as one had to find each magic constant. Unfortunately, Blizzard neglected to consider the policy of someone releasing a hacked version of the client that zeroed the RC4 initialization parameters, such that the entire encrypted stream became plaintext.

After several patches, Blizzard implemented a scheme by which a Warcraft III client could verify that it was indeed connecting to a genuine Blizzard Battle.net server. This scheme worked by having the Battle.net server sign it's IP address and send the resulting signature to the client, which would refuse to log on if the server's IP address did not match the signature. However, in the original implementation, the game client only checked the first four bytes of the signed data, and did not validate the remaining (normally zero) 124 bytes. This allows one to easily brute-force a signature that has a designed IP address, as one only has to check 32 bits of possible signatures at most to find it.

Chapter 5

Conclusion

Developing a platform to support a diverse set of requirements such as Battle.net is certainly no easy task. Though the original design could have perhaps been improved upon, it is the author's opinion that given what they had to work with, Blizzard did a reasonable job of ensuring that the service they set out to create stood the test of time, especially considering that support for all the future features of their later game clients could not have been predicted at the time the system was originally created. Nevertheless, it is the author's opinion that a system designed where clients are untrusted and all actions performed by them are subject to full validation would have been far more secure from the start, without any of the various problems Blizzard has encountered over the years.