

Trust in Security Architectures for MMOG and DRM
Henry Nguyen
Eastern Michigan University

Abstract

Client-Server technology and models have been in use for the greater part of current computing. It has evolved into uses for both Massively Multiplayer Online Games (MMOGs) and Digital Rights Management (DRM). Security concerns have been focused on protecting data that's on the client, server, and what's transmitted. The following is an exploration into the history and architecture of the client-server model, DRM, and MMOGs. Specifically, where the technology has been and where it is now. The goal is to help more clearly define the data flow that exists between the client and server. The data flow itself can then be further broken down into One-Stop Data Flow and Persistent Data Flow categories. Related to data flow in general is a brief look into Trusted Computing which could lead to future related works concentrating on Operating System level DRM implementation (e.g. Microsoft Windows Vista) and web based games that run within the Java applet. By looking at the historical paths of client-server, DRM technologies, and MMOGs along with defining data flow and non-data flow categories, one can see that the evolution and functionality of current media is going to converge. This particular point of overlap is the moment of potential for the enhanced development of more secure data transactions and trust.

Trust in Security Architectures for MMOG and DRM

1. Client-Server Architecture

1.1. Introduction: Client-Server

The client-server architecture (CSA) has become one of the standards used in computer networking. Traditionally, the sharing of machines came from users who were connected to large mainframes by terminals. Those users shared CPU cycles and storage on the mainframes. Eventually, in the 1980's desktop computing experienced a huge growth. Although basic computing could be done on a desktop, they were incapable to handling large amounts of data processing that was needed by businesses and corporations.[1] In the 1990's even though desktop machines become even more powerful, the establishment of PCs and LANs in the corporate environment maintained that most of the mission critical applications remained on the mainframe. The CSA grew out of the balance in shifting the mainframe applications in order to "downsize" to PC servers, database servers, and LANs. The CSA can be one or more clients with one or more servers. The essence is that they form a composite, distributed system.

1.2. Client-Server Architecture

The client-server system can be simplified by thinking of them as the relationship between two programs. The client is defined as one program that makes a request of a service. The server is a program that provides the service and ultimately fulfills the request.[2] A single machine can both be a client and server depending on the type of software installed. Within the client-server architecture, there are multiple sub-types such as two tier client-server architectures, three tier client-server architectures, three tier architecture with transaction processing (TP) monitor technology, three tier architecture with message server, three tier architecture with an application server, three tier with an ORB architecture, and distributed/collaborative enterprise architecture among others.[2]

1.2.1. Two Tier Client-Server Architecture

To better understand interaction and data flow, a look at the most basic type of client-server architecture, the two tier architecture, is needed. The two tier architectures has three components that are distributed in two layers (client and server). The three components are:

- User System Interface (session, text input, dialog)
- Processing Management (process development, process enactment, process monitoring, and process resource services)
- Database Management (data services and file services)

The user system interface is tied to the client. The database management is on the server. The processing management is split between the client-server. So there's only two layers involved.[3]

The user system interface (client) invokes a request to the database management (server). The application portion of processing is handled by the client. Processing in relation to accessing data is handled on the database management side and then replies to the request. The communication between the two is made by remote procedure calls (RPC) or standard query language (SQL) statements.[4][5] See figure 1.

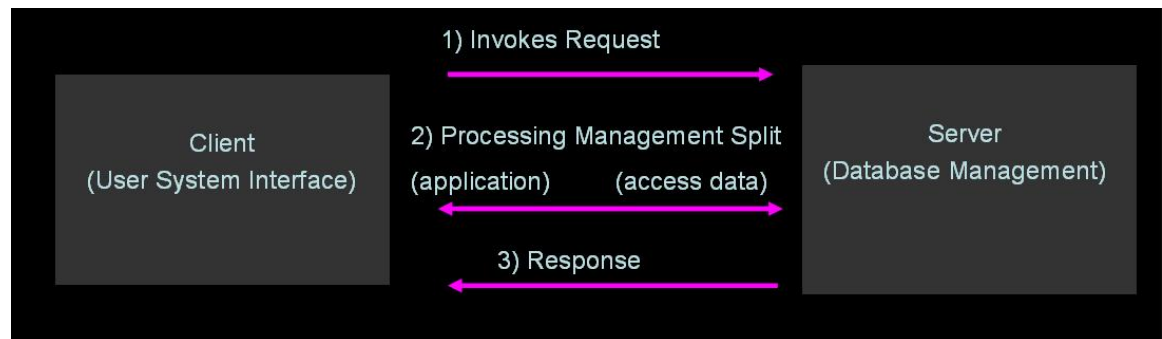


Figure1.

1.3. Example of Client-Server

The best example to illustrate a two-tier client server architecture example is the use of a web browser to look at websites. The client side program would be a web browser such as Mozilla Firefox or Microsoft Internet Explorer. The server side contains web servers, database servers, e-mail servers, FTP servers, or many other kinds of servers used to respond to queries made by the client. “The common use of client/server model is in Computer transactions. For example, if you have to check a bank account from your computer, you have to send a request to a server program at the bank. That program process the request and forward the request to its own client program that sends a request to a database server at another bank computer to retrieve client balance information. The balance is sent back to the bank data client, which in turn serves it back to your personal computer, which displays the information of balance on your computer.”[6] The banking example given, although multi-tiered, is still an example of a client-server architecture.

2. DRM

2.1. Introduction: DRM

Digital rights management (DRM) is a general term used to describe a way to manage copyright, distribution, and content control of digital media such as music and movies. Due to the general nature of a term such as DRM, there have been several different types of implementation for digital management. To clarify the different DRM

methods, it has been divided into two separate camps: Data Flow and Non-Data Flow methods with supporting examples under each.

- Non-Data Flow:
 - o DVDs
 - o CDs
 - o Blu-Ray & HD-DVDs

- Data Flow:
 - o Downloaded Music

The categorization of data flow is based upon whether or not there is data that's traveling across the network in a client-server interaction or if the DRM management exists only between the media and player. Examples of each will be discussed to more clearly define the DRM techniques.

2.1.1. Non-Data Flow

2.1.1.1. DVDs

One of the first widely used examples of DRM implementation was the Content Scrambling System (CSS). CSS was used in 1996 as a means to employ control over the DVD market by having encryption placed on film DVDs and only able to be decrypted with hardware that was licensed. In September 1999, a teenager from Norway, named Jon Johansen, reverse engineered a licensed DVD player and obtained the player keys and other information needed to decrypt CSS. The program named DeCSS was made in order to decrypt DVDs. The intention of Johansen was to enable a DVD player on Linux to play DVD movies. However, the result was that the DRM scheme was broken and DeCSS allowed users to decrypt and copy the content of the DVDs onto a hard drive with the film quality identical to that of the original. [9] Because the solution of content encryption and decryption was hardware based along with licensing agreements from the manufacturers of the DVD playing equipment, that meant that once the decryption method was compromised, all of the future discs could not change the DRM keys due to the need for backwards compatibility with older players.

2.1.1.2. CDs

CDs also have used DRM technology to try to protect music from being copied and distributed illegally. Copy-protection technology must be able to stop the copying software on a computer from reading the music files, yet the reading of the files must be easily done on computer player software and also all other ordinary CD playing devices. To do this, there could be two ways. The first is a Passive Protection method. This method tries to confuse the copying software by exploiting the subtle reading differences between the way

audio players read the disc and computers read the disc. The second method is Active Protection. This method assumes that the computer will read all music files on the disc, so it installs software on a computer that will interfere with software that attempts to read the disc.

One controversial instance is the “rootkit” incident involving Sony-BMG. In this example, in 2005, Sony-BMG used two separate anti-copying technology software developed by British company First4Internet and US company SunnComm, called XCP and MediaMax respectively. Both of the software targeted Microsoft Windows. [10] The problem with the solution was that it installed the program unknowingly on users machine and also installed a component that hides itself. The other problem is that the program and process left many computer users with programs freezing up, slow applications, and behaving like the equivalent of having spyware installed. The XCP and MediaMax software was essentially a rootkit that based it’s design on malware methods. The rootkit in hiding itself created a hiding place for other malware programs to reside. This coupled with the nearly impossible-to-uninstall program has made both the active protection and non-active protection DRM methods a less than viable solution to security.

2.1.1.3. Blu-ray Disc & High Definition DVDs

As new high definition media, Blu-ray disc (BD) and High Definition DVDs (HD DVD), is introduced into today’s markets. The DRM technology behind the high definition media is named the Advanced Access Content System (AACS) and is the successor to the CSS DRM system used in DVDs. It uses the Advanced Encryption Standard (AES) with 128-bit keys, making the data on the disc more secure. Similar to DVD DRM techniques, HD DVD and Blu-ray drives/players have device/player keys that ship with the unit to decode the encrypted information on the disc. [16][20]

Taking a lesson learned from CSS DRM and the DeCSS problem, the AACS Licensing Authority has included a built-in method to handle the revocation of any key sets that have been compromised. The way it’s done is by adding a key management process that uses a Media Key Block (MKB). “Device Keys are used to decrypt one or more elements of a Media Key Block (MKB), in order to extract a secret Media Key” [17] In other words, all discs containing AACS encryption also has a MKB that allows the decryption of data. This MKB can be updated in future titles by the AACS Licensing Authority to revoke a compromised key thereby not allowing it to be used with future titles. So a legitimate but poorly secured player with a compromised key can no longer be used with future titles. It’s a message to manufacturers to make sure that the devices made are secure, don’t allow easy user compromises, and that the device keys are treated confidentially.[16]

In the first quarter of 2007, hackers managed to find a way to compromise and publish the hexadecimal encryption key used for HD DVD discs.[19] As of February 19, 2008 Toshiba announced the discontinuation of HD DVD media and players. [18] The focus from groups attempting to compromise high definition media is now on Blu-ray.

While AACS is the basis of both HD DVD and Blu-ray, the Blu-ray Disc Association (BDA) added features to BD to make it more secure than just AACS alone. The additions are called BD+ and ROM Mark.

BD+ allows the BDA to update the entire encryption system of new players and existing players that have already shipped and are in living rooms. If the encryption is compromised or cracked, new media will include information that changes the players' decryption code.[16] This is different than the revocation system in AACS because it can retroactively affect the existing library of movies and not just newer titles.

ROM Mark is a security addition designed to combat industrial-scale piracy, takes place during the manufacturing of Blu-ray discs. A legitimate Blu-ray disc has a unique and undetectable identifier (detectable by the player) that can refuse to play discs without ROM Mark.

2.1.2. Data Flow

The main difference between Non-Data Flow and Data Flow technologies is the inclusion of the internet and the client-server interaction involved. Whether it's using a downloadable music service such as iTunes or a downloadable movie service such as Vongo, each service requires users to download client software that will enable DRM of some kind to manage rights to the digital content. Depending on the service used for movies, some "terms of use" can be more restrictive than others. Also different for movies are due to the larger file size compared to music files, movies could be either downloaded or streamed. Each of those options has its own restrictions depending on the service used.

2.2. Data Flow Architecture

2.2.1. iTunes

In order to see the importance of DRM Data Flow, a step by step look into the process of how Apple implemented DRM is examined here. Apple's iTunes client software which allows users to download and play music uses the FairPlay software based DRM system.[11] When the content owner uses DRM technology, they encrypt their content. When the user gets the content, the owner also delivers a license and decryption key for the media player to use.

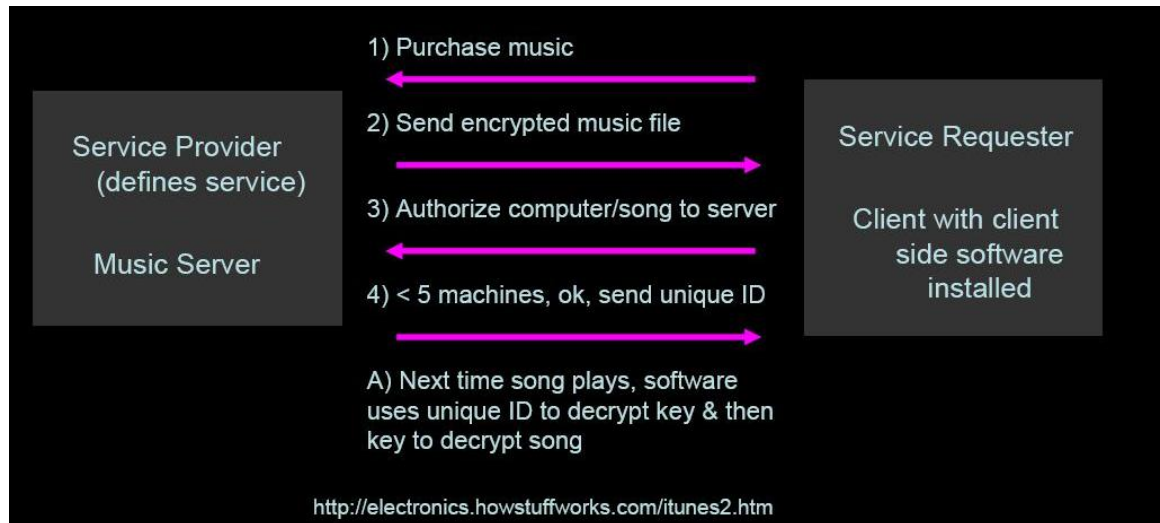


Figure 2

See Figure 2. There is the assumption that the process of purchasing the downloaded content is complete and secure. See figure 2, step 1 and step 2. To play the purchased file, the FairPlay DRM will go through the process to authorize. In this phase when the player tries to play the encrypted song, the computer generates a unique ID and sends it to the iTunes server requesting authorization. See figure 2, step 3. If the user's has less than five authorizations on his/her account, the server adds the unique ID to the user's account and sends back a decryption key to the iTunes player. This key is encrypted so someone else can't use the key other than the intended user's machine. See figure 2, step 4. The machine is now authorized to play the FairPlay protected songs. The next time that song is selected to play, iTunes client software uses the authorized ID to decrypt the key and then uses that decrypted key to play the song. See figure 2, step A. [12]

The problem to this method of DRM is that the resulting subsequent plays already assume that the authorization is legitimate. Jon Johansen, of DeCSS fame, in 2004 released code that disabled Apple's DRM scheme on the user's end, enabling Linux users to play FairPlay protected files on Linux systems.

2.2.2. Download/Streaming Movies

Downloadable music has been successful and the next step is for downloads of movies to start penetrating the market. The way it works is similar to music downloads. But the execution varies from service to service. Some require download managers that will check, after a user logs into his/her account, the settings that the user chose earlier. For instance, some services could check parental control preferences. After the delivery system goes through its checks, such as such as verifying the billing for a pay-per-movie service, verifying the date if it's a monthly service, or combinations of both, the delivery of the movie will start and it'll let the user know when he/she could start watching. Again,

depending on the service selected, each employs a different method of DRM. PC Computer users will likely be required to have Windows XP to run the client DRM software. Some services will require Internet Explorer and Windows Media Player. Other services, like MovieLink, will only work with Internet Explorer with Active X enabled and not work with Mozilla Firefox. Amazon Unbox requires the Microsoft.net framework. The Vongo service requires proprietary software. The terms of use is also dependent on the movie service. Some will let users play movies on multiple machines or even burn them onto a DVD. Yet others will only let a DVD be made that will only play on the machine that created it.[13] On a higher level, the process for controlling movie content on a computer is much like how it's handled by music DRM for movies that are fully downloadable and is on whole fully licensed to play on a computer. As there's such a wide variety of DRM service restrictions, pay-per-play or monthly timed services, the intricacies are due for deeper research on how each service is worked out in the Data Flow architecture. As it stands, the DRM control method is to control the content delivered after authentication, this is definitely the same for a single play through. Subsequent play through of the file (assuming it's downloaded and not streamed) is another matter, pending on the service.

3. Massively Multiplayer Online Games

3.1. Introduction: Massively Multiplayer Online Games

Massively Multiplayer Online Games (MMOG or MMO) is a video game played on the computer that is with hundreds or thousands of other players in a large and pervasive world. In the early onset of the MMO genre, the computers used to participate communicated using primitive graphics due to bandwidth limitations. As the computer graphics evolved in the middle 1990's, 3D games were becoming more popular but the bandwidth issue still persisted as the yield still was not high enough. So the machines were able to calculate large graphical data, yet the information sent to other machines was still statistical in nature to keep bandwidth down in an effort to prevent the lagging of a game.[14] The traditional two tier client-server model was used for games that were level based. For instance a sports game, such as soccer, would use the server to be a central point in communications between other computer players.

The most popular type of MMO today is the Massively Multiplayer Online Role Playing Game (MMORPG). Role playing games such as Ultima Online, Everquest, or World of Warcraft have been dominant in the MMO genre boasting up to 1 million subscribed members. Another game that's a popular MMO is Second Life. It's a game that doesn't have the medieval fantasy look, but it has just as many players around the world. The difference between previous MMOGs and current ones is the introduction of a large world that is persistent. Not only do players engage in logging onto a game server that is ongoing whether or not the particular player is playing, but the world itself is no longer regulated to a board or area. The game world is large

enough to hold thousands of players at any one time and players can go from one end of that world to another interacting with in-game characters or other players.

3.2. Architecture

Each “game server” is actually comprised of many servers linked together to facilitate a player moving from one section of a world’s map to another in a seamless transition. Although there may be 1 million people registered with a game, in order to make sure a single player’s slow connection doesn’t interfere with the other players, the games usually have multiple “game servers” that each have the game’s world on it. Some may have a player versus player option while others may not allow that option. The splitting and limiting the number of players makes sure that the computing power and data management required of the main game servers is handled properly. As a side reasoning, the limited number of players helps to ensure a player’s individuality by having a created character is not duplicated by a different player due to the limited character customization tools. In the end, each interaction between the player and the game world must still utilize a client-server model.

3.2.1. MMORPG Data Flow Example

In an MMORPG, the process for Data Flow is as follows.

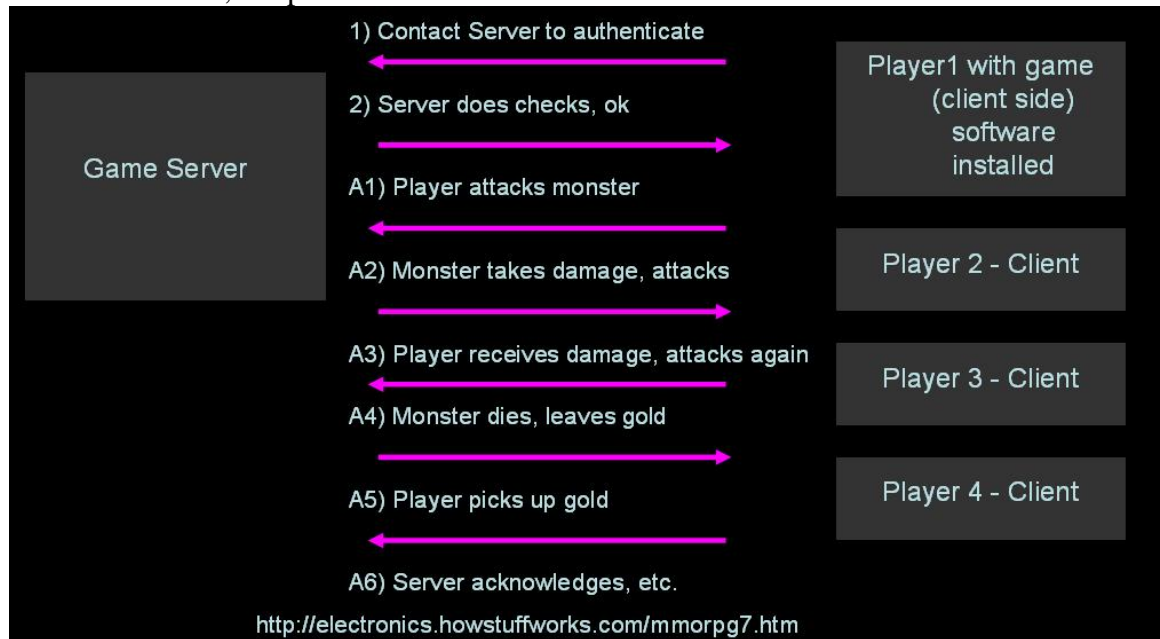


Figure 3.

See figure 3. Each player or client on the right side, much each go through an initial server authentication to validate the authenticity of the game, the verification of being able to use the service (e.g. the monthly service fee has not expired), and the client’s game version is updated to the current version. See Figure 3, step 1 and step 2. Next each player interacts with the game server in the game fashion (step A1-A6). For instance for Player 1, if the player decides to

interact with the game world (e.g. attacking a monster that's near by) that data is sent to the game server (see step A1). The server calculates the amount of damage done to the monster and also sends out the amount of damage the monster gives out (see step A2). The client takes that data and translates it to be graphically shown to the player. The player then attacks (interacts) with the monster (server) again (see step A3). The server calculates if the damage dealt is necessary for the monster to die. Then it sends that information back to the client along with the effect of slaying the monster, in this case, gold (see step A4). The client interacts again with the server showing the player picking up the gold (see step A5). The last step is that the sever acknowledges the request that the gold is picked up (step A6).[15]

Since the game world is ongoing with interactions from all different players at all different times, the amount of data that's being transmitted from the client to the server and back is constant.

4. Data Flow defining Security

With preliminary research into current uses and trends from the client-server model, it is easier to understand the differences in Data Flow. In the DRM Data Flow case, it can be seen as a One-Stop Data Flow. The MMOG Data Flow is a Persistent Data Flow.

4.1. One-Stop Data Flow

In the DRM example, particularly Apple iTunes, the initial authentication process involves multiple steps in verifying and validating the song purchased with the iTunes music player. However, after the initial playing of the song, it seems that subsequent plays of the music only requires that the iTunes player has the necessary unique ID to decrypt the key needed to play the song. There doesn't seem to be any data flow back to the iTunes server afterwards. It just stops. After the validation, if one were to not be connected to the internet, thus the iTunes server, the music should still be able to be played on the particular machine.

4.2. Persistent Data Flow

In the MMOG example, the requirement that the client is always online and connected to the game server is much different than the One-Stop Data Flow. Because the game world is constantly running, the data that flows from the client, to the server, back to the client, must always be updated. Even if a player (player1) is standing still and not actively interacting with the game server, the data flow must still take place since a different player (player2) may be approaching the still player (player1).

4.3. Further Break Downs

The two types of data flow may be broken down further by looking at the variant types of DRM and more complicated interactions between client-server for MMOGs.

4.3.1. DRM Breakdown

What's still unknown is the frequency or amount of data that is needed for certain types of DRM such as the pay-per-play service or streaming. If a movie has a limit to play a total of three times, there maybe a need of some interaction between the server and client to keep track of the number of times played. The second possibility is that the number of times a movie may be played could be passed onto the client where the client software must keep track of the number of times played. Streaming content from movies may also have client player's check to see if that content is being recorded in any way. If that's the case, there must be some sort of reporting feature that communicates back to the server with that info.

Another variant in the movie downloading and DRM scheme is the new video game systems. Further research is needed, but as is the Microsoft Xbox 360 and Playstation 3 consoles are able to connect to the internet and owners can pay to download and watch movies. The DRM mechanisms could possibly use both hardware and client side software to determine movie validity.

4.3.2. MMOG Breakdown

The persistent data flow in MMOGs seem like they're only two-ways, with data flowing to and from the game server as a nexus to all machines. As games become more complicated, each server (within the a game server) may be delegated to not only a section of a map, but down to game basics such as an enemy server that keeps track of monsters, a weather server that maintains changing weather patterns, or even player attribute changes that update all players in an area with changes in statistics. Current MMOGs have multiple servers that interact with one another and possibly with multiple clients. So the data flow may be different since the client may receive data from multiple servers, thereby initiating a three-way persistent data flow.

Also if it's the case that multiple servers must interact with on another, the question is in the security of the data flow from one server to another. Is that data trustworthy?

5. Related & Future Work and Conclusion

As there are many roads this preliminary work and proposal can lead to, there's a definite amount of importance that can come from understanding data flow and where things need to be secure so that trust can be implemented whether it be DRM or MMOGs.

Related and Future works

5.1.1. Related works: Trusted Computing

5.1.1.1. Introduction: Trusted Computing

Trust, in relation to computer security can be defined in a general sense, but also in a technical manner. The focus on the general term will be used. However, to understand that better within the scope of computer security, the technical definition will be explored along with examples. Within the past several years a new paradigm shift has started to occur in the way security and trust is viewed. The Trusted Computing Group (TCG) has developed a model that is linked to trusted systems. It's not the same as a trust system where that has a reference monitor that's ideally resistant to tamper and assures trust. Basically, the TCG model is a way to standardize and make a device behave consistently and in specific ways.[7] The way it is done is by using hardware and software. An individual machine that has TCG will have built-in hardware (based upon the Trusted Platform Module-TPM) and software (Trusted Computing Module Software Stack-TSS). "Simply stated, TPM instruments hardware and software with core security technologies that can generate and store keys securely for use in digital certificates and encryption. These operations are accessed and controlled through standard TSS interfaces and readily available to security management software for file/folder encryption, secure e-mail, identity and access management, and remote access."[8]

5.1.1.2. Trusted Computing Architecture

In the traditional client-server model, the server is trusted since it's protected by many layers of security. So the content that's served by it is supposed to be trustworthy. The problem is when the content is released to the client. The transmission itself is an issue of trust as there's no control of the content after it leaves the server.[7] Through the journey and possibly once the content reaches the intended client, the threat can come from interception of the deliverable content and modification to the malware that may reside on the client. Strong client side security mechanisms are needed to prevent software or content attacks from occurring.

5.1.1.3. Trusted Computing Importance

The importance of Trusted Computing is in the notion of both hardware and software support of hardware (TPM and TSS) being integrated and leaving only the meta data content going back and forth from the client to server as the only possible compromised area. The less area is exposed, in this case hardware, the more robust and secure the checks can be for the actual data being transferred.

5.1.2. Future Related works

The proposed data flow clarifications also warrant a look into Microsoft Windows Vista's DRM system called the Protected Media Path. The gist of the program is that it won't allow DRM restricted content to play while there is unsigned/unverified software running on the machine. The theory is that if only sanctioned software is signed, all of the unsigned software can't circumvent DRM protected media if it can't read it.

Another thing to lead to possible further research is the popularity of web based online games. Those games run within the java applet. Java was designed as a type of sand-box for programs to run within, so that any program that was not secure could be contained and not jeopardize the security of the main system. The question is whether the games that are not really client-server based really more secure with the data that's exchanged with the server.

5.2. Conclusion

This paper has had an aim to clearly define the large boundaries, One-Stop Data Flow and Persistent Data Flow, of data flow in relation to client-server services currently in use. DRM and MMOG applications certainly fall under the client-server architecture. However, each has its own new implementation where it may not fall under current client-server definitions and large boundaries set in this paper. While downloaded content and DRM methods may fall within the large boundaries, the new streaming media and pay-per-play uses end up making it more difficult to put them under the defined data flow boundary. There's the possibility that the MMOG Persistent Data Flow definition could only be used in this small window of time. The data that's being used may grow to such amounts that a single server may not be able to handle the data as a single unit anymore. That said, there is great potential in this paper to explore and categorize dataflow. By looking into the history of the client-server model and DRM technologies, one can see the evolutions of both are starting to overlap onto the other's boundaries. With a better understanding of how client-server architecture works in current applications, the potential for development of secure data transactions and trust may be enhanced greatly.

References

Raimes, Ann (1999). Keys for Writers a Brief Handbook. Sample Documented Paper: APA Style, 154-162

- [1] Sinha, Alok (1992) <http://delivery.acm.org.ezproxy.emich.edu/10.1145/130000/129908/p77-sinha.pdf?key1=129908&key2=6862898021&coll=Portal&dl=GUIDE&CFID=25288899&CFTOKEN=20456547>
- [2] Carnegie Mellon Software Engineering Institute, software technology descriptions. Client Server. http://www.sei.cmu.edu/str/descriptions/clientserver_body.html
- [3] Carnegie Mellon Software Engineering Institute, software technology descriptions. Two Tier. <http://www.sei.cmu.edu/str/descriptions/twotier.html#512860>
- [4] Schussel, G. (1995) Client/Server Past, Present, and Future. <http://www.dciexpo.com/geos/dbsejava.htm>
- [5] Edelstien, H. (1994) Unraveling Client/Server Architecture. DBMS 7. May 1994. Volume 7, Issue 5, page 34. http://find.galegroup.com.ezproxy.emich.edu/itx/retrieve.do?contentSet=IAC-Documents&resultListType=RESULT_LIST&qrySerId=Locale%28en%2CUS%2C%29%3AHQE%3D%28_HR_%2CNone%2C41%29sn+1041-5173+and+iu+5+and+sp+34+and+vo+7+%24&sgHitCountType=None&inPS=true&sort=DateDescend&searchType=CCLSearchForm&tabID=T003&prodId=ITOF&searchId=R1¤tPosition=1&userGroupName=lom_emichu&docId=A15319460&docType=IAC
- [6] Jorwekar, S. (2005) <http://www.buzzle.com/editorials/2-20-2005-66015.asp>
- [7] Sandhu, R., Zhang, X. (2005) SESSION: Access management for distributed systems. P147-158. <http://portal.acm.org.ezproxy.emich.edu/citation.cfm?id=1063979.1064005>
- [8] The Enterprise Strategy Group (2006) Trusted Enterprise Security. https://www.trustedcomputinggroup.org/news/Industry_Data/ESG_White_Paper.pdf
- [9] Ferrera, G., Lichtenstein, S., Reder, M., Bird, R., & Schiano, W. (2004). *CyberLaw Text and Cases, 2e*. Ohio: Thompson South-Western. 378-379
- [10] Felten, E. , Halderman J.A., (2006) Digital Rights Management, Spyware, and Security. *IEEE Security and Privacy*. January/February. 18-23 <http://csdl.computer.org.ezproxy.emich.edu/dl/mags/sp/2006/01/j1018.pdf>

- [11] Geer, D. (2004) Digital Rights Technology Sparks Interoperability Concerns. *IEEE Computer Society*. December. 20-22
<http://csdl2.computer.org.ezproxy.emich.edu/comp/mags/co/2004/12/rz020.pdf>
- [12] Layton, J. (2006) How iTunes Works. *Howstuffworks.com* Retrieved on April 19, 2008 from
<http://electronics.howstuffworks.com/itunes4.htm>
- [13] HowStuffWorks.com (2007) How Movie Download Services Work. Retrieved on April 20, 2008 from <http://entertainment.howstuffworks.com/movie-download.htm>
- [14] Bogojevic, S., Kazemzadeh, M. (2003) The Architecture of Massive Multiplayer Online Games. Department of Computer Science, Lund University.
<http://graphics.cs.lth.se/theses/projects/mmogarch/som.pdf>
- [15] Wilson, T. (2007) How MMORPGs Work. *Howstuffworks.com* Retrieved on April 01, 2008 from <http://electronics.howstuffworks.com/mmorpg7.htm>
- [16] Anderson, N. (2006) Hacking Digital Rights Management. *Arstechnica.com* Retrieved on June 24, 2008 from <http://arstechnica.com/articles/culture/drmhacks.ars/3>
- [17] Advanced Access Content System (AACs) Technical Overview (informative) Retrieved on June 24, 2008 from
http://www.aacsla.com/marketplace/overview/aacs_technical_overview_040721.pdf
- [18] Toshiba Announces Discontinuation of HD DVD Business (February 19, 2008) Press release. Retrieved on June 24, 2008 from
<http://hddvd.highdefdigest.com/pressreleasetoshiba021908.html>
- [19] Bangeman, E. (2007) HD DVD Cracks: there's no going back. *Arstechnica.com* Retrieved on June 24, 2008 from <http://arstechnica.com/news.ars/post/20070502-hd-dvd-cracks-theres-no-going-back.html>
- [20] Suba, F. (2007) Usability and Security of DRM Architectures. Helsinki University Technology. http://www.tml.tkk.fi/Publications/C/23/papers/Suba_final.pdf