Skype Fingerprint

Ronald C Dodge, JR.
Department of Electrical Engineering and Computer Science
United States Military Academy
West Point, New York 10996
ronald.dodge@usma.edu

Abstract

The wealth of data available about a person's computer activity is immense. Digital forensic sciences have progressed such that tools are readily available to recover deleted data, identify data hidden through obfuscation and other information hiding techniques, and construct usage profiles for each user. Each of these areas provides unique information about the target of the investigation. The third area, usage profiles, requires continual evaluation as new applications and services emerge. This goal of this paper is not to completely reverse engineer the Skype client. This paper will document a fingerprint for the widely popular Skype application. The fingerprint analysis will document profile information on a Windows system that is resident in the Windows registry and the hard disk file system.

1. Introduction

Communication using the digital infrastructure has changed how people communicate. Cellular phones increased the capability for people to contact other people from almost any location at any time. However, this connectivity came at a new cost – cell phone service and usage charges. Now, the internet infrastructure is being used to service voice communication by employing Voice over IP (VoIP) from either computer workstations or multi-mode phones [1, 2]. The largest footprint for commercial providers of VoIP uses the IETF Session Initiation Protocol (SIP) [3]. The preponderance of non-business users, however, that use VoIP technology (including many cell phone vendors), are using the proprietary Skype service [4]. The service footprint varies from report to report, however estimates of 34 million users in 2007 appear in several references. Skype provides the following services: signaling and session initiation, protocol for exchange of voice data, instant messaging, audio conferencing, and file transfer. Skype provides these services free for computer-to-computer connections however, offers several payment plans to connect to a cellular or typical PSTN phones (SkypeIN and SkypeOUT) [4]. Much like email, instant messaging, or web history files, the content of the Skype sessions, both header (or pen registry for the phone corollary) information and payload, can provide a wealth of data on the actions of a user. This goal of this paper is not to completely reverse engineer the Skype client. This paper is focused towards an analysis of the information that can be gleaned from a Skype installation and use on a client system. We will document the registry and file system locations where Skype (v3.2)

1530-1605 2008 U.S. Government Work Not Protected by U.S. Copyright

logs information about the users, data, and options.

The paper is organized as follows: In section 2, we present on overview of how Skype initiates and maintains connections between two peers. We present in section 3, previous analysis on Skype. In section 4, we describe the test bed and present the results of our analysis. We conclude our work in section 5.

2. Skype – making the connection

Skype utilizes a peer-to-peer (p2p) network to implement its multi service network. Skype was originally developed by Kazaa in 2003 and was purchased by Ebay in 2005. The Skype network consists of three different components; the Skype Client, Super Nodes, and Login Servers, as shown in Figure 1.

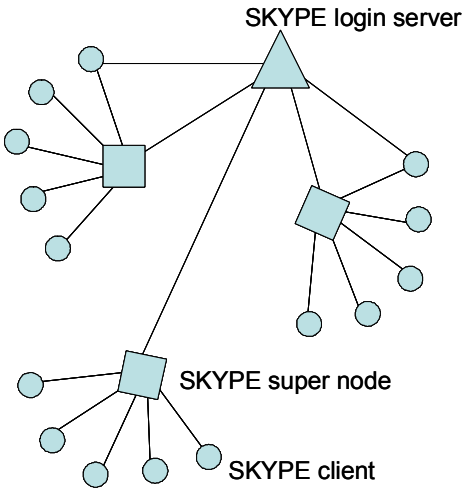

Figure 1: Skype Network

Any node that meets a minimal performance spec can become a Super Node. The user cannot control this behavior. In [9], the authors claim that over 20,000 super nodes exist. The network architecture allows Skype Clients to use the network of Skype Super Nodes to find a specific Skype User and support call connections for firewalled and/or NAT'd clients. The data in the call establishment packets is encrypted, so the specifics of the connection protocol are not openly available. It is believed that Skype uses a STUN [6] variant to assist in the case of a client node being behind a firewall or NAT.

When a client first starts, it will connect to a Skype web site on port 80 (ui.skype.com, our resolution was 212.72.49.131) and check for the most current version of Skype. The following TCP stream was observed (the current version is indicated by the box):

```
GET /ui/0/3.2.0.158/en/getnewestversion
HTTP/1.1
User-Agent: 3.2.0.158
Host: ui.skype.com
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Thu, 14 Jun 2007 17:47:12 GMT
Server: Apache
Cache-control: no-cache, must revalidate
Pragma: no-cache
Expires: 0
Set-Cookie:
SC=CC=:CCY=:LC=en:TM=118184323
2:TS=1181843232:TZ=:VER=0/3.2.0.15
8/0; expires=Fri, 13-Jun-08 17:47:12
GMT; path=/; domain=.skype.com;
Content-Length: 9
Connection: close
Content-Type: text/html; charset=utf-8
Content-Language: en

3.2.0.158
```

After the initial connection, the Skype client attempts to retrieve information about other Super Nodes over UDP. If it

cannot connect over UDP it attempts the same connection over TCP 80 then TCP 443. The payloads of these requests are encrypted. According to the Skype website [4], Skype uses 256 bit AES encryption for all signaling and data traffic. Skype uses 1024 bit RSA to negotiate symmetric keys.

The data gathered about Super Nodes called the host Cache is stored in the share config file described in section 4.2. When Skype is first installed, the Host Cache list is empty. It is reasonable to assume that the Skype Client application has a small set of default Super Nodes that it attempts to connect to, as indicated by the initial set of connections. When compared to a second fresh installation, the same IP addresses were observed (from a network trace using Wireshark). We will discuss more details of this file in section 4.

After establishing a list of Super Nodes and connecting to one of them, the Skype Client passes credentials to a Skype Login Server. Again, these packets are encrypted. In [7], the authors detected several Super Node IP addresses and two Login Server IP addresses; however, they were not the same as discovered in our experiments. Similarly, in [5], the IP addresses observed did not match those documented by the authors. This indicates either a robust and dynamic set of static Super Nodes and Login Servers or an infrastructure change since 2006. The authors in [7] and [5] provide a very detailed analysis of the network traffic created by Skype.

Once a user is logged into the Skype network, any of the services (VOIP, IM, File Transfer…) are available if a Skype *peer* is located. A Skype peer is simply another Skype Client. There are two ways to locate a peer Skye Client; either a peer is already stored in a "buddy list", or the client uses the Super Node network to located a requested peer. Skype claims that it can us a *Global Index* [8] to locate any user that has logged into Skype in the last 72 hours.

The encryption of all payloads, except for the version check, makes the Skype application very difficult to analyze and exploit from a network perspective. We will see in section 4, that while the majority of the data on the hard drive and in the Registry is also encrypted (or encoded), there are a few nuggets of data that are retrievable.

3. Previous Work

There has not been a tremendous amount of forensic analysis of Skype. The use of encryption, as previously noted, makes it very difficult to determine the details of the protocol. The administrator guide provided by the makers of Skype [5] provides a general functional explanation of the Skype network, however it does not provide any details of any fixed static network components or the protocols involved in call establishment. The administrators guide goes further to detail how to set up and install Skype in a variety of local network configurations. The authors in [6] provide one of the first analyses of Skype (2004). They detail from a network perspective, the call signaling, establishment, and tear down of connections in Skype as well as the sign-in process. The authors in [7] extend the work in 2006 by analyzing Skype version 2.0 and conclude with an IDS signature to detect the presence of Skype traffic on a network. This analysis relies on the sequencing of packets rather than the content of any specific packet.

Phillippe Biondi and Fabrice Descaux [9] briefed one of the most detailed and truly innovative explanations of Skype at Blackhat Europe in 2006. The briefing is hosted on many security websites – we retrieved a copy from [10]. In [9] the authors provide a clear description of methodologies to break the obfuscation scheme used by Skype to build an interpreter to display Skype packets.

4. Test Bed and Results

The results presented by the references cited in the preceding paragraph focus mostly on the workings of the Skype protocol – not on the artifacts left on a system by Skype. As mentioned earlier, much of the data stored on the computer is encrypted like the network traffic; however, a few items do remain. We built a very basic environment to examine the local data stored by Skype. Unlike the connection-focused analysis done previously, the way Skype connected is not the focus of this study – rather what is stored for history and state. For the test environment, we setup two computers running windows XP with the latest security patches. The Windows user account on the test systems was "Skype". Both of these computers were connected directly to the internet using static IP addresses. We proceeded with the tests by examining the following system state changes: Installation of Skype, Connecting to the Skype network for the first time, completing a voice call, and completing a chat (IM) session. For each of these tests we noted changes to the file system as well as registry updates. A commercial registry and file system monitor was used to record state changes.

4.1. Installation

The first step in determining the footprint Skype places on a system is to observe the installation and record the change made to the system state. After installation, Skype was turned off and the file system and registry comparison was completed.

During installation, Skype created 305 new files and directories. The majority of the files created were in:
C:\Documents and Settings\All Users\Application Data\Skype.
This majority of the content is plug-ins and wallpaper/images. Additionally, 909 registry keys and values were added. The keys control program options, display version information, and indicate local machine parameters. No win.ini or system.ini changes or additions were noted.

4.2. First Sign-on

At this point, no user-profile data has been stored in the file system or registry. To determine what system state changes take place when a user is added, Skype was started and then the monitoring application was started. After the current system state was noted, we created the first account and again, shut down Skype. After comparing this step with the state recently stored, we found 140 registry keys created, another 103 registry values changed, 97 files and folders created, and 10 files changed. Registry inspection by visual means and string searches based on values entered during the user creation process did not reveal any important registry values. Example registry values are shown below:

HKEY_CURRENT_USER\Software\Microsoft\Windows\ ShellNoRoam\MUICache C:\Program Files\Skype\Phone\ Skype.exe = "Skype. Take a deep breath "
HKEY_CURRENT_USER\Software\Skype\Phone

SkypePath = "C:\Program Files\Skype\Phone\Skype.exe"
HKEY_CURRENT_USER\Software\Skype\Phone\UI
Version = "50462878"
HKEY_CURRENT_USER\Software\Skype\Phone\UI\
General Language = "en"
HKEY_CURRENT_USER\Software\Skype\PluginManager
Country = "us"
HKEY_CURRENT_USER\Software\Skype\PluginManager
Language = "ENU"
HKEY_CURRENT_USER\Software\Skype\PluginManager
Locale = "us.ENU"
HKEY_CURRENT_USER\Software\Skype\PluginManager
PluginManagerDir = "C:\Program Files\Skype\Plugin Manager\"
HKEY_CURRENT_USER\Software\Skype\PluginManager
Plugins Root = "C:\Documents and Settings\All Users\Application Data\Skype\Plugins"
HKEY_CURRENT_USER\Software\Skype\PluginManager
RecentUsr = "ronald.dodge"
HKEY_CURRENT_USER\Software\Skype\PluginManager
Version = "1.2.0.257"

While the registry entries were not interesting, several files containing user profile information were created. The files created were in two directories:

- C:\Documents and Settings\skype\Application Data\Skype
and
- C:\Documents and Settings\skype\Application Data\Skype\ronald.dodge

Recall that the Windows user account on the test system was "Skype". The Skype user account created on the first test system was "ronald.dodge". The First directory contains two files: shared.lck and shared.xml. The shares.lck file is an empty file; regardless if Skype is running or not. It is not clear what this file is designed for. The second file is an XML file that contains data that, by the XML tags, is very interesting. This data however is encrypted or encoded in most cases. Some values appear to control basic settings such as microphone volume. The second system had two Skype accounts created. It is interesting to note that the first system had a <default> user tag with a value of ronald.dodge. This tag was missing on the second test system, where two user accounts were created.

The second folder contained several files that contained configuration data on the user and historical data from the Skype services. The interesting files created were:
- config.lck
- config.xml
- contactgroup256.dbb
- profile256.dbb
- user1024.dbb

The *config*.xml file contains configuration information for the current user logged in (such as an encrypted authentication string, as well as the users "buddy list", default location for transferred files, and a brief history of the last IM chat messages sent. The number of messages retained can be set in this file. The *contactgroup* file did not contain data that appeared useful. The *profile* file contains profile information for the current user (such as user ID and city/state). The *user* file contains information on the buddies in the buddy list. The entries for each buddy contain the buddy's profile information (like that in the profile file) and approximately 720 bytes that appear to be encrypted data. The size of his area varies slightly for each user in the buddy list. It is interesting to note that every separate section in each file begins with the ASCII sequence "1331". It is not evident why this is.

4.3. Examination of other Skype features

From this point, we followed a similar sequence of steps; recording the system state, then working through several of the other Skype features, such as voice calls, chat, and file transfer. Several additional registry values were added and updated; as well, several files were added to the user directory described in

the preceding paragraph. Each service that we tested added a new file to the user folder. Each file name had a maximum block size appended to the end (as shown below).
The significant files added were:
- chatmember256.dbb
- chat512.dbb
- chatmsg256.dbb
- transfer512.dbb
- call256.dbb

The *chatmember* file contains a list of usernames that the current user has chatted with. The *chat* and *chatmsg* files appear to contain the same information – a record of the text exchanges in the chat session. There is an option setting that allows the turning off chat logging. The *transfer* file (shown below) includes information such as the file transferred and the location it was saved.

l33l→ ┌) A◀ –
Á┐ ┌ └ Ä┐ ronald.dodge2 └ È┐ Ronald Dodge2 └ Ú^º¾└ └ à┐ New Text Document.txt Í┐ ◘ └ ä┐ 11 Õ┐ ŸÅË³┘ Ù┐ ÓÅË³┘ └ è┐ 11 • ÿÿÿÿ☼ ☼ Å©² ‼ ┘ A┌ ┌ 5ýgÏ–Y.E└ Ü┐ C:\Documents and Settings\skype\My Documents\New Text Document.txt └ ù• C:\Documents and Settings\skype\My Documents\New Text Document.txt

There is also other data in the file that is encoded, however we were not able to determine the decoded values. Finally, the *call* file contains information on who was called. As in the *transfer* file, additional data was present that appeared to be encoded.

5. Conclusions

The artifacts left on a system by Skype can indeed present valuable information to an investigator. We demonstrated that the registry is not used to store user activity as much as default Skype behavior. The data stored in the file system does contain valuable information that can be used to assist an investigator in determining the activity of a user. The data stored by Skype is a mix of clear text and either encoded or encrypted data. It is reasonable to expect future evaluations of the Skype storage system to shed additional light on the contents of these areas.

6. References

[1] Lawson, S. "Skype Sets Its Sights on Cell Phones," http://www.pcworld.com/article/id,119652-page,1/article.html, accessed on 1 June 2007.

[2] Charny, B., "VoIP cozies up to cell phones," http://news.zdnet.com/2100-1035_22-5759701.html, accessed on 1 June 2007.

[3] Rosenburg, J. and Schulzrinne H., "Session Initation Protocol," RFC 3261, 2002

[4] http://www.skype.com,, accessed on 1 June 2007

[5] "Guide for Network Admins," www.skype.com/security/guide-for-network-admins.pdf, accessed on 1 June 2007

[6] Baset S. and Schulzrinne H., "An analysis of the skype peer-to-peer Internet telephony protocol," Columbia University Technical Report, CUCS-039-04, September 2004.

[7] Ehlert, S., Petgang, S., Magedanz, T., and Sisalem, D., "Analysis and Signature if Skype VOIP Traffic," Proceedings of the Fourth IASTED International Conference, Communications, Internet, and Information Technology, Nov 29 – Dec 1, 2006, St. Thomas, US Virgin Islands

[8] http://www.skype.com/skype_p2explaijned.html,, accessed on 1 June 2007

[9] http://www.blackhat.com/html/bh-europe-06/bh-eu-06-speakers.html#Biondi, accessed on 1 June 2007

[10] http://www.secdev.org/conf/skype_BHEU06.handout..pdf, accessed on 1 June 2007