# A Concept of File Deletion and Restoration as a Threat to Commit Cyber Crime

**ANISHA KUMAR**

Department of Studies in Computer science, Pooja Bhagavat Memorial Mahajan PG Centre, Department of Computer Science, Mysore - 570 017 (India).

(Received: March 16, 2011; Accepted: April 17, 2011)

## ABSTRACT

Information Technology solutions have paved a way to a new world of internet, business networking and e-banking, budding as a solution to reduce costs, change the sophisticated economic affairs to more easier, speedy, efficient, and time saving method of transactions. Internet has emerged as a blessing for the present pace of life but at the same time also resulted in various threats to the consumers and other institutions which results in committing cyber crime. Despite the increase in government compliance requirements and the proliferation of security tools, companies continue to underestimate the threat from phishing, data loss, and other cyber vulnerabilities.This paper contributes an understanding of the effects of negative use of Information technology, as how a simple technological aspect of file deletion can result in cyber crime. A few aspects to trace the deleted information with restoring software is also mentioned in this paper.

**Key words:** File deletion; Restoring; Cyber crime; Internet

## INTRODUCTION

Cyber crime consist of any criminal act dealing with computers and networks called as hacking. Additionally, cyber crime also includes traditional crimes conducted through the Internet like telemarketing and Internet fraud, identity theft, and credit card account thefts, as these illegal activities are committed through the use of a computer and the Internet [1]. The first recorded cyber crime took place in the year 1820.That is not surprising considering the fact that the abacus, which is thought to be the earliest form of a computer, has been around since 3500 B.C. in India, Japan and China. The era of modern computers, however, began with the analytical engine of Charles Babbage. In 1820, Joseph-Marie Jacquard, a textile manufacturer in France, produced the loom. This device allowed the repetition of a series of steps in the weaving of special fabrics. This resulted in a fear amongst Jacquard’s employees that their traditional employment and livelihood were being threatened. They committed acts of sabotage to discourage Jacquard from further use of the new technology. This is the first recorded cyber crime [2].

In the field of computer security, phishing is the criminally fraudulent process. It is a form of Internet fraud that involves tricking people into revealing confidential information like credit card details, user names and passwords by means of a fake e-mail that appears to come from a well-known, legitimate organization like a bank or any institution heads .This paper gives a brief description on the concept of file deletion and restoration in windows which is an aid to lead to the phishing activity as a result of threat to commit a cyber crime.

### File allocation techniques

File allocation is done using different allocation methods such as contiguous allocation, linked allocation and indexed allocations depending on the disk partitions and mounting resulting in disk space. Each partition can either be “raw” containing no file system or “cooked” containing a file system. Raw disk is used as swap space, as it uses its own format on disk and not any file system[3]. It is also used for database purpose to format the data to suit some needs.

In general the process of file allocation using links helps for a better understanding about the concept of file deletion. With a linked allocation (see fig (i)), each file is a linked list of disk blocks, the disk blocks may be scattered anywhere on the disk. The File allocation table consist of link details as pointers to the starting address of a particular file (see fig (ii)).

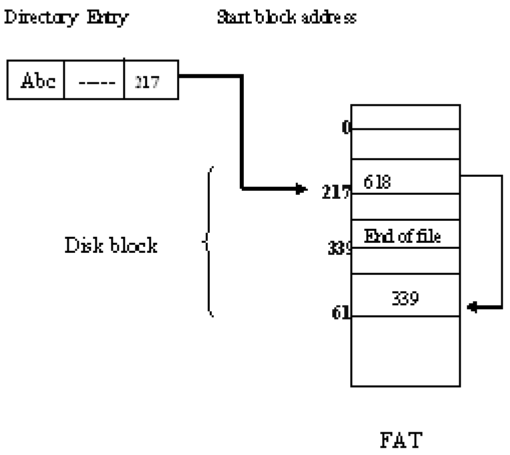

**Fig. 1: File allocation table**

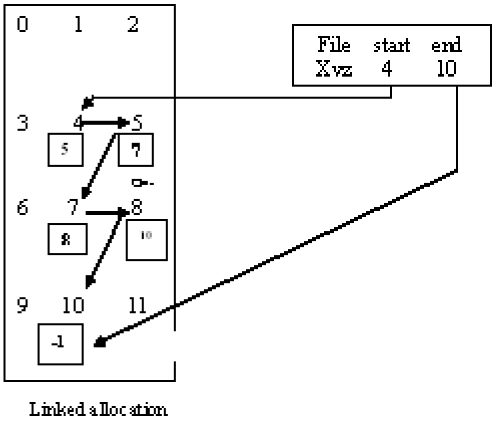

**Fig. 2: Linked Allocation**

Since disk space is limited, we need to reuse the space from deleted files for new files. To keep track of free disk space the system maintains a free space list which records all free disk blocks those not allocated to some file or directory. When a file is deleted its disk space is also added to the free space list.

**File deletion basic concepts**

When a file is deleted from the computer, its contents are not immediately destroyed. Windows simply marks the hard drive space as being available for use by changing one character in the file table so that the file entry won’t be displayed. That is an operating system first marks the space on the hard drive that the file occupied as a free space. Since the link is lost and the space is free to be overwritten by new data on top of the existing deleted contents, at this point file recovery process becomes a lot more difficult. Hence if a little computing is done the chance of retrieving the deleted file using restoration tools are high.

There are some other serious issues that could avoid the restoration as the hard drive is overwritten in cases where it is pretty full , the odds are much greater that Windows will grab your precious unallocated space for its next write. Or, if you defrag the hard drive, you run the risk of unused parts of the drive being overwritten. But these issues occur very rarely in personal computers so its always better to be safe from keeping our id’s or information secure by simply starting up Windows or, to a lesser extent, shutting down Windows which causes many tiny files to be written as the computation increases and restoration becomes difficult.

Restoration is a freeware program that restores deleted files no longer in the Recycle Bin and from disk space. There are many restoration software available like FileRestorePlus[2] which is a quick and effective way to restore accidentally deleted files. It can also recover files that have been emptied from the Recycle Bin, permanently delete files within Windows using the Shift + Delete, and files that have been deleted from within a Command Prompt.

Recovery software created to recover detail when a mobile phone software crashes or to recover details from a disk is similar to the basic common c source code given below for the restoration software work that was carried out in torus solutions.[4]

## Finding

### Program created for restoration

The list of header files included in the source programme created as a restoration software for the cause mentioned above is given below [4]

```
#include <ctype.h> // character type for upper and lowercase etc.
#include <errno.h>
#include <fcntl.h>// file control option header file for the purpose of setting flags etc.
#include <getopt.h>
#include <limits.h>
#include <linux/input.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/reboot.h>
#include <sys/types.h>
#include <time.h>
#include <unistd.h>

static const struct option OPTIONS[] = {
{ “send_intent”, required_argument, NULL, ‘s’ },
{ “update_package”, required_argument, NULL, ‘u’ },
{ “wipe_data”, no_argument, NULL, ‘w’ },
{ “wipe_cache”, no_argument, NULL, ‘c’ },
};

static const char *COMMAND_FILE = “CACHE:recovery/command”;
static const char *INTENT_FILE = “CACHE:recovery/intent”;
static const char *LOG_FILE = “CACHE:recovery/log”;
static const char *SDCARD_PACKAGE_FILE = “SDCARD:update.zip”;
static const char *TEMPORARY_LOG_FILE = “/tmp/recovery.log”;

/*
* The recovery tool communicates with the main system through /cache files.
* /cache/recovery/command - INPUT - command line for tool, one arg per line
* /cache/recovery/log - OUTPUT - combined log file from recovery run(s)
* /cache/recovery/intent - OUTPUT - intent that was passed in
*
* The arguments which may be supplied in the recovery.command file:
* —send_intent=anystring - write the text out to recovery.intent
* —update_package=root:path - verify install an OTA package file
* —wipe_data - erase user data (and cache), then reboot
* —wipe_cache - wipe cache (but not user data), then reboot
*
* After completing, we remove /cache/recovery/command and reboot.
*/

static const int MAX_ARG_LENGTH = 4096;
static const int MAX_ARGS = 100;

// open a file given in root:path format, mounting partitions as necessary
static FILE*
fopen_root_path(const char *root_path, const char *mode) {
if (ensure_root_path_mounted(root_path) != 0) {
LOGE(“Can’t mount %s\n”, root_path);
return NULL;
}

char path[PATH_MAX] = “”;
if (translate_root_path(root_path, path, sizeof(path)) == NULL) {
LOGE(“Bad path %s\n”, root_path);
return NULL;
}

// When writing, try to create the containing directory, if necessary.
// Use generous permissions, the system (init.rc) will reset them.
if (strchr(“wa”, mode[0])) dirCreateHierarchy(path, 0777, NULL, 1);

```

```
FILE *fp = fopen(path, mode);
if (fp == NULL) LOGE(“Can’t open %s\n”, path);
return fp;
}

// close a file, log an error if the error indicator is set
static void
check_and_fclose(FILE *fp, const char *name) {
fflush(fp);
if (ferror(fp)) LOGE(“Error in %s\n(%s)\n”, name, strerror(errno));
fclose(fp);
}

// command line args come from, in decreasing precedence:
// - the actual command line
// - the bootloader control block (one per line, after “recovery”)
// - the contents of COMMAND_FILE (one per line)
static void
get_args(int *argc, char ***argv) {
if (*argc > 1) return; // actual command line arguments take priority
char *argv0 = (*argv)[0];

struct bootloader_message boot;
if (!get_bootloader_message(&boot)) {
if (boot.command[0] != 0 && boot.command[0] != 255) {
LOGI(“Boot command: %.*s\n”, sizeof(boot.command), boot.command);
}

if (boot.status[0] != 0 && boot.status[0] != 255) {
LOGI(“Boot status: %.*s\n”, sizeof(boot.status), boot.status);
}

// Ensure that from here on, a reboot goes back into recovery
strcpy(boot.command, “boot-recovery”);
set_bootloader_message(&boot);

boot.recovery[sizeof(boot.recovery) - 1] = ‘\0’; // Ensure termination
const char *arg = strtok(boot.recovery, “\n”);
if (arg != NULL && !strcmp(arg, “recovery”)) {
* MAX_ARGS);//
(*argv)[0] = argv0; // use the same program name

char buf[MAX_ARG_LENGTH];
for (*argc = 1; *argc < MAX_ARGS && fgets(buf, sizeof(buf), fp); ++*argc) {
(*argv)[*argc] = strdup(strtok(buf, “\r\n”)); // Strip newline.
}

check_and_fclose(fp, COMMAND_FILE);
LOGI(“Got arguments from %s\n”, COMMAND_FILE);
}


// clear the recovery command and prepare to boot a (hopefully working) system,
// copy our log file to cache as well (for the system to read), and
// record any intent we were asked to communicate back to the system.
// this function is idempotent: call it as many times as you like.
static void
finish_recovery(const char *send_intent)
{
// By this point, we’re ready to return to the main system...
if (send_intent != NULL) {
FILE *fp = fopen_root_path(INTENT_FILE, “w”);
if (fp != NULL) {
fputs(send_intent, fp);
check_and_fclose(fp, INTENT_FILE);
}
}

// Copy logs to cache so the system can find out what happened.
FILE *log = fopen_root_path(LOG_FILE, “a”);
if (log != NULL) {
FILE *tmplog = fopen(TEMPORARY_LOG_FILE, “r”);
if (tmplog == NULL) {
LOGE(“Can’t open %s\n”, TEMPORARY_LOG_FILE);
} else {
static long tmplog_offset = 0;
fseek(tmplog, tmplog_offset, SEEK_SET); // Since last write
char buf[4096];
while (fgets(buf, sizeof(buf), tmplog)) fputs(buf, log);
tmplog_offset = ftell(tmplog);
check_and_fclose(tmplog, TEMPORARY_LOG_FILE);
}
check_and_fclose(log, LOG_FILE);
}

// Reset the bootloader message to revert to a normal main system boot.
struct bootloader_message boot;
memset(&boot, 0, sizeof(boot));
set_bootloader_message(&boot);

// Remove the command file, so recovery won’t repeat indefinitely.
char path[PATH_MAX] = “”;
if (ensure_root_path_mounted(COMMAND_FILE) != 0 ||
translate_root_path(COMMAND_FILE, path, sizeof(path)) == NULL ||
(unlink(path) && errno != ENOENT)) {
LOGW(“Can’t unlink %s\n”, COMMAND_FILE);
}

sync(); // For good measure.
}
if (key == KEY_DREAM_BACK && ui_key_pressed(KEY_DREAM_HOME)) {
// Wait for the keys to be released, to avoid triggering
// special boot modes (like coming back into recovery!).
while (ui_key_pressed(KEY_DREAM_BACK) ||
ui_key_pressed(KEY_DREAM_HOME)) {
usleep(1000);
}
break;
} else if (alt && key == KEY_W) {
ui_print(“\n”);
erase_root(“DATA:”);
erase_root(“CACHE:”);
ui_print(“Data wipe complete.\n”);
if (!ui_text_visible()) break;
} else if (alt && key == KEY_S) {
ui_print(“\nInstalling from sdcard...\n”);
int status = install_package(SDCARD_PACKAGE_FILE);
if (status != INSTALL_SUCCESS) {
ui_set_background(BACKGROUND_ICON_ERROR);
ui_print(“Installation aborted.\n”);
} else if (!ui_text_visible()) {
break; // reboot if logs aren’t visible
}
ui_print(“\nPress Home+Back to reboot\n”);

// If these fail, there’s not really anywhere to complain...
freopen(TEMPORARY_LOG_FILE, “a”, stdout); setbuf(stdout, NULL);
freopen(TEMPORARY_LOG_FILE, “a”, stderr); setbuf(stderr, NULL);
fprintf(stderr, “Starting recovery on %s”, ctime(&start));

ui_init();
} else {
status = INSTALL_ERROR; // No command specified
}

if (status != INSTALL_SUCCESS) ui_set_background(BACKGROUND_ICON_ERROR);
if (status != INSTALL_SUCCESS || ui_text_visible()) prompt_and_wait();

// If there is a radio image pending, reboot now to install it.
maybe_install_firmware_update(send_intent);

// Otherwise, get ready to boot the main system...
finish_recovery(send_intent);
ui_print("Rebooting...\n");
sync();
reboot(RB_AUTOBOOT);
return EXIT_SUCCESS;
}[4]
```

## CONCLUSION

In this paper the first level work on the concept of file deletion and restoration software is explained which can avoid cyber crime to some extend in personal computers. In the second level of work c programming language and concepts of pointers are used to retrieve the memory address space of deleted information which works like a restoration software. This work was carried out at Torus Solutions, Mysore under the guidance of Mr.Senthil Kumar and the concepts were studied with the help of Dr. P. Vinod.

## REFERENCES

1. (Dr) P. Vinod Bhattathiripad Cyber crime investigation consultant Kerala (http://www.saintgits.org/main/sie/Computer%20Science/Annual%) 20Report.asp A seminar conducted by Dr Vinod at kerala on 7th August,2009 as given in the annual report of the address mentioned above is the guidelines for basic ground work on cyber crime and file system.
2. http://download.cnet.com/Restoration/3000-2094_4-10322950.html
3. OperatingSystem Concepts, 5th Edition ,published in 2000 Authors : Silberschatz, Galvin, Gagne Chapters 10,11, pages 343-355,360-380
4. www.torussolution.com a software firm in mysore,Karnataka.