PandaLabs Quarterly Report

July - September **2011**

01| Introduction

Summer is gone and it is time to take a look back at everything that has happened in the last three months. The holiday season has not brought a halt to malware creation, as shown by the more than 5 million new malware strains created during this period.

Trojan creation has hit a new record high, as revealed by the fact that 3 out of every 4 new malware specimens created this quarter was a Trojan –the weapon of choice of today’s cyber-crooks.

We’ll take a look at the most important events in the security world: from the latest hacking activities of the Anonymous group, to the most recent acts of cyber-warfare and the new threats targeting cell phones, social networks, etc.

02| Q3 at a Glance

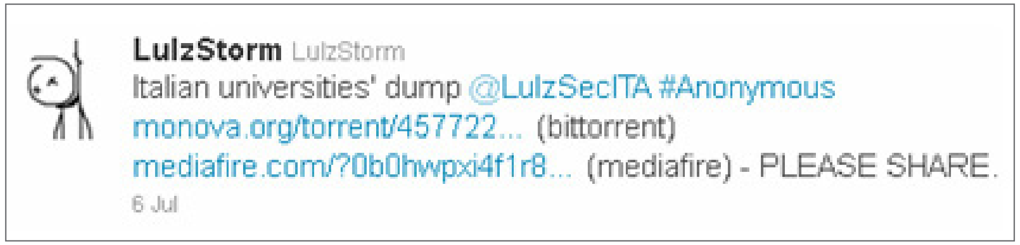
The summer months have been plagued by malware attacks of all kinds, while the infamous hacker group Anonymous has continued making headlines around the world. However, they haven't been alone, as cyber-crooks have continued with their criminal activities, especially those involving large-scale data breaches, and we have seen more acts of cyber-warfare.

Anonymous

The quarter started in the worst possible fashion for Anonymous as 15 alleged members of the cyber-crime organization were arrested in Italy. It is worth mentioning that all detainees ranged in ages from 15 to 28, with 8 minors. Police raided more than 30 households, seizing various materials (computers, etc.). The leader of the cell was a 26-year-old man living in Switzerland.

Soon after these arrests, we learned that Anonymous had broken into Universal Music's servers, stealing user data. This information included user names and passwords, and therefore Universal advised all users to change their login data. Events like this once again reveal that there are companies which do not take security seriously, as making the mistake of storing users' passwords in plain text is absolutely unforgivable. Having said this, let's not forget that the Anonymous collective acts like a group of vandals only looking to cause havoc; and what's worse, they usually end up damaging the very same users they claim to defend, as they post the information they steal, making it available to anybody who wants to use it for malicious purposes.

Also in July, a group of hackers stole and published data from the databases of 18 Italian universities.



LulzStorm LulzStorm
Italian universities' dump @LulzSecITA #Anonymous
monova.org/torrent/457722... (bittorrent)
mediafire.com/?0b0hwpxi4f1r8... (mediafire) - PLEASE SHARE.
6 Jul

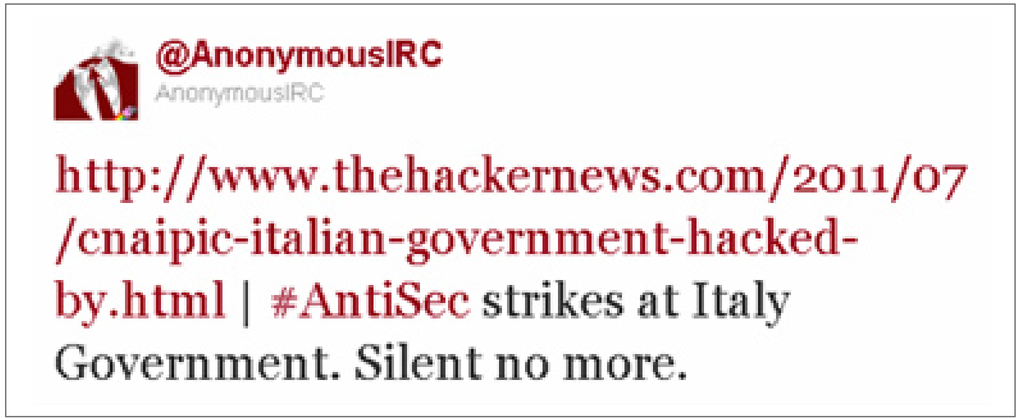
FIG.01. *TWEET ANNOUNCING THE HACK*

In the United States, Anonymous went one step further and hacked into the systems of Booz Allen Hamilton (a government contractor with strong ties to the US Department of Defense – DoD), stealing 90,000 military email addresses and passwords. They managed to enter the system through an outdated server with no antivirus protection at all.

Soon after these attacks the FBI arrested 16 Anonymous members in the US. All of these people could face 5 to 10 years in jail if found guilty.

However, none of these actions seem to have stopped Anonymous, who actually seems to have redoubled its efforts. Just days after the arrests, Anonymous posted links to two NATO confidential documents, and claimed to have one more gigabyte of confidential data which they refused to publish as it would be "irresponsible".

Meanwhile, Anonymous stroke once again in Europe, stealing over 8 gigabytes of data from Italy's CNAIPIC (National Center for Computer Crime and the Protection of Critical Infrastructure).

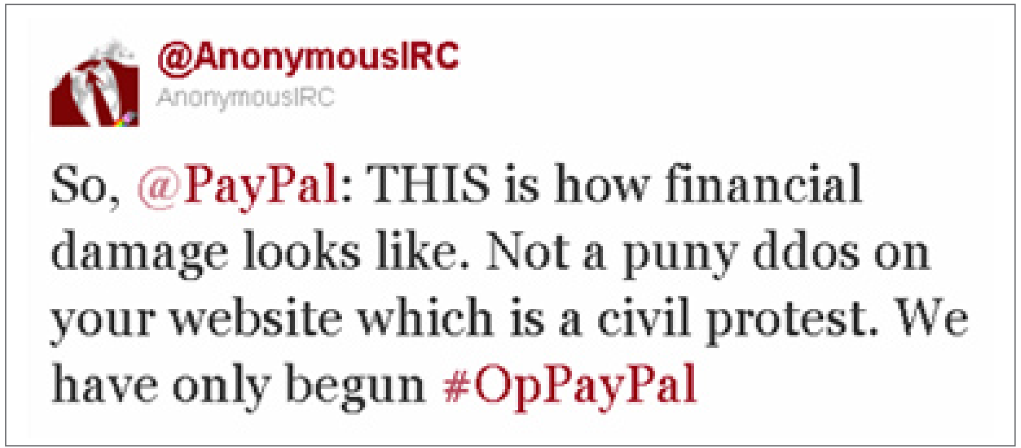


@AnonymousIRC
AnonymousIRC

http://www.thehackernews.com/2011/07/cnaipic-italian-government-hacked-by.html | #AntiSec strikes at Italy Government. Silent no more.

FIG.02. *MESSAGE POSTED BY ANONYMOUS, BOASTING OF THEIR LATEST ATTACK .*

After this string of cyber-attacks, some sense entered the minds of those behind Anonymous as they finally decided to do some real activism. The group dubbed this effort Operation #OpPayPal and encouraged supporters to close their PayPal accounts to boycott the company. Despite grabbing some headlines, the protest didn't have much of an impact.



@AnonymousIRC
AnonymousIRC

So, @PayPal: THIS is how financial damage looks like. Not a puny ddos on your website which is a civil protest. We have only begun #OpPayPal

FIG.03. *MESSAGE FROM ANONYMOUS ANNOUNCING ITS NEW CAMPAIGN.*

However, as the PayPal attack didn't turn out as expected, in August they decided to go back to what they do best: steal data. They released the stolen personal data of thousands of U.S. law enforcement officers, including their email addresses, user names, passwords and in some cases even their social security numbers. And they did it again a few weeks later, as they exposed personal data of San Francisco-area subway police officers. But, if this was not enough, the group hacked yet another U.S. Department of Defense contractor (this time Vanguard Defense Industries), stealing 1 gigabyte of data such as emails and confidential documents from one of the company's top executives.

Two more alleged members of Anonymous were arrested in the United States and the United Kingdom in September, but this does not seem to have deterred Anonymous from their activities.

Cybercrime

The security world has also had its share of good news, mainly the arrest of an increasing number of cyber-crooks. In July, Rogelio Hackett, 25, was sentenced to 10 years in prison and a $100,000 fine for stealing 675,000 credit card numbers and related information. The fact that there are tough sentences being handed out is very important as it sends out a strong dissuasive message to criminals: impunity is not as option.

Not everything has been positive, however, as the threat of cyber-crime continues to be present all over the world. Cyber-crooks continue to use social engineering techniques to deceive users and steal their data, taking advantage of headline-grabbing events such as the untimely death of singer Amy Winehouse or the Oslo massacre tragedy.

One of the key instruments in the fight against cyber-crime is international cooperation. Cyber-crime is transnational and requires a transnational response to tackle it. In this respect, the collaboration agreement signed between the United States' and India's Computer Emergency Response Teams (US-CERT and CERT-In respectively) is very important. The generalization of this type of agreement represents a major step forward in the fight against cyber-crime.

Cyberwar

In July, the US Deputy Defense Secretary Bill Lynn revealed that foreign intruders had taken 24,000 files of classified information about a top secret weapon system in an attack suffered in March this year. Lynn said that a "foreign intelligence service" was behind the theft of the secret weapon blueprints but declined to specify which nation had carried out the attack.

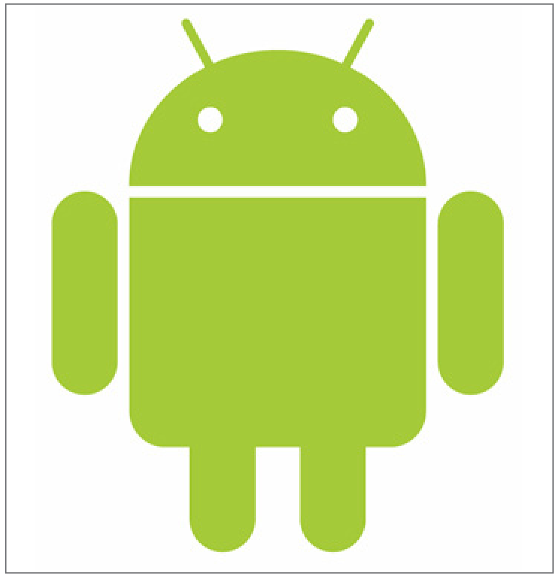
FIG.04. *24,000 PENTAGON FILES STOLEN IN MAJOR CYBER-BREACH.*

Just when most of us had forgotten about the Stuxnet worm, the DEBKAfile website published a report citing "intelligence sources" to claim that the Iranian government had had to replace an estimated 5,000 uranium-enriching centrifuges as a result of last year's attack, and that since then the country had not been able to return its uranium enrichment efforts to 'normal operation'. In fact, the foreign ministry of Iran acknowledged that they were installing "newer and faster" centrifuges to speed up the uranium enrichment process.

In July, the U.S. Department of Homeland Security said to the Congress that it was aware that a Stuxnet-like virus could be used to attack critical infrastructures in the country. Others have similar fears. Within DHS, many worry that other attackers could use 'increasingly public information' about the worm to launch variants that would target other industrial control systems.

If something can be said about cyber-war or cyber-espionage attacks is that most of them appear to originate from China. However, on one hand it is obvious that China is not behind every single attack and, on the other, China itself must be suffering attacks from others. When, for example, a country in the European Union suffers a computer attack, as has happened so many times this year, it becomes public knowledge. However, this is not the case in other countries. Is it that some countries are never attacked? Absolutely not, it is just that they do not make attacks known. And China, for once, has opened to the rest of the world and has admitted that it was hit by nearly 500,000 cyber-attacks last year, about half of which originated from foreign countries.

In September, we learned that Japanese company Mitsubishi Heavy Industries had also been hit by a cyber-attack. Almost 100 computers had been compromised, despite the company claiming that no confidential information had been stolen. This company builds highly critical equipment, like guided missiles, rocket engines and nuclear-power equipment. Chinese language was found in one of the viruses found in the cyber-attack, so once again all eyes turned to the Asian giant.



Macs, cell phones…

As previously explained in other reports, malware is becoming a very dangerous problem for Mac computers and cell phones' operating systems, especially Android. This quarter has been no different. We have seen a significant growth in the amount of malware for Mac computers, with increasingly sophisticated attacks that combine vulnerability exploitation and backdoor installation.

FIG.05. *ANDROID HAS BECOME A FAVORITE TARGET FOR CYBER-CROOKS.*

In July, Zitmo, a variant of the famed Zeus banking Trojan, hit the Android platform. If a user's cell phone was infected with the Trojan, the cyber-criminals could gain access to the victim's bank account and intercept the one-time transaction password sent by the bank to the user. This way, cyber-criminals could perform any kind of online transaction from the victim's account

If this was not enough, we learned that Android has some very basic security holes, as shown by the fact that it stores the passwords for email accounts on the phone's file system in plain text, with no encryption. This makes it an easy target for criminals, who can easily extract all passwords once they have hacked into the device.

The appearance of new Android malware is becoming increasingly frequent, and the final objective is always the same: to steal users' data. Finally, we have seen different variants of a new family of Android malware which not only copies data from the device and sends it to cyber-crooks, but also records phone conversations.

Social networks

The biggest news story in this area was the launch in June of Google+, as a direct competitor to Facebook. While Google+ is far simpler and less sophisticated than Facebook, it has nevertheless achieved millions of users in just three months.

Despite this, criminals have not targeted it as much as Facebook. However, right after its launch, as invitations were not open to everyone and there was huge expectation and interest in getting one, it became the subject of a scam… on Facebook. Fraudsters created a page titled "Get Google Plus Invitation FREE" where users just had to click the 'Like' button to get an invitation. Obviously, you also had to provide your email address to receive the invitation which, unfortunately, never came.

These scams are actually quite frequent on Facebook, cyber-crooks' favorite platform for launching social engineering attacks by exploiting real or fake news stories.

We cannot finish this section without mentioning Twitter which, although less exploited than Facebook, is also used by criminals to send spam and malicious links. One of cyber-crooks' favorite activities is account hacking. Fox News's Twitter account was hacked on July 4 and posted a series of alarming tweets reporting that U.S. President Barack Obama had been assassinated. The Twitter account of PayPal UK was also hacked and used to criticize its poor security in offensive language.

However, other attacks are far more serious. A group of attackers hacked the Twitter account of a financial institution and started sending Direct Messages ((DMs) to its followers instructing them to click on a link due to a security problem in their account. This link took users to a phishing page that imitated that of the bank and requested data that could then be used by attackers to impersonate the victims and steal their money.

03| Quarterly Figures

In this section we'll take a look at the Q3 malware statistics. Despite being the holiday period for many countries, malware writers have taken no rest days, generating more than 5,000,000 new malware strains this quarter. A closer look at the data reveals that, as usual since the popularization of computer crime, Trojans are the most prevalent type of malware. It is important to highlight that Trojans have accounted for 3 out of every 4 new malware strains created this quarter (76.76%), which sets a new record.

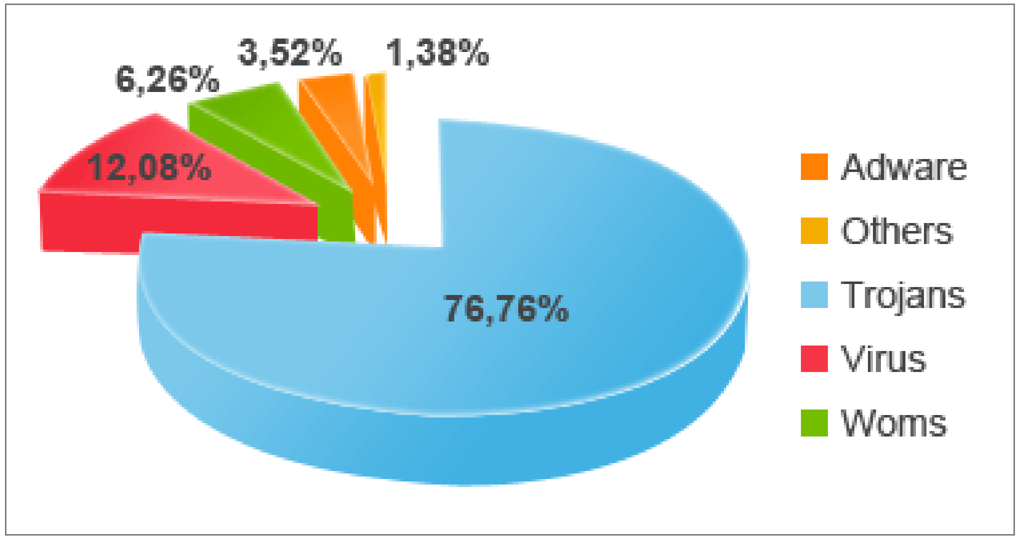


FIG.06. *NEW MALWARE SAMPLES DETECTED AT PANDALABS.*

Viruses came in second place (12.08%), followed by worms (6.26%) and adware (3.53%), a category which includes fake antivirus software and has increased significantly compared to last quarter.

In any event, these figures reflect the number and type of malware strains created, which does not always correlate directly with the number of infections, as a single malware strain can be responsible for many infections. Let's analyze the data collected by our Collective Intelligence sensor network in order to get a clearer picture of the global malware situation.

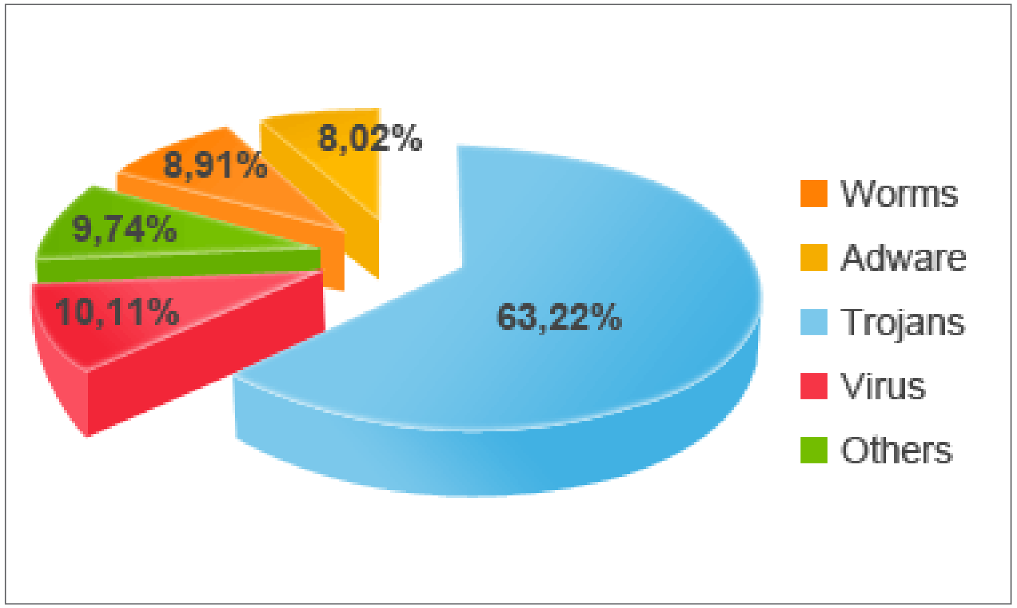


FIG.07. *INFECTIONS PER TYPE OF MALWARE.*

Trojans were once again the dominant malware category during this quarter, causing 63.22% of all infections. Surprisingly, despite there have been major changes in new malware development (Trojans have surged significantly), the data collected by PandaLabs during this period shows that the distribution of malware infections by type is mostly the same as last quarter.

As can be seen from the graph below, the Top 10 most prevalent malware specimens account for 49.97% of all infections. However, this can be a bit misleading as many of the entries on the list are generic detections (detected by Collective Intelligence) which include several malware families. The details are as follows:

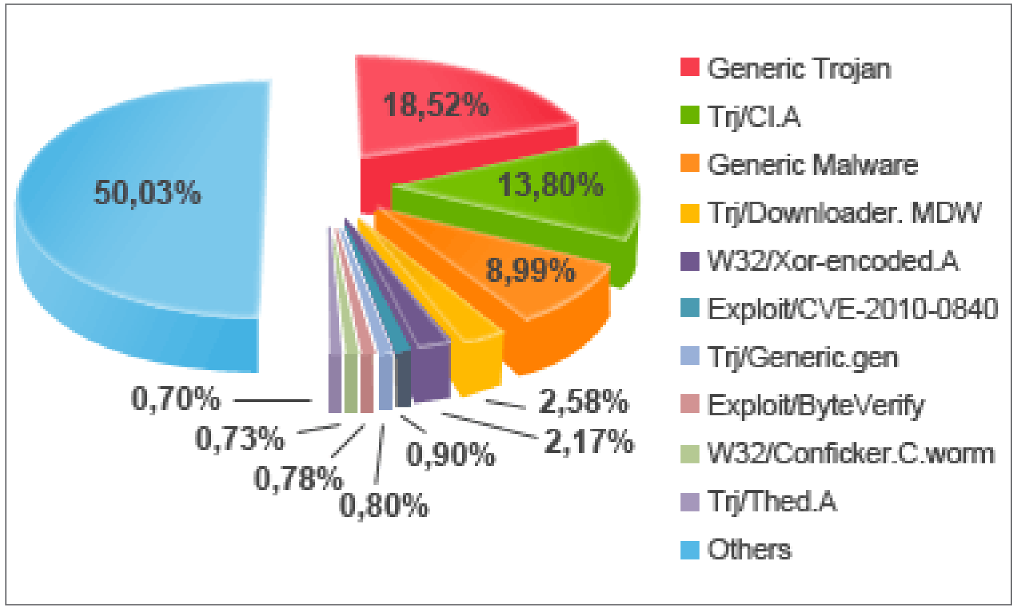


FIG.08. *MALWARE FAMILIES.*

Now we'll take a look at the infection rankings by country according to Collective Intelligence. The graph below shows the 20 countries with the highest rates of malware infection in Q3 2011:

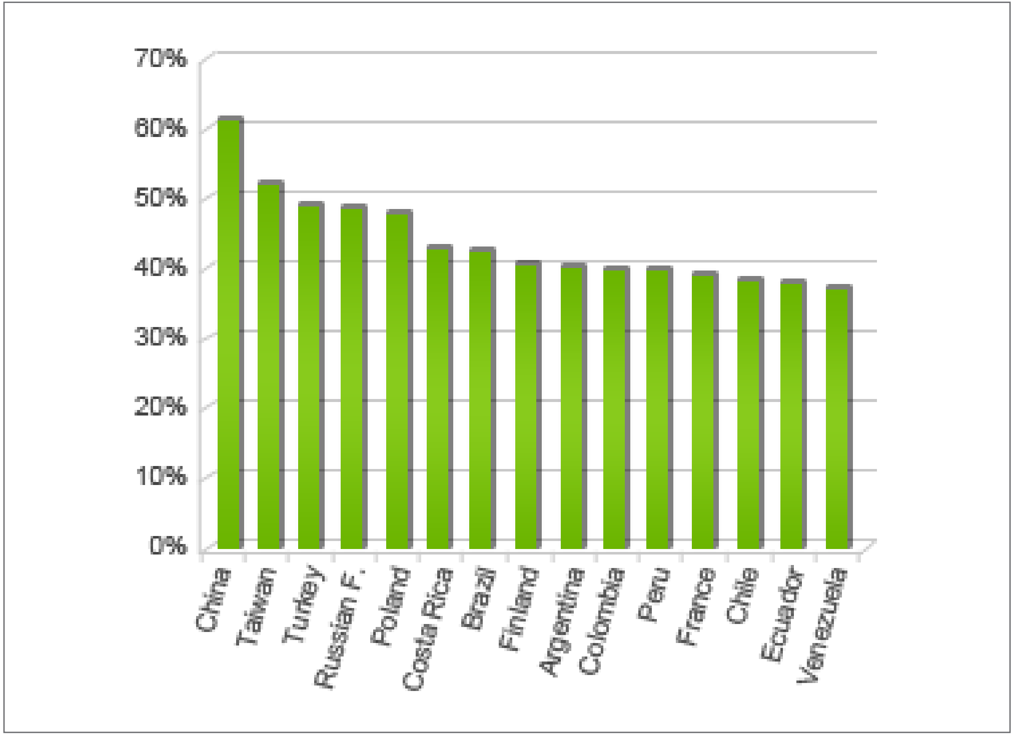


FIG.09. ***INFECTION RATE PER COUNTRY – TOP 20.***

The average infection ratio was 37.87%, 2 percentage points lower than in Q2. China once again had the most infected PCs, with a 62.47 % corruption, followed by Taiwan (50.93 %), Turkey (46.68 %) and Russia (45.73 %).

The country with the least infections is Sweden (23.36 %), followed by the United Kingdom (26.53 %), Switzerland (26.57 %) and Germany (28.20 %). The graph below shows the countries with the lowest infection rates. It is interesting to note that all of them are European with the exception of Japan and Australia:

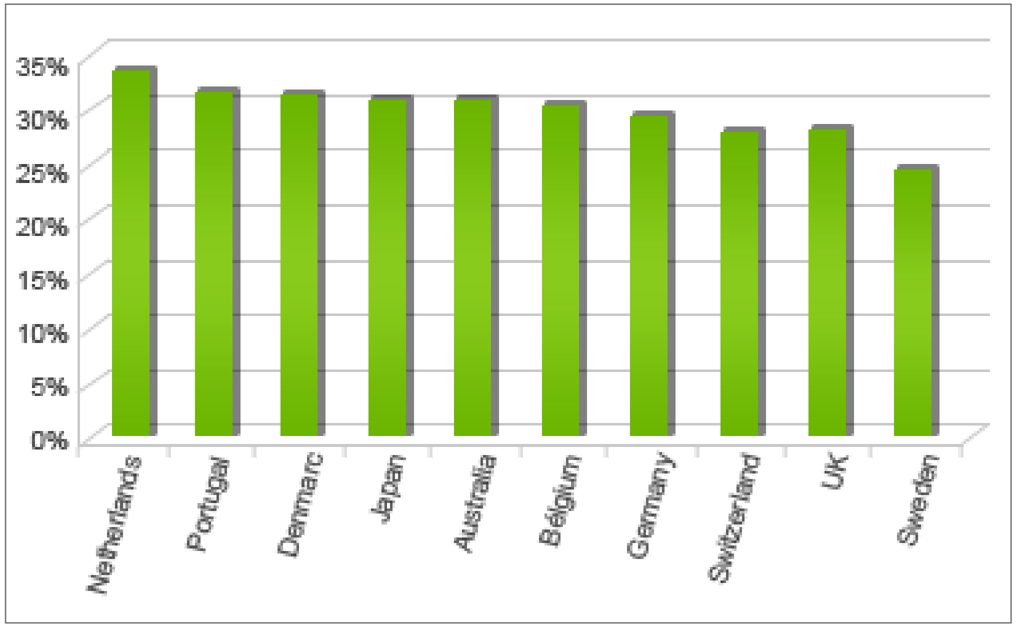


FIG.10. ***COUNTRIES WITH THE LOWEST INFECTION RATE IN THE WORLD.***

04| How Do Exploits Work?

Over the last few quarters, the vast majority of articles in this section have dealt with the various vulnerabilities that have been discovered and their impact.

However, the last couple of articles introduced a change in subject as we decided to discuss the Pown2Own hacking competition and the latest Web browsers' security level. We explained how Google Chrome left unharmed from the contest despite hackers' best efforts to break it. It seemed that Google Chrome was the safest browser in the world but, was that really true? The second quarter of the year brought the answer to that question: after two months of hard work, researchers from Vupen Security managed to hack it, confirming what hackers have always said: "100% security is impossible".

This does not mean that we should stop using such an important tool as the Internet. But it does indicate that we must do all that we can to make it difficult for malware to perform malicious actions. On many occasions we have advised users to keep their operating system, antivirus and any other security software always up-to-date. Obviously, an athlete will find it more difficult to run a 100 meter hurdle race than a regular 100 meter race. It is our job as security professionals to create those 'hurdles' or obstacles that hinder malicious actions. And users must also understand that those 'hurdles' are necessary to protect their resources.

This article explains the reasons why software vulnerabilities exist and how they are exploited by malicious code (from a technical point of view).

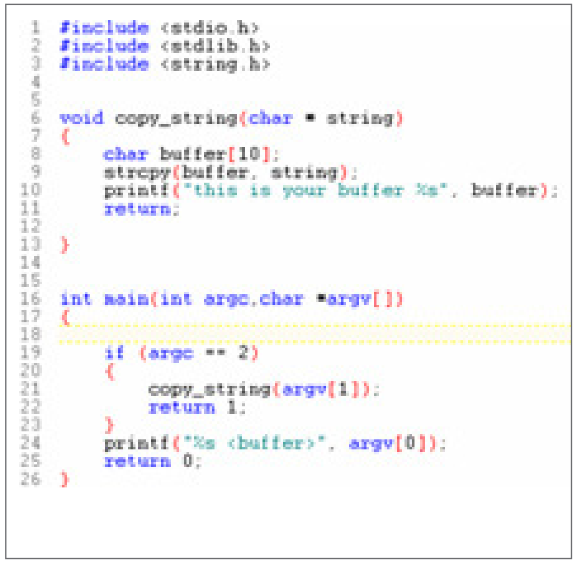
This is the first of a series of articles about the state-of-the-art in software vulnerabilities and their exploitation as well as the capacity of current protection systems to neutralize them. Let's start with the basics: why do computer security vulnerabilities exist?

What is a security vulnerability?

According to Wikipedia, in computer security, a vulnerability is:

A weakness in a system allowing an attacker to violate the confidentiality, integrity, availability, access control, consistency or audit mechanisms of the system or the data and applications it hosts. Some of the most serious vulnerabilities allow the execution of arbitrary code.

In short: Vulnerabilities are the result of insecure programming techniques, and therefore they will always exist. Consequently, a user could take advantage of a software flaw to carry out actions the software was not designed for.



```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <string.h>
4
5
6   void copy_string(char * string)
7   {
8       char buffer[10];
9       strcpy(buffer, string);
10      printf("this is your buffer %s", buffer);
11      return;
12
13  }
14
15
16  int main(int argc,char *argv[])
17  {
18
19      if (argc == 2)
20      {
21          copy_string(argv[1]);
22          return 1;
23      }
24      printf("%s <buffer>", argv[0]);
25      return 0;
26  }
```

Now, the next question would be: How can a user perform actions (that is, run code) the application was not designed for? In order to answer that question we must first explain how the operating system runs an application's code.

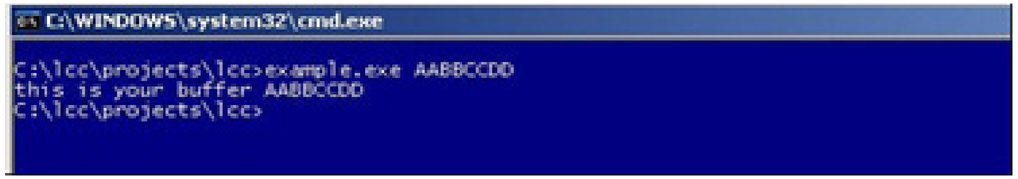
We'll start by programming the following application using the C1 programming language.

FIG.11. *IMAGE 1.*

NOTE: This code has been compiled with the LCC –Win32 compiler:
http://www.cs.virginia.edu/~lcc-win32/

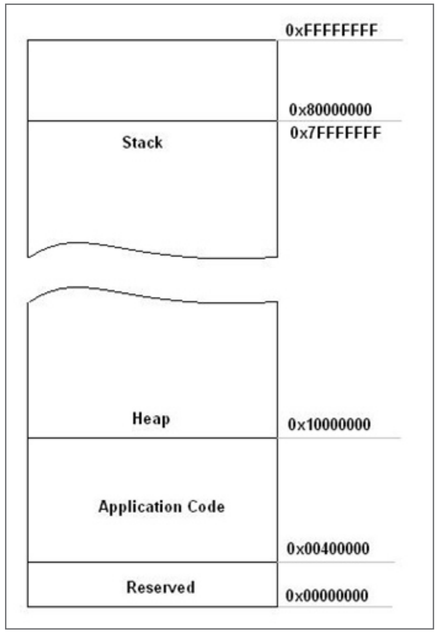
As you can see, this program is very simple. It is designed to take the first parameter entered in the console by the user (line 21), copy the value to the ***buffer*** variable (line 9) and display the variable's content to the user (line 10).

For example, once the program has been compiled, if you run it and pass the AABBCCDD string as a parameter, it will return the following expected result. That is, it carries out the function it has been designed for.



```
C:\lcc\projects\lcc>example.exe AABBCCDD
this is your buffer AABBCCDD
C:\lcc\projects\lcc>
```

FIG.12. *IMAGEN 2.*



What has happened? The program has been loaded into memory by the operating system which, among other things, creates a ***stack***2, a ***heap***3 data structure and a series of other operations for the computer's processor or ***CPU***4 to run the program's code.

The figure below shows the basic layout of the program in memory.

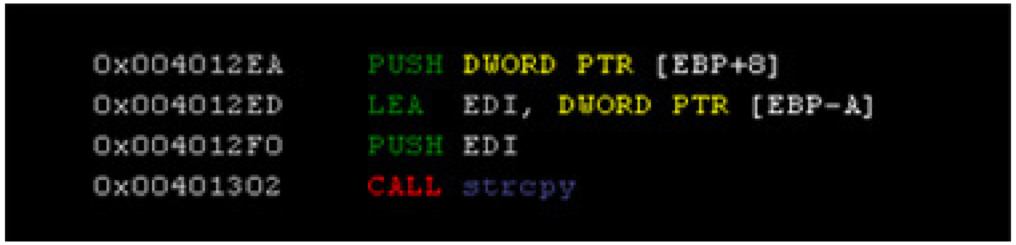
FIG.13. *IMAGE 3.*

The processor ***registers***, the ***stack*** and the ***heap*** [5] are the key elements a security researcher must take into account to develop an exploit. But there are other elements that must be considered to exploit a vulnerability successfully. Some of them are:

- Operating system version
- Operating system protection (ASLR[6], DEP[7])
- Compatibility between the application and the operating system protection
- Protection implemented by the compiler. (/GS[8])
- Protection implemented by the developer.

Vulnerability exploitation is a very wide topic discussed in many books and online articles. Bear in mind that this phenomenon dates back to the late 80's and things have changed significantly since then. Nevertheless, the aim of this first article is to explain the basics of how to run arbitrary code via an application flaw. The next articles will present a more technical explanation of the current situation of vulnerability exploitation and mitigation.

Let's start at the beginning. Once a process is created in memory for an application, and when the operating system deems appropriate, the processor will run each of the instructions that make up the program right to the end. The EIP register holds the current instruction being run and stores the pointer to the next instruction to be run.



```
0x004012EA      PUSH DWORD PTR [EBP+8]
0x004012ED      LEA  EDI, DWORD PTR [EBP-A]
0x004012F0      PUSH EDI
0x00401302      CALL strcpy
```

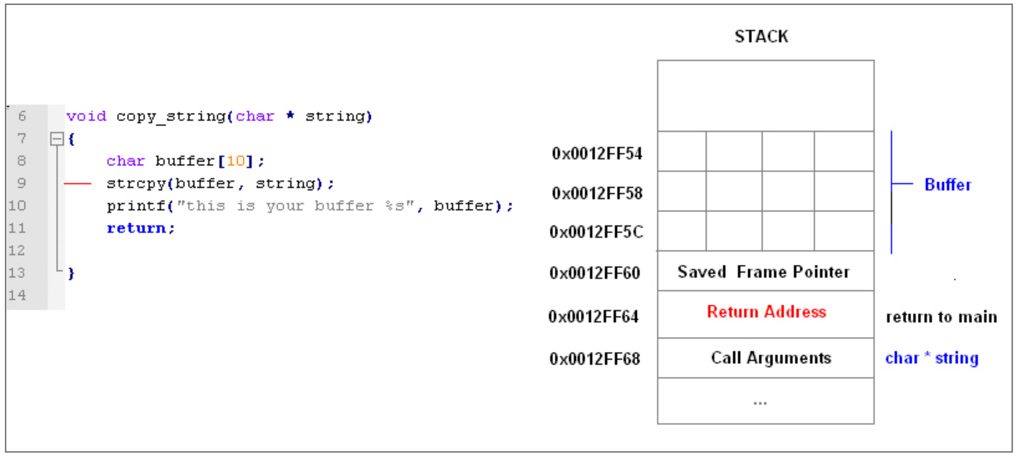
FIG.14. *IMAGE 4.*

That is, as the processor runs all the instructions, the ***EIP*** register will take the following values: *0x004012EA*, *0x004012ED* and *0x004012F0*, *0x00401302*. For example, when the ***EIP*** register takes the value *0x004012F0*, the processor will run the instruction ***"PUSH EDI"***.

Bearing this in mind, the purpose of our ***exploit***[9] will be to modify the content pointed to by one of the addresses used by the EIP register. This way, we'll make the ***EIP*** register point to our ***shellcode***[10], and we'll manage to run arbitrary code.

Right now you are probably thinking: How can you modify the content that the ***EIP*** register points to? Well, to answer that question it is important to fully understand how processor registers work and how program variables are stored on the stack during program execution, among other things. Let's analyze what happens on the stack prior to the call to the ***strcpy*** function (line 10) in ***copy_string*** (line 21).

The processor runs line 9 in the program. It calls the ***strcpy*** function, passing the ***buffer*** (character array) and ***string*** variables as parameters to it. Before the function is called, the stack (see figure 5) will look like this:

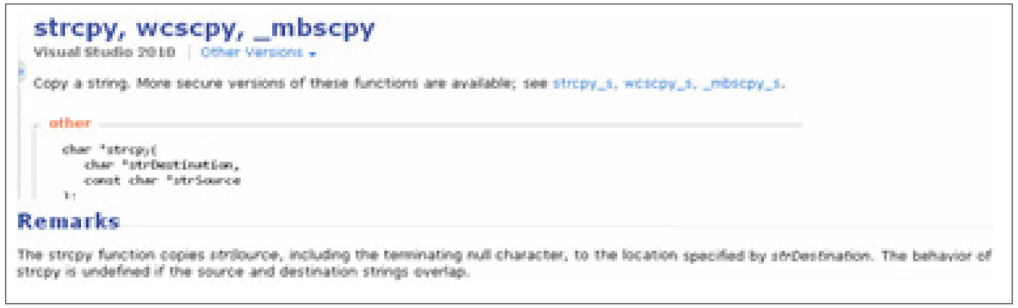


```
6   void copy_string(char * string)
7   {
8       char buffer[10];
9       strcpy(buffer, string);
10      printf("this is your buffer %s", buffer);
11      return;
12
13  }
14
```

FIG.15. *IMAGE 5.*

The first piece of data stored in the stack -at 0x0012FF68- is the address of the ***string*** parameter passed to the ***copy_string*** function (lines 6 and 21). This refers to the parameter entered by the user in the program. The second stored parameter is ***Return Address***. This is the address used by the ***EIP*** register to return to the ***main*** function. It is actually the memory address that points to the next instruction (line 22 in the code) in ***main*** once the execution has returned from the call to ***copy_string***.

The next instruction to be run is line 9, where the ***buffer*** and ***string*** arguments are passed to the ***strcpy*** function. Microsoft defines this function as:



strcpy, wcscpy, _mbscpy

Visual Studio 2010 | Other Versions

Copy a string. More secure versions of these functions are available; see strcpy_s, wcscpy_s, _mbscpy_s.

other

```
char *strcpy(
   char *strDestination,
   const char *strSource
);
```

Remarks

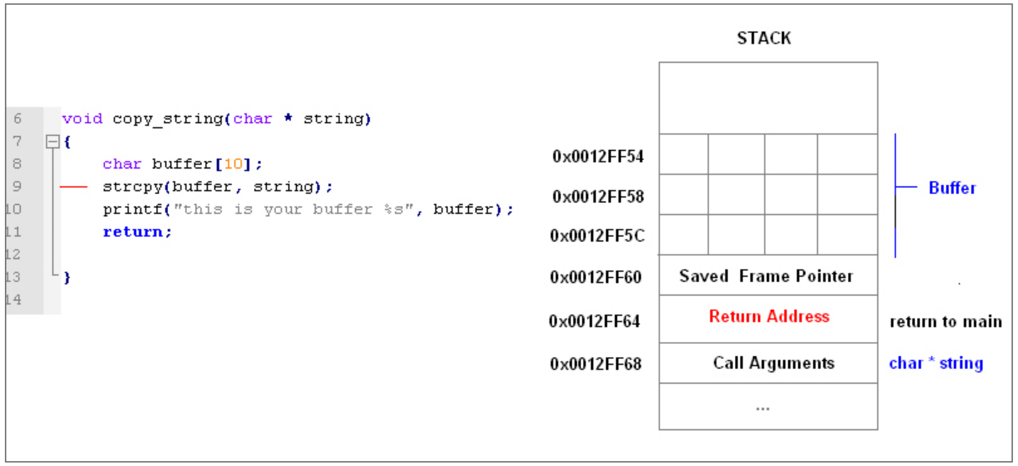
The strcpy function copies strSource, including the terminating null character, to the location specified by strDestination. The behavior of strcpy is undefined if the source and destination strings overlap.

FIG.16. *IMAGE 6.*

This function works in a very simple way: it copies the content of the source string to the location specified by the destination string. The behavior of ***strcpy*** is undefined if the source and destination strings overlap. How can that be? ***Strcpy*** is a function that copies strings, and it is unsafe as it doesn't check if the destination string has enough space to store the content of the source string. Should this happen, a memory area allocated to other data would be overwritten. This might cause the program to behave unexpectedly, as there would be incorrect data in the memory address whose content has been modified.

Let's see how the application behaves when the string entered is smaller than 10 bytes (the capacity of the ***buffer*** variable, line 8).

Let's run the following command: ***example AAAAA.***



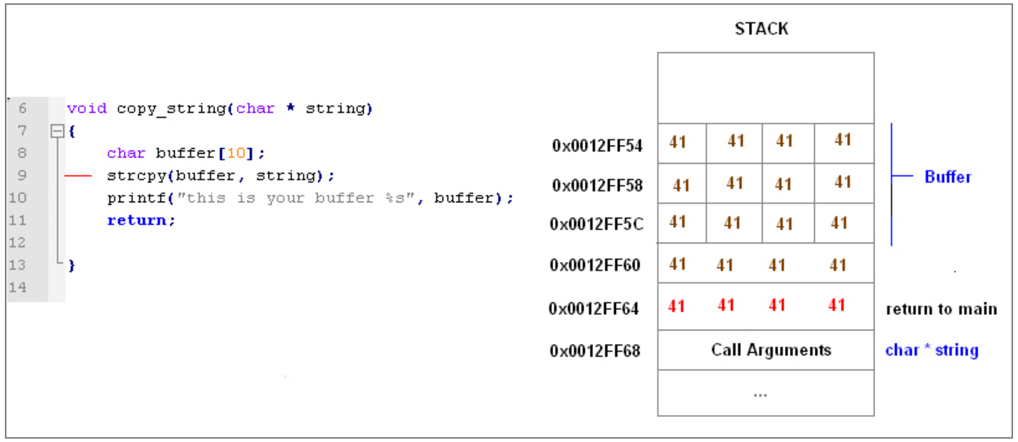
```
6   void copy_string(char * string)
7   {
8       char buffer[10];
9       strcpy(buffer, string);
10      printf("this is your buffer %s", buffer);
11      return;
12
13  }
14
```

FIG.17. *IMAGE 7.*

As seen in the figure above, 6 out of the10 "cells" (bytes) included in the ***buffer*** variable get filled (the figure shows the number 41, which is the hexadecimal value of the A character, and the null character 00 indicating the end of the string). That is, given the way the application has been programmed and the use of the unsafe ***strcpy*** function, there will be as many cells occupied in the stack memory as the length of the string passed as a parameter, regardless of the length of the ***buffer*** variable.

With this in mind, can you think of a way to change the application flow knowing that ***strcpy*** is an unsafe function? No? C'mon, sure you can. You simply have to send a long enough string to occupy as many 'cells' as possible in the ***stack*** memory. This will modify the value stored in the 0x0012FF64 address, which represents the ***Return Address*** value and is run by the processor's ***EIP*** register on return to the ***main*** function.

According to this, the stack should look like this if you passed a 20-character string to the application.

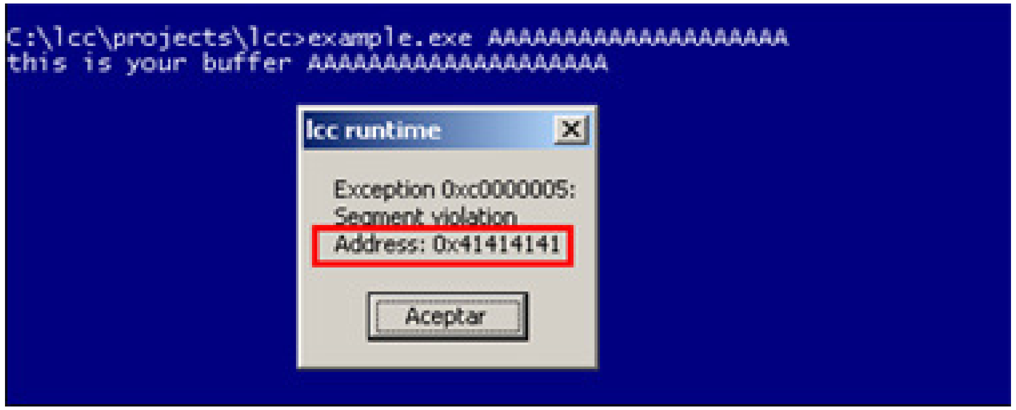


```
void copy_string(char * string)
{
    char buffer[10];
    strcpy(buffer, string);
    printf("this is your buffer %s", buffer);
    return;

}
```

FIG.18. *IMAGE 8.*

Let's see it!!
Run the following command: ***example AAAAAAAAAAAAAAAAAAA***A



```
C:\lcc\projects\lcc>example.exe AAAAAAAAAAAAAAAAAAAA
this is your buffer AAAAAAAAAAAAAAAAAAAA
```

```
lcc runtime

Exception 0xc0000005:
Segment violation
Address: 0x41414141

Aceptar
```

FIG.19. *IMAGE 9.*

Perfect!! We did it: The ***0x41414141*** address has been run and an exception has occurred as it is not a valid address for the process.

This is known as a DoS[11] or denial of service vulnerability, and is aimed at stopping an application service. This type of vulnerability can have serious consequences. Imagine what would happen if a bank's Internet servers suddenly stoped working: every minute of downtime could mean millions of dollars in lost revenue.

At the beginning of this article, we mentioned that the purpose of our exploit would be to allow arbitrary code execution. The question now is: How? The answer should be simple at this point: Instead of using the invalid ***0x41414141*** address, the attacker would have to use a valid memory address that pointed to the shellcode that they want to run. But, how do you inject the shellcode into the program memory? Is it possible to run arbitrary code in all vulnerabilities? We'll leave you to do your own research on these questions and we will answer them in subsequent articles.

For more information on creating shellcode and generating exploits, we recommend that you read ***The Shellcode Handbook*** as well as our next articles.

1 http://en.wikipedia.org/wiki/C_%28programming_language%29
2 http://en.wikipedia.org/wiki/Stack_%28abstract_data_type%29
3 http://en.wikipedia.org/wiki/Heap_%28data_structure%29
4 http://en.wikipedia.org/wiki/Central_processing_unit
5 http://en.wikipedia.org/wiki/Processor_register
6 http://en.wikipedia.org/wiki/ASLR
7 http://en.wikipedia.org/wiki/Data_Execution_Prevention
8 http://en.wikipedia.org/wiki/Stack-smashing_protection#Microsoft_Visual_Studio_.2FGS
9 http://en.wikipedia.org/wiki/Exploit_%28computer_security%29
10 http://en.wikipedia.org/wiki/Shellcode
11 http://en.wikipedia.org/wiki/Denial-of-service_attack

05| Conclusion

We live in an increasingly interconnected world where Internet use and social networking (Facebook, Twitter, etc) is no longer restricted to your home or office computer; mobile devices offer multiple options to stay connected to your digital world.

This not only benefits users, but also opens a window of opportunity for cyber-criminals who, as you have seen, don't miss any chance to profit from their creations. As cyber-crime becomes an increasingly organized, professional activity, criminals are always looking for new ways to optimize their work, which mainly consists of stealing user data.

Finally, the concept of cyber-war is alive and well and becoming ever more present. It is now clear that cases like Stuxnet are not an isolated incident but represent just the tip of the iceberg. It seems that we are right now in the early years of a cyber-war arms race much like in the Cold War, although this time the battlefield is the Internet.

And this is everything that has happened in the computer security sector in Q3 2011. Our next report will give you a summary of 2011 as well as information on the new trends for next year.

06| About PandaLabs

PandaLabs is Panda Security's anti-malware laboratory, and represents the company's nerve center for malware treatment:

- **PandaLabs** creates continually and in real-time the counter-measures necessary to protect Panda Security clients from all kind of malicious code on a global level.

- **PandaLabs** is in this way responsible for carrying out detailed scans of all kinds of malware, with the aim of improving the protection offered to Panda Security clients, as well as keeping the general public informed.

Likewise, PandaLabs maintains a constant state of vigilance, closely observing the various trends and developments taking place in the field of malware and security. Its aim is to warn and provide alerts on imminent dangers and threats, as well as to forecast future events.

- For further information about the last threats discovered, consult the PandaLabs blog at: **http://pandalabs.pandasecurity.com/**