The development and application of new and innovative data hiding and steganography weapons by criminals and worse is clearly on the rise. We added over 140 new data hiding and steganography tools to our cyber weapons repository during 2011. In addition to the increase in the number of new applications, we observed a dramatic increase in the sophistication, depth and support for a wider array of computing platforms such as smart mobile devices.

**2011 Quantitative Results**

![Image of new steganography applications in 2011](image-url)
Threat Evolution

Advanced Persistent Threat (APT)

We recognize that data hiding and steganography are beginning to play an important role in the command, control and communication between deployed threats and their operators. Our ability, as we move forward, to detect and mitigate such threats is paramount.

Let’s take a look at a couple of these threats and their impact.

Operation Shady Rat:

*Operation Shady RAT* was a well planned and executed advanced persistent threat (APT) that has been ongoing for at least 5 years. The attack impacted over 70 organizations including corporation, government agencies, and non-profits in as many as 14 countries. The targeted organizations were infiltrated by the malicious code as most malware is deployed today, but the goal was to keep the breach hidden and slowly exfiltrate information from the infected locations. This was facilitated through the use of innocuous digital images that contained command and control instructions and additional malware components. This use of steganography has been long proposed and now proven to be a viable element of sophisticated attacks (whether considered APT or not).

Alureon

According to sources at Microsoft and others, the Alureon Trojan is part of a new genre of malware categorized as data stealing. What makes Alureon interesting is the use of data hidden in jpeg files that were distributed across the internet at innocuous locations. The trojan would reach out to those images via the web, and decode the hidden contents. The images contain information that is interpreted by the trojanized *com32* software,
allowing Alureon to obtain a list of trusted command and control servers in order to continue and expand operations.

**Aleuron Trojan**

1. Downloader portion of the trojan downloads a component named 'com32'.

2. The downloaded file contains an encrypted list of URLs to a number of free blogging websites.

3. The websites contained a number of innocent looking jpeg images hosted from the website imagestock.us.

4. The code in Aleuron parses specific images from the various webpages. The raw data of which, along with a 61 character ASCII string appearing to be a password, are passed back to the ‘com32’ component.

5. Using the raw jpeg data, lists of IP addresses for command and control centers are extracted.

**Microsoft Malware Protection Center**

**Figure 3 - Alureon Trojan CONOPS**
As the explosion of smart mobile devices continues to expand along with the applications available for Android, iPhone and Windows Mobile, new data hiding applications have emerged. The table below itemizes just some of the new steganography offerings now available on smart mobile platforms. There are a couple of surprises and innovative techniques. On the other hand, as we expected, some of the methods are very simple and not stealthy, while others provide a window into the innovation we are likely to see continue.

<table>
<thead>
<tr>
<th>APP Name</th>
<th>Author</th>
<th>Platform</th>
<th>Basics</th>
<th>File Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>StegSec</td>
<td>Raffaele De Lorenzo</td>
<td>iPhone</td>
<td>Simple text hiding. Information is hidden in the comment field of the jpeg header.</td>
<td>.jpg</td>
</tr>
<tr>
<td>iStego</td>
<td>Antonio Calatrava</td>
<td>iPhone</td>
<td>Allows you to hide text or an image into a cover image. The hidden data is stored in the IDAT image data using LSB methods.</td>
<td>.png</td>
</tr>
<tr>
<td>Spy Pix</td>
<td>JuicyBitsSoftware</td>
<td>iPhone</td>
<td>Hides a secret inside a selected cover image. Interesting approach that required a special detection approach to uncover.</td>
<td>.png</td>
</tr>
<tr>
<td>Concealment</td>
<td>David Berroa</td>
<td>iPhone</td>
<td>Hides an image inside an image by appending an additional IEND marker to the .png.</td>
<td>.png</td>
</tr>
<tr>
<td>Pixogram</td>
<td>UnderWare LLC</td>
<td>iPhone</td>
<td>Hides text within a selected cover image. Text data is compressed, encrypted and embedded in the last</td>
<td>.png</td>
</tr>
<tr>
<td>Application</td>
<td>Developer</td>
<td>Operating System</td>
<td>Description</td>
<td>File Format(s)</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------</td>
<td>--------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>PrivateTIP</td>
<td>ADJ-soft</td>
<td>iPhone</td>
<td>Hides text with a cover image. Hidden text can be encrypted. Ultimately stored in the JPEG scan data.</td>
<td>.jpg</td>
</tr>
<tr>
<td>Hide it In</td>
<td>Jorge Blasco Alis</td>
<td>iPhone</td>
<td>Program crashes on iPad.</td>
<td></td>
</tr>
<tr>
<td>CoverText</td>
<td>J &amp; R Technologies, Rabah Rahil</td>
<td>iPhone</td>
<td>Hides text inside cover image. Data is hidden JPEG header comment field.</td>
<td>.jpg</td>
</tr>
<tr>
<td>InvisiLetter</td>
<td>Samurai-apps</td>
<td>iPhone</td>
<td>Allows you to draw (with finger or stylus) a message or drawing on an image. The data is embedded in IDAT chunks by modifying the LSB values.</td>
<td>.png</td>
</tr>
<tr>
<td>Secret Letter</td>
<td>ivanaLum</td>
<td>Android 2.1+</td>
<td>Hides Text into images, accepts images from camera or gallery, and utilizes a password to encrypt message.</td>
<td>.png</td>
</tr>
<tr>
<td>Da Vinci Secret Image</td>
<td>RadJab</td>
<td>Android 2.1+</td>
<td>Hides text into images. Accepts an image from gallery. Allows for an optional password and selecting different sizes for the image.</td>
<td>.png</td>
</tr>
<tr>
<td>My Secret</td>
<td>Tipspedia Ro</td>
<td>Android 1.6+</td>
<td>Hides text into images. Selects an image from the gallery. Only supports png/jpg and requires an SD Card to run the app</td>
<td>.jpg/.png</td>
</tr>
<tr>
<td>Stega</td>
<td>Danny Thuering</td>
<td>Android 2.1+</td>
<td>Hides text into images, accepts images from the gallery.</td>
<td>.jpg</td>
</tr>
<tr>
<td>Application</td>
<td>Developer</td>
<td>Android Version</td>
<td>Description</td>
<td>File Extension</td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
<td>-----------------</td>
<td>-------------</td>
<td>---------------</td>
</tr>
<tr>
<td>StegDroid</td>
<td>Tom Medley</td>
<td>2.1+</td>
<td>Hides text into audio. Accepts both recorded audio and direct microphone input.</td>
<td>.ogg</td>
</tr>
<tr>
<td>MobiStego</td>
<td>Pasquale Paola</td>
<td>2.0+</td>
<td>Hides text into images by using LSB embedding within the IDAT Chunks.</td>
<td>.png</td>
</tr>
</tbody>
</table>

In 2011 we saw many of these new threats evolve and allow for covert communication using image and multimedia file interchange. As this first wave of apps advances, we expect to see improvements in usability and improvements in the core algorithms. However, several of the algorithms, above, that hide small amounts of text into large images or multimedia carriers provide a viable means of covert communications due to the difficulty in detecting statistical variations with such tiny payload to carrier file ratios.
Multimedia Steganography

With the benefits we all experience from the increased exchange and streaming of multimedia, there is also a downside. These same streams and multimedia files offer the opportunity to hide and exchange much larger amounts of information. A couple of notable offerings in this category include OpenPuff and MSU Stego.

OpenPuff Steganography is a free tool for the Windows Operating Systems. OpenPuff is semi-open source in that the encryption algorithm is open source, but the rest of the program is proprietary. The way OpenPuff works is that the data to be hidden is split up and then hidden inside many carrier files using a variety of embedding methods. The program allows users to hide data in a plethora of carrier types ranging from image, audio, and video files. Before the data is hidden, it is encrypted, whitened, then encoded (whitening is a decorrelation method). The advantage of hiding encrypted data into carrier files using stego is that not only are you hiding data, but you are hiding the fact that you have hidden data. This makes it much easier to pass data back and forth without arousing suspicion. In short, the use of
steganography and cryptography together protects both the hidden data as well as the people using the files to communicate. A notable feature of OpenPuff is the decoy file. If someone finds your file and demands the password from you, you can give a decoy password that will allow you to successfully extract a file that would not include incriminating data.

MSUStego

MSU StegoVideo is a free non-open source steganography program available from Moscow State University in Russia. Its key features are an ability to hide text-based data efficiently into a video sequence with very low distortion, and an ability to extract that data with a small amount of errors. This program also allows you to protect the data being hidden into the image using a passcode. MSU StegoVideo is unique in the fact that it is the only known product that can hide data in the actual frames of the video versus simply adding data to redundant parts of the video file.

Though the algorithm that MSU StegoVideo uses is not yet known, there are some things that we do know about it. For one, MSU StegoVideo stores information into the video by modifying the video frame by frame. This is done by decomposing each frame
into bit planes and modifying the lower (least significant) planes. Modifications in these lower planes, in most cases, have no visual effect on the frame. We also know that the method that MSU StegoVideo uses to prevent data loss is redundant storage of the data throughout the video. It stores all data at least twice in the video; however, the level of redundancy can be chosen by the user. Overall, MSU StegoVideo could be considered dangerous because without some in-depth analysis of a video file, the fact that data was hidden inside of it would be completely indistinguishable.
Streaming Methods

As we briefly discussed above, it is possible to insert hidden information within data streams not only within multimedia or digital image carriers. One of the methods that continues to evolve, and is an ongoing research project, involves the use of StegoSIP and Lost Audio Packet Steganography. The diagram below depicts the use of this process. The steps involved are pretty straight-forward and demonstrates the weakness that exists today in most network protocols.

1. StegoSIP is launched on both Alice and Bob’s computer and listens for activity.
2. Next a Voice over IP session is initiated between Alice and Bob and a standard voice conversation ensues.
3. The Lost Audio Packet Stego method is then employed. This method delays selected RTP packets from delivery. The VOIP program on the receiving end automatically compensates for the delayed or missing packet and voice communication continues normally.
4. The delayed packets are then modified by embedding the hidden data in the desired payload of theses delayed packets.
5. The delayed packet with the hidden content is reinserted into the data stream.
6. The excessively delayed packet is detected and then intercepted by StegoSIP and the hidden content is decoded. The packet is allowed to continue on as normal to the VOIP program, but due to the delay in the delivery the VOIP program ignores the packet as it was already previously compensated for.
To avoid the risk of detection, the user employing the lost audio packet method must randomly select packets and ensure that the frequency of the process is such that it creates a low level observable.
One of the newest Data Hiding weapons is a technology that combines the power of TrueCrypt® (one of the best known and easiest to use encryption programs) with a steganography twist. This latest advancement hides a TrueCrypt container inside an existing MP4 or QuickTime multimedia file.

TCStego is a straight-forward python script that works with both QuickTime and MP4 multimedia containers.

The tcsteg.py application combines an existing MP4 or QuickTime multimedia file with a TrueCrypt file container in such a manner that the resulting file operates as both a standard multimedia file and as a mountable TrueCrypt volume.

With the rapid increase in movie files exchanged over the Internet (YouTube, etc.), a huge haystack in which to hide or exchange covert information exists right now, and the size of this haystack is predicted to increase exponentially over the next decade. Clearly this provides a new method for pedophiles to
exchange their content through innocuous sharing of benign looking digital media, and criminals or worse to continually exchange large amounts of clandestine information.
Mitigation Strategies

Over the past decade we have developed a comprehensive set of mitigation strategies and technologies to assist government, law enforcement and commercial entities deal with the escalating data hiding and steganography threat.

1. Awareness and Training
2. Detection, mitigation and blocking of unauthorized Data Hiding and Steganography Programs
3. Discovery of images, multimedia files and datastreams that have been utilized as a carrier of hidden content
4. Recovery of hidden information once carriers have been detected
5. Data Leak Protection
6. Nondestructive jamming of images, multimedia files and network protocols to prevent leakage of sensitive data

If you would like more information regarding our product, training or services offerings please feel free to contact us.
Future Predictions

New data hiding and steganography methods are likely to lock down our networks and end points by employing network intrusion prevention solutions, content filters, application firewalls, data leak prevention solutions and host intrusion prevention capabilities, criminals, malicious insiders and worse will continue to exploit weaknesses in protocols, images, multimedia and general data structures. These exploits will provide mechanisms to conceal and exfiltrate data and provide a viable means to command and control the deployed malicious code.

Without significant and immediate measures to uncover, mitigate or otherwise disrupt these operations we risk further attacks and continued loss of intellectual property or state secrets.