Keeping Secret Secrets Secret and Sharing Secret Secrets Secretly

Vic Vandal
vvandal@well.com

DefCon-XVI / 2008

DARK SECRETS
(JUST BETWEEN YOU AND ME)
SECRETS ARE LIKE CHOCOLATE: BEST DARK, AND ONLY SHARED WITH SOMEONE THAT YOU CAN TRUST

Vic’s Usual Disclaimer

- Don’t do anything illegal.
- If you do anything illegal, don’t get caught.
- I take no personal responsibility for the subsequent illegal use of any information provided here by anyone.
- I don’t condone/support espionage, treason, child porn, terrorism, or general stupidity. I “may” take more liberal views on some of the other usage examples herein.
- I own my own words, which means I may legally challenge any use or duplication of them for which I have not provided explicit permission.
- Any duplication of this presentation must include this slide.

Stego Files Link

(for later in the presentation)

http://www.well.com/~sthomas/.Secret-Page.html

Introduction

- What is a secret?
  - DUH!…anything secret, keys to important items, secret plans/designs/code, sensitive info/items, etc.
- Why protect secrets?
  - Because “SECRET = SECRET” (see Webster’s definition for more on that secret formula, cough)
- Why share secrets?
  - Secrecy
  - Fault-tolerance
  - Profit
  - Fun

Chaffing and Winnowing

What Is It?

- *Winnowing* -- separating out or eliminating the chaff
- *Chaffing* -- adding chaff to a collection
- By a sender strategically adding chaff to a message, it becomes garbled/hidden
- By the receiver correctly winnowing out the chaff, they can retrieve the original message

How It’s Done

1. Authenticate each packet of message with a shared authentication key

2. Append a Message Authentication Code (MAC) to each packet

3. After authentication, add chaff packets with bad MAC’s to the message stream

- MAC’s are added to message packets by calculating the MAC value and appending the fixed length MAC to each corresponding packet

Authentication Key

- Authentication key used to calculate MAC’s is shared by sender and receiver
- When recipient receives packets, they compute the MAC with the message-packet content and the shared key
- If the calculated MAC equals the transmitted MAC, the origin of the packet is known to be the sender and the packet is kept
- Otherwise when the calculated and transmitted MAC’s are not equal, the packet is discarded

Packet Sequence Number

- Assign each packet a sequence number for:
  - Removal of duplicate packets
  - Identification of missing packets
  - Correct ordering of received packets

- An example sequence might look like:

(1,Hello David,465231)

(2,The code is,782290)

(3,Joshua,344287)

(4,Sincerely-Falken,312265)

Adding The Chaff

- After authentication of the packets, chaff packets with incorrect MAC’s are added to the message stream
- It is important that chaff packets:
  - Are correctly formatted
  - Have reasonable sequence numbers
  - Have reasonable message contents
  - Have invalid MAC’s

Sample Message

(1,Hello Lizzie,532105)

(1,Hello David,465231)

(2,The code is,782290)

(2,Your contact is,793122)

(3,Flex,891231)

(3,Joshua,344287)

(4,Peace-Vic,553419)

(4,Sincerely-Falken,312265)

Receiver Separates Chaff

(1,Hello Lizzie,532105) ➔ *789326*

(1,**Hello David**,465231) ➔ 465231

(2,**The code is**,782290) ➔ 782290

(2,Your contact is,793122) ➔ *878933*

(3,Flex,891231) ➔ *643236*

(3,**Joshua**,344287) ➔ 344287

(4,Peace-Vic,553419) ➔ *003732*

(4,**Sincerely-Falken**,312265) ➔ 312265

MAC Algorithm Notes

- The MAC Algorithm is simply a hash function
- Once hashed, the original message cannot be retrieved
- It must look random to an eavesdropper, where similar inputs generate distinctly different outputs

Packet Construction Notes

- Smaller packets = more packets
- More packets = brute force more difficult
- Too small = inefficient use of bandwidth
  - Tough on recipient
  - Tough on eavesdropper

Chaff Construction Notes

- Creating chaff is easy/inexpensive, one can pick any sequence number and content and append a random MAC value
- If the MAC is 64 bits long, the chance of accidentally creating the correct MAC for the sequence number and content is extremely unlikely ($1/2^{64}$)
- Creating bogus MAC's for chaff packets requires no knowledge of the secret key

Chaffing / Winnowing Uses

- Transmitting secret codes, messages, and/or any other sensitive information

- Why use it instead of crypto?
  - Bypass U.S. government regulations on encryption exports
  - Exportable encryption defined as systems with key lengths <= 56 bits, or systems that implement key recovery/key escrowing
  - Chaffing and Winnowing is a direct challenge to encryption-restricting legislation, proposed by Ron Rivest
  - Chaffing and Winnowing doesn't fall under the restriction because messages are sent plain-text, using only a shared key for authentication
  - In essence to say; "Screw you Big Brother!"

Secret-Splitting / Sharing

What Is It?

- Just what it sounds like
- Take a secret, split it up, and share it – to make it more secure
- Explained in detail in forthcoming slides

Why Not Use Crypto Instead?

- One can encrypt data to protect it, but then you still need to protect the encryption key
- One can store the key in a secure location, but if it is lost you can never recover it and/or the data
- One can store multiple copies of the key in different locations, but then the potential for key compromise is increased
- To make the secret reliable and robust, the idea is to break the secret into pieces and then distribute those pieces to different persons in a group

Secret-Splitting / Sharing

- Secret-sharing scheme
  - Two parts:
    - Set of “Holders”
    - “Access Structure”
- Shares
  - Pieces of a split secret or information that every holder receives

Splitting Scheme Details

- Two Components:
  - Set of Holders: H={H1,H2,H3}
  - Access Structure set:
    - Subsets of holders who can discover the secret by pooling their information together;
      - i.e., AS={ {H1,H2}, {H2,H3}, {H1,H2,H3} }
    - Monotone access structure:
      - i.e., AS0={ {H1,H2}, {H2,H3} } consists of all “minimal” sets of AS

- A perfect secret-sharing scheme:
    - Any qualified subset can reconstruct the secret
    - Any un-qualified subset has absolutely no viable information on the secret

Access Structure Variance

- Generalized Access Structures:
  - For any set of Holders, there can be many different kinds of Access Structures
  - i.e., For a set of 3 users, Access Structures that may be employed by different secret sharing schemes include;
    - **A+B+C**, **A+BC**, **AB+BC**, **AB+BC+AC**, **ABC**

- Why study different Access Structures?
  - Different secret-sharing schemes may require different levels of fault tolerance and secrecy, which can be realized by different Access Structures
  - **A+B+C** increases the fault tolerance of the key, but it decreases the secrecy of the key
  - **ABC** increases the secrecy of the key, but the key becomes more unreliable
  - Also of great importance, different Access Structures may require different sizes of shares

Share Sizes

- For different Access Structures, they may require different bounds on the amount of information that its users must remember
- If the amount of information that must be kept secret increases, then the security of the system will degrade
- To prevent the appearance of *Bob* and *Alice,* and to keep the presentation time in check, we shall delve no further on this subset topic. A reference to more topic info is on the final slide.

Secret-Splitting Risks

- Risk-based consequences of authentication error include:
  - Inconvenience
  - Distress
  - Damage to organization/group programs or reputation
  - Financial loss
  - Personal safety

Secret-Splitting Uses*

- Nuclear launch codes
- Bomb components
- Chemical weapons ingredients
- Level 3,4 PKI management for legal non-repudiation
- Monetary asset protection
- Intellectual property protection
- Warez/file-sharing legal liability/protection
- Terrorist plots
- Illegal substance smuggling/possession

* Not necessarily endorsed/condoned by presentation author

Steganography

What Is It?

- Steganography = *covered writing*
  - cover-image + hidden-data + stego-key
- Use dates back several millennium:
  - wax tablets
  - messages tattooed on scalps
- Other examples:
  - Invisible ink
  - Hide an image under another image in a PPT presentation (cough)
  - Hide text in same color as background
  - Spammers employ by encoding spam messages into literature text
- Digital steganography can hide information in image/video/audio files (or any binary file)
- Primary “legitimate” use is digital watermarking

Why Is It Effective?

- **Point to ponder #1:**
  How do you know when you’ve run out of invisible ink?

- **Point to ponder #2:**
  Male mosquitoes buzz loudly but do not sting.
  Female mosquitoes are silent and do sting.
  Therefore if you DO hear a mosquito buzzing around, no worries.
  If you DO NOT hear buzzing, there may be trouble present.

- ***Ximenez* in Monty Python:**
  “*Nobody expects the Spanish Inquisition.*”

- ***Keyser Soze* in The Usual Suspects:**
  “*The greatest trick the devil ever pulled was convincing the world he didn’t exist*.”

Modern Implementations

- Modern steganography attempts to be detectable only if secret information is known
- For steganography to remain “undetected”, the unmodified cover medium must be kept secret
- If exposed, a comparison between the cover media and stego media immediately reveals the changes/differences

- An information-theoretic model that considers the security of steganography systems against passive eavesdroppers:
  - Assume that the adversary has complete knowledge of the encoding system, but does not know the secret key
  - Attacker must devise model for the probability distribution *PC* of all possible cover media, and *PS* of all possible stego media
  - Adversary can then use *detection theory* to decide between hypothesis *C* (that a message contains no hidden information) and hypothesis *S* (that a message carries hidden content)
- A system is perfectly secure if no decision rule exists that can perform better than random guessing

The Color of Secrets

- Image file colors are important in the amount of redundant bits available to hide data
- Steganography works best in cover files with; high energy, bright colors, high volume
- 24-bit color is *True Color*
  - 1 pixel requires three bytes, each representing level of Red/Green/Blue (RGB) color
    - Color of this line is denoted 0xbf-1d-98 [i.e., Red=191 (0xbf), Green=29 (0x1d), Blue=152 (0x98)]
  - 16,777,216 (2 to 24th power) possible colors/image
- 8-bit color is *also* True Color, but...
  - Image contains a palette with up to 256 (2 to 8th power) unique colors, each denoted by a 24-bit RGB value
  - Each pixel requires 1 byte to point to palette entry

Use of LSB Overwriting

- Concept is to embed a secret message in the Least Significant Bits (LSB’s) of images
- Works because the human visual system isn’t acute enough to pick out changes in color
- Basic algorithm for LSB substitution would be to take the first M cover pixels (where M is the length of the secret message to be hidden in bits), then replace every pixel's last bit with one of the message bits
- Problem with approach is one ends up with the first region of the image having different statistics than the rest of the image
- Amongst possible improvements an effective/common one is to seed a pseudo-random number generator then use the numbers output

LSB Substitution Example

- **“LSB** substitution” overwrites the least significant bit of target bytes
- Example: Hide "G" (01000111) in 3 pixels
  - Original Data
    - **10010101 00001101 11001001**
    - **10010110 00001111 11001011**
    - **10011111 00010000 11001011**
  - Stego Data
    - **1001010*0* 00001101 1100100*0***
    - **10010110 0000111*0* 11001011**
    - **10011111 0001000*1* 11001011**

Overview of “some” Stego Tools

S-Tools

- Designed for lossless compression - hides information inside BMP, GIF, or WAV files using LSB overwriting
- Password used for LSB randomization and encryption
- S-Tools allows conversion of 8-bit “cover” images to 24-bit, as well as attempted color reduction in any 24-bit images one wishes to embed within a 8-bit “cover”
- Can hide multiple files within a single image or sound file
- Multiple algorithm support

JPHS

- Designed for lossy compression - hides information inside JPG files using LSB overwriting of DCT coefficients
- No installation required - two programs, JPHIDE and JPSEEK - JPHIDE hides data in JPG file, JPSEEK recovers file hidden with JPHIDE, and JPHSWin.exe performs both/either function(s)
- Linux version also available
- Distributes the hidden file in the JPG image so that both the visual and statistical effects are minimized
- Uses LSB overwriting of the discrete cosine transform coefficients used by the JPG algorithm (difference in file statistics minimized, decreasing risk of unauthorized recovery)
- Uses Blowfish crypto algorithm for LSB randomization and encryption (to determine where to store the bits of the hidden file)

MP3 Stego

- Compresses, encrypts, then hides data in MP3 bit stream
- Linux version also available
- Written with stego applications in mind, but could be used as copyright marking system for MP3 files (weak but still much better than the MPEG copyright flag defined by the standard)
- Opponent can uncompress the bit stream and recompress it, which will delete the hidden information (only attack known) – but at the expense of severe quality loss
- The hiding process takes place at the heart of the Layer III encoding process…namely in the *inner_loop*
- Requires text file and wav file - converts combined output to MP3 file

Camouflage

- Camouflage allows one to hide files by first scrambling them and then attaching them to file of choice
- Appends hidden file to carrier file
- Camouflaged file looks and behaves like normal file (can be stored, used, or emailed without attracting attention)
- Can password-protect camouflaged file
- Can camouflage files within camouflaged files

Stego (for Mac)

- Embed data in and retrieve data from Macintosh PICT format files, without changing the appearance or size of the PICT file
- Works by slightly altering pixel values
- Rasterizes image, then stegs data into the LSB of each of the RGB color values
- In the case of indexed color, Stego stegs data into the LSB of the index values
- File length of data file to be stegged is hidden in the LSB's of the first 32 steggable bytes
- To disguise value somewhat, it takes the 2nd-least significant bits of the 2nd 32 steggable bytes, XOR’s them with the 32 bit file length, then stegs the XOR'd file length into LSB's of the first 32 steggable bytes

NCovert

- NCovert designed to function as a TCP covert channel
- Runs on Linux
- Based in part on Chris Rouland's covert_tcp program
- Program functions as both client and server, depending options used
- Data transferred using the IP ID field and TCP sequence number – loose form of stego

Hydan

- Hides data in a Windows or Linux binary (executable) file
- Takes advantage of redundancies in i386 assembler
  - i.e., A+B vs A-(-B)
- Can hide one byte in ~110 instruction bytes
- Maintains size of carrier file

http://www.crazyboy.com/hydan/

StegFS

- Hidden file system for Linux
- Does not require the use of a separate partition of a hard disk
- Instead places hidden files into unused blocks of a partition that also contains normal files, managed under a standard file system
- Uses a separate block allocation table
- Fulfills a similar function as the block allocation bitmap in traditional file systems, but instead of a single bit for each block it contains an entire 128-bit encrypted entry
- Encrypted entry indistinguishable from random bits unless the key for accessing a security level is available
- Steganographic file system driver isn’t well hidden, but doesn’t need to be

Detecting Steganography Use

- WetStone Technologies (commercial)
  - **Gargoyle** (formerly StegoDetect)
    - finds remnants of stego (or other malware) software using file hashes
  - **Stego Suite** (Stego Analyst, Stego Break, Stego Watch)
    - applies statistical methods on suspect files to determine probability that stego was employed, guesses which algorithm was employed, and attempts to break the password

- Neils Provo (outguess.org) - FREE
  - **Stegdetect**
    - detects stego in JPG images using several algorithms
  - **Stegbreak**
    - launches brute-force dictionary attacks on JPG image
  - Runs on Windows or Unix

Online Tools / Examples / Demos

- http://www.well.com/~sthomas/.Secret-Page.html

- GifItUp (jpg in gif example)
- S-Tools (jpg in bmp example)
- JPHS (jpg in jpg example)
- MP3Stego (txt in mp3 example)
- Camouflage (jpg in jpg example)
- Camouflage (jpg in xls example)

Steganography Uses*

- Hiding/sharing codes or other sensitive information
- Hiding images of former girlfriend/boyfriend or current mistress/master from current partner
- Hiding embarrassing images of one’s self
- Hiding/sharing illegal pr0n
- Hiding/sharing illegal plans and terrorist plots
- Hiding/sharing “evil” source code
- Hiding/sharing cracks/serials/passwords/netcat backdoors, etc.

* Not necessarily endorsed/condoned by presentation author

Hiding / Sharing Secrets Physically

Spy Tactics/Practices

- Don’t steal secrets – make surreptitious copies
- Establish creative two-way communication
- Use “dead drops” – parties “never” meet
- Employ creative “envelopes” for secrets to be exchanged
- Use common/open/public areas for communication and dead drop sites

Spy Tactics/Practices (cont.)

- Must keep secrets hidden until time of transfer
- Caching
- Accessibility weighed against secrecy
- Booby-trapped containers
- 30 unique forms/categories of concealment for spies
- Destroy obsolete records, receipts, tools, maps, passwords, etc.

Two-Way Communication

- Two-way communication may be needed for secret(s) delivery, payment, etc.
- Must have signals to note that drop has been made at dead-drop location
- May need separate signal that money or other trade item has also been left in return
- Strategically placed drink can, broken tree branch, Christmas lights, chalk marks, obscure signal flags, etc., etc., etc.

Secret “Envelopes”

- Only limited by imagination
  - * Inside smoke detector (airplane bathroom, stairwell, etc.)
  - * Inside books/magazines (library, bookstore)
  - * Taped to underside of desk or chair
  - * Fake (or real) electrical outlet
  - * Box of tissues
  - * Velcro-ed patches on clothes, backpacks
  - Inside electronic devices
  - Behind back plate/battery of cell phone
  - Hole in wall covered with flier or poster
  - Taped to top of door
  - Diaper genie, cat litter box, sharps medical waste disposal
  - False bottom containers
  - Laptop/PC drives
  - Inside dog food bag, cereal box, detergent box, etc.
  - Car steering wheel, door panel, under back seat, etc.
  - Junk mail envelopes
  - Gutted VHS tape
  - Buried
  - Stack of paper plates
  - Band-aid tin, bandages
  - Inside shower curtain rod
  - (*continued*)

Secret “Envelopes” (cont.)

- Only limited by imagination
  - * Inside walnuts
  - * Inside currency
  - * Calculator
  - * Video of secret docs
  - * Drop item attached to string and washer inside wall from ceiling/attic, retrieve when needed using strong/small magnet on string
  - * Hollow glass eyeball
  - HVAC unit filter
  - Dog's collar
  - Molding covering hollowed area
  - Hollowed out planter stand (reverse-thread so opening isn’t obvious)
  - False bottom drawer
  - Hollow shoe heel
  - Plant container
  - Children’s toys
  - Hide important humans in city ghetto, hospital, etc.
  - Create hollow chocolate bar mold, insert secret, pour other bar side sealing secret in, wrap
  - Inside cigarette
  - Tattoo to scalp, shave head to read
  - Inside body (swallow, up anus or vagina, under tongue, under skin)
  - *etc., etc., etc.*

Dead Drop Uses*

- Drug deals
- Illegal classified information exchange
- Corporate espionage information exchange
- Bio/nuclear weapons exchange
- Stolen goods exchange
- Exchange diskette with “Leonardo da Vinci” virus, culled from the hacked “garbage” file on the Gibson
- Use your imagination…..

* Not necessarily endorsed/condoned by presentation author

A Few “Live” Containers

References / Additional Reading

- Gary Kessler stego links:
- (*www.garykessler.net/library/securityurl.html#crypto*)
- • Gary Kessler's awesome overview of Stego:
- (http://www.garykessler.net/library/fsc_stego.html)
- • Ron Rivest paper on Chaffing and Winnowing:
- (http://theory.lcs.mit.edu/~rivest/chaffing.txt)
- • Notre Dame student paper on Secret Splitting:
- (http://www.nd.edu/~cseprog/proj02/cryptogrophy/final.pdf)
- • Scott Guthery paper on Secret Splitting:
- (http://arxiv.org/ftp/cs/papers/0307/0307059.pdf)

The End