Steganography in Commonly Used HF Radio Protocols

@pdogg77 @TheDukeZip
pdogg

- Paul / pdogg / @pdogg77
- Day Job: Security Researcher at Confer Technologies Inc.
- Hobby: Licensed as an amateur radio operator in 1986, ARRL VE
- This is my second trip to DEF CON
thedukezip

- Brent / thedukezip / @thedukezip
- Software & Systems Engineer (RF)
- Licensed ham radio op since 2006, ARRL VE
Why You Shouldn't Do This And Why We Didn't Do It On The Air

FCC Regulations (Title 47 – Part 97)
§ 97.113 Prohibited transmissions.
(a) No amateur station shall transmit:

…

(4) Music using a phone emission except as specifically provided elsewhere in this section; communications intended to facilitate a criminal act; messages encoded for the purpose of obscuring their meaning, except as otherwise provided herein; obscene or indecent words or language; or false or deceptive messages, signals or identification.
How This Project Started...

Final Warning Slide...

- Hackers + Drinks = Project
- WANC - We are not cryptographers
- We are not giving cryptographic advice
- You should talk to a cryptographer
- If you are a cryptographer, we welcome your input
What?

We set out to demonstrate it was possible (or impossible) to create a:

- Low Infrastructure
- Long Range
- Covert
- Point to Point, Broadcast or Mesh
- Short Message Protocol

Using existing consumer radio and computer equipment, leveraging a commonly used digital mode
Why?

- Avoid censorship
- Avoid spying

- We believe you have the right to communicate without this interference
- You COULD use our method to communicate, OR use similar techniques to create your own method
... Or “The Terrorists”
No Internet?

Amateur radio operators have expertise in this!
Amateur Radio

- Many frequency bands reserved for amateur radio operators to communicate
- Voice chat, digital modes...
- Take a multiple choice test to get licensed

Reminder: The rules say you can't do what we're showing you...
AirChat

- Anonymous Lulzlabs
- Encrypted communication in plain sight
- Cool project with a different purpose
- Also breaks the rules
Good Steganography / Good OPSEC

- ... means hiding well in plain sight.
- Invisible to normal users
- “Plausible deniability”
- Not this →
More Like This
Ways to Hide...

- Protocol features (headers, checksums etc)
- Timing or substitution
- Errors
- No “spurious emissions” etc... (against the rules, obvious, very “visible”)

Candidate Protocol must:

  ... be in widespread common use
  ... have places to hide
  ... be relatively power efficient

Need no special hardware or closed software
Popular Sound Card Digital Modes

- RTTY
  - In use on radio since at least the 1920s
  - Baudot code – 5 bit symbols with a stop and a shift – “mark and space”
  - Amateurs almost always use a 45 baud version with 170hz carrier shift
  - Limited character set

- PSK31 etc.
  - Phase shift keying 31 baud...
  - Developed by Peter Martinez G3PLX in 1998
  - VERY tight protocol - “Varicode”
JT65

- Developed by Joe Taylor – K1JT – 2005
- Original paper: “The JT65 Communications Protocol”
- Designed for Earth-Moon-Earth communications. Also now widely used for skywave contacts
- Very power efficient
- Structured communication, very low data rate
- Open source implementation
JT65 Conversations

Some Common HF Ham Freqs:
20m 14.076MHz
15m 21.076MHz
10m 28.076MHz
Upper Side Band
Some JT65 Technical Details

KB2BBC
KA1AAB
DD44
Some JT65 Technical Details

User Message
Some JT65 Technical Details

User Message

KB2BBC KA1AAB DD44

34 20 05 42 26 09 03 05 60 06 24 22
Some JT65 Technical Details

User Message
Source Encoding
Some JT65 Technical Details

<table>
<thead>
<tr>
<th>Source Encoding</th>
<th>User Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>34 20 05 42 26 09 03 05 60 06 24 22</td>
<td>KB2BBC KA1AAB DD44</td>
</tr>
</tbody>
</table>
Some JT65 Technical Details

User Message

Source Encoding

FEC
Some JT65 Technical Details

User Message
Source Encoding
FEC
Some JT65 Technical Details

User Message
Source Encoding
FEC
Some JT65 Technical Details

User Message
Source Encoding
FEC
Some JT65 Technical Details

User Message

Source Encoding

FEC

Matrix Interleaving
Some JT65 Technical Details

User Message
Source Encoding
FEC

Matrix Interleaving
Some JT65 Technical Details

User Message
Source Encoding
FEC
Matrix Interleaving
Gray Coding
Some JT65 Technical Details

User Message
Source Encoding
FEC
Matrix Interleaving
Gray Coding
Audio

- JT65 “packet” sliced into 126 .372s intervals – 47.8s
- 1270.5 Hz sync tone - “pseudo-random synchronization vector”
- Symbols - $1270.5 + 2.6917(N+2)m$ Hz
  - $N$ is the integral symbol value, $0 \leq N \leq 63$
  - $m$ assumes the values 1, 2, and 4 for JT65 sub-modes A, B, and C
Hiding in Reed Solomon Codes

- Exploit error correction!
- Easy/PoC Mode: Shove in some errors... :) (static “key”)
- Medium mode: Shove in errors, add some random cover
- Hard Mode: Encrypt and pack message, add FEC
- Prior Work: Hanzlik, Peter “Steganography in Reed-Solomon Codes”, 2011
Encoding Steganography (Basic)

Steg: DEF CON 22
Encoding Steganography (Basic)

Steg: DEF CON 22

Source Encoding:

19 51 00 26 06 17 52 04 31 15 56 28
Encoding Steganography (Basic)

Steg: DEF CON 22

Source Encoding:

```
19 51 00 26 06 17 52 04 31 15 56 28
```

FEC:

```
24 42 21 21 43 42 56 22 19 51 00 26 06 17 52 04 31 15 56 28
```

Can tolerate 4 errors
Hiding Steganography

Key: pdogg thedukezip

Generate 20 'locations' based on SHA512
Hiding Steganography

Key: pdogg thedukezip

Generate 20 'locations' based on SHA512

09 29 41 35 20 32 27 15 23 18 53 12 45 03 13 40 49 22 25 37
Injecting Errors

JT65: KB2BBC KA1AAB DD44

<table>
<thead>
<tr>
<th>39</th>
<th>19</th>
<th>16</th>
<th>44</th>
<th>29</th>
<th>13</th>
<th>58</th>
<th>19</th>
<th>13</th>
<th>14</th>
<th>20</th>
<th>44</th>
<th>17</th>
<th>20</th>
<th>25</th>
<th>31</th>
<th>46</th>
<th>02</th>
<th>29</th>
<th>35</th>
<th>56</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>11</td>
<td>20</td>
<td>39</td>
<td>51</td>
<td>07</td>
<td>30</td>
<td>26</td>
<td>11</td>
<td>17</td>
<td>27</td>
<td>21</td>
<td>11</td>
<td>30</td>
<td>34</td>
<td>46</td>
<td>48</td>
<td>15</td>
<td>53</td>
<td>14</td>
<td>26</td>
</tr>
<tr>
<td>12</td>
<td>07</td>
<td>05</td>
<td>08</td>
<td>42</td>
<td>41</td>
<td>37</td>
<td>19</td>
<td>16</td>
<td>35</td>
<td>63</td>
<td>20</td>
<td>03</td>
<td>12</td>
<td>38</td>
<td>26</td>
<td>08</td>
<td>37</td>
<td>22</td>
<td>23</td>
<td>29</td>
</tr>
</tbody>
</table>
Injecting Errors

JT65: KB2BBC KA1AAB DD44

Steg: DEF CON 22
Injecting Errors

JT65: KB2BBC KA1AAB DD44

Steg: DEF CON 22

Key: pdogg thedukezip
Injecting Errors

JT65: KB2BBC KA1AAB DD44

Steg: DEF CON 22

Key: pdogg thedukezip
Injecting Errors

JT65: KB2BBC KA1AAB DD44

Steg: DEF CON 22

Key: pdogg thedukezip
Injecting Errors

JT65: KB2BBC KA1AAB DD44

Steg: DEF CON 22
Key: pdogg thedukezip
What About Encryption?
What About Encryption?

- We have $12 \times 6 = 72$ bits to play with
- We need 8 bit bytes...
- Well that gives us exactly 9 bytes
What About Encryption?

- We have $12 \times 6 = 72$ bits to play with
- We need 8 bit bytes...
- Well that gives us exactly 9 bytes
"Packing" Function

<table>
<thead>
<tr>
<th>Status 1 byte</th>
<th>Data 8 bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000001</td>
<td>11001001</td>
</tr>
<tr>
<td>10010011</td>
<td>10110001</td>
</tr>
<tr>
<td></td>
<td>11110010</td>
</tr>
<tr>
<td></td>
<td>01111000</td>
</tr>
<tr>
<td></td>
<td>00011001</td>
</tr>
<tr>
<td></td>
<td>00001001</td>
</tr>
</tbody>
</table>
“Packing” Function

Status
1 byte

Data
8 bytes

10000001
10010011
11001001
10110001
11110010
01111000
10000001
01110010
00101010
00011001
01111000
00001001

Steganography
12 6-bit symbols

100000 011100 100110 110001 111100 100111
100010 010011 001010 100001 100100 001001
100010 010011 001010 100001 100100 001001
"Status" Byte

- Track how many total packets in message
- Flags for first / last packet
- Track size for stream ciphers

Waiting for start of minute...
Decoded JT65 message 0 : KB2BBC KA1AAB DD44
Steg detected! (1/3) total packets received.
Monitoring...
Decoding...
Waiting for start of minute...
Decoded JT65 message 0 : KA1AAB KB2BBC DD44
Steg detected! (2/3) total packets received.
Monitoring...
Decoding...
Waiting for start of minute...
Decoded JT65 message 0 : KB2BBC KA1AAB DD44
Hidden message : SEE YOU AT DEF CON 22
“Status” Byte – Stream Cipher

First packet: (0x80) | (# of total packets)

Middle packets: Packet Number

Last packet: (0x40) | (# of bytes in packet)

Max size: 64 packets (512 bytes)

1 bit 1 bit 6 bits

| First Packet? | Last Packet? | First? : # of total packets
|---------------|--------------|-----------------------------
|               |              | Last? : # of bytes in packet
|               |              | Else : Packet Number        |
# “Status” Byte – Block Cipher

<table>
<thead>
<tr>
<th>First Packet?</th>
<th>First? : # of total packets</th>
<th>Else : Packet Number</th>
</tr>
</thead>
</table>

First packet: 
(0x80) | (# of total packets)

Other packets: 
Packet Number

Max size: 128 packets (1024 bytes)
Hiding the Status Byte

- We'll talk about analysis in a bit...
- Steganography traffic was trivial to pick out of normal traffic because of this byte :(

![Uh-oh spaghettios](image)
Perform Bit Swap

<table>
<thead>
<tr>
<th>Status</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 byte</td>
<td>8 bytes</td>
</tr>
<tr>
<td>100000001</td>
<td>11001001</td>
</tr>
<tr>
<td>10010011</td>
<td>10110001</td>
</tr>
<tr>
<td>10010011</td>
<td>11110010</td>
</tr>
<tr>
<td>01111000</td>
<td>00001001</td>
</tr>
<tr>
<td>00011001</td>
<td>00011001</td>
</tr>
<tr>
<td>00001001</td>
<td>00001001</td>
</tr>
</tbody>
</table>
Perform Bit Swap

<table>
<thead>
<tr>
<th>Status 1 byte</th>
<th>Data 8 bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>10111000</td>
<td>11001001</td>
</tr>
<tr>
<td>00010011</td>
<td>10110001</td>
</tr>
<tr>
<td>00101010</td>
<td>01110010</td>
</tr>
<tr>
<td>00110001</td>
<td>00011001</td>
</tr>
<tr>
<td></td>
<td>01001001</td>
</tr>
</tbody>
</table>
Perform Bit Swap

Status
1 byte

Data
8 bytes

10111000 11001001 10110001 01110010 00111000
00010011 00101010 00011001 001001

Steganography
12 6-bit symbols

101110 001100 100110 110001 011100 100011
100000 010011 001010 100001 100001 001001

001110 011100 101100 110010 001110 100011
000101 000011 001010 000110 001010 001001
Tool Demo...
“Feed Reader” RasPi Demo...
Decoding...
Waiting for start of minute...

Decoded JT65 message 0 : HE0JER WI5LB

Decoded JT65 message 0 : KCØREY KB4DCG 73
Monitoring...
Decoding...
Waiting for start of minute...

Decoded JT65 message 0 : EI5HV K4RAD EM82
Monitoring...
Decoding...
Waiting for start of minute...

Decoded JT65 message 0 : W9OY NJ9U RRR

Decoded JT65 message 0 : CQ K0KC EN80
Monitoring...
Analysis/Steganalysis

• Defined set of legitimate JT65 packets
• “Known Cover Attack”
• Receive packet → Decode → Encode
• Demodulator provides “probability” or confidence
• Theory:
  - Packets suspected to contain steganography can be easily distinguished by some quantitative measure
Analysis Module
Finding Steganography is Easy
Finding Steganography is Hard
Finding Steganography is Hard
Interesting Patterns (and a warning)
Distance

- Considering we cannot SEND these packets
- Let's pretend we received them (<= 7 errors)
- How far away were the senders?

Number of packets in set with 7 or less errors: 5631
  - 2659 have distance data
  - Max Distance of Set: 10518.49107
  - Median Distance of Set: 1179.20450965
  - Average Distance of Set: 1858.87448753
  - 90% Distance of Set: 3687.56180076
Effectiveness as a World Wide Short Message Protocol
“Vulnerabilities” / Known Limitations

- Analysis and Detection
  - As discussed / other methods
- Transmitter location (foxhunting)
  - Well studied problem/game by amateurs and TLAs
  - FCC/DEA/NSA - SANDKEY(1)
- Message Forgery
- Storage / long term cryptographic analysis

(1) http://cryptomeorg.siteprotect.net/dea-nsa-sandkey.pdf
How to get it?

PRIVATE pdogg / jt65stego

jt65stego project

157 commits

3 branches

branch: master jt65stego / +
Available today!

Oh yeah, it's on your conference DVD too...
Conclusions

● Protocols and methods such as those presented can, in theory, provide a platform for short message communications with desirable properties:
  - Low infrastructure
  - Long distance
  - Covert
  - Plausibly deniable

● Potential for analysis and detection
  - Especially for well equipped adversaries
Next Steps / Further Areas of Study

- Continued Detection / Counter Detection Work
- Cryptographic Improvements
- Enhanced amateur applications
- Useful protocols and networks
Ham Exam
Crypto & Privacy Village
Sunday 12 PM – 3 PM

Cram Session
Wireless Village
Sunday 9 AM – 12 PM
THANKS!

@pdogg77
@TheDukeZip

https://www.github.com/pdogg/jt65stego/

Special Thanks @masshackers