GMR-1 Speech codec
GMR-1 Speech codec
The problem

- AMBE: Advanced Multi-Band Excitation
- Not documented in the standard
  - Barely a high level description
  - No reference code
- Proprietary codec by *DVSI Inc.*
  - Not supported by their "cheap" hardware USB decoder
  - Cheapest hardware is the NET-2000 appliance (2kEUR)
- But:
  - mbelib: Code for other documented IMBE/AMBE variants (P25)
  - Implemented in SO2510 phone DSP (TI C55x)
AMBE Codec

Description

- Highly specialized for voice (vocoder)
- Divides speech in small segments
  - For GMR-1: 20 ms frames subdivided into co-quantized 10 ms sub-frames
- Represent each speech (sub)frame as a set of parameters
  - $f_0$: Fundamental frequency (*pitch*)
  - $G$: Gain (*volume*)
  - Voiced / Unvoiced decision (per band)
  - Spectral Magnitudes
- Decoding can be summarized as 3 steps:
  - **Unpacking**: Unpack the raw frame bits into quantized parameters
  - **De-Quantization**: From quantized parameters to actual values
  - **Synthesis**: From the parameters set to actual audio
AMBE Codec

Synthesis

- Fundamental "pitch" $f_0$
- Voiced
- Unvoiced
- Spectral
- Envelope

Frequency $> 4$ kHz

Amplitude
AMBE Reversing

DSP Code analysis

- Target: SO-2510 phone
- Codec has to be in the DSP, nowhere else it could be!
- DSP firmware extracted from firmware update package
  - Supported by IDA
- But where?
  - 250k binary blob
  - No strings
  - Obscure TI C55x assembly
- Dieter Spaar to the rescue!
  - Identified entry points for encode/decode functions
  - Look for Audio DMA / Interrupts
  - Search for constants
  - Stack Switching
AMBE Reversing

Simulator

- TI Code Composer Studio - Simulator
  - Accurately simulates supported DSP
  - Arbitrary memory layout
  - fread() / fwrite() from host
  - Tracing of all memory access
  - Windows only :(

- Use the original firmware to decompress audio for us
  - DSP dump converted to a valid COFF .OBJ file for linking
  - Custom linker script
  - Simple main() that fread() frames and fwrite() audio

- Success!
  - It took quite a few tries, lots of traps
  - But it works and we get audio out
  - Slow (not real-time) and not practical though
Real HW would be faster and more convenient. But:
- Code has to run at the physical address it has been linked for
- OMAP has a DSP MMU, but standalone DSP don’t
- Need a cheap board with a compatible memory map

Dieter found one with SDRAM where needed and Ethernet
- Success! About 16x faster than real-time
- SDRAM is not fast, relocate some data tables to SRAM

I intended to buy the same board
- But in my haste ... I ordered the wrong one ... *facepalm*
- No SDRAM, more SRAM, but at the wrong physical address
- Easy, just relocate the code! Can’t be that hard, right?
- Use IDAPython + simulator trace mode
- Success!
Hardware USB decoder is nice, but not enough

Decompression process:
  - **Unpacking**
    - Early, simple bit manipulation, easy to follow
  - **Dequantization**
    - Easily 95% of the work
    - Hard to follow fixed point math in DSP assembly
  - **Synthesis**
    - Started by just re-using mbelib code
    - Then rewrote using P25 specs and some guessing

Resulting PoC/reference code in GIT
  - Not same audio quality as the original, but perfectly intelligible
A better attack

Overview

- Based on the same A5/2 GSM attack
  - Don’t do anything fancy, just tweak for A5-GMR-1
- Both known-plaintext and ciphertext-only variant
- Targets FACCH3 control frames instead of TCH3 voice frames
- FACCH3 advantages:
  - Simpler modulation and better training sequence → less bit-errors
  - Predictable plaintext → known-plaintext attacks
  - Much more redundancy (more FEC) → less bursts needed for ciphertext-only attacks
  - Used to negotiate TCH6/TCH9 channels → attack works for CSD/Fax
A better attack

Known plaintext

Sylvain Munaut

osmo-gmr: What’s up with sat-phones ?
Goal is to describe cipher as a linear operation: $A \cdot x = b$

- $A$ = matrix describing cipher, $x$ = internal state and $b$ = cipher stream
- Each row of $A$ and $b$ is a bit of the output

Internal cipher state dependency on FN and Kc is linear

- Possible to combine equations from different bursts at different FN
- Can recover Kc from the state

Non-linear elements:

- Majority function: $M(a, b, c) = a + b \cdot c$
  - Introduces quadratic terms
  - Linearize by adding one new unknown for every possible quadratic term
  - 594 new unknowns

- Irregular clocking depending on R4 value
  - R4 is 17 bits but one is forced to '1' at init. Small enough for brute force!
  - Assume a given value for R4
  - Repeat 65536 times
In $A_n \cdot x = b$, some equations are redundant

- We can get a parity-check matrix $H_n$ such that $H_n \cdot b = 0$
- Those 65536 $H_n$ matrices can be precomputed offline
- With a single matrix-multiply we can check if a given R4 value is even a possibility
  - If result is non-zero, we can skip that R4 value
  - If result is zero, then we try to solve the system
  - In practice, only a few R4 value ever matches
A better attack
Ciphertext only

- Channel coding operation: $m = d \cdot G + g$
- Let $H$ be the parity-check matrix so that $H \cdot (m + g) = 0$
- Encryption operation: $y = m + b$
- $H$ can be used to derive equations from the ciphertext $y$:  
  $H \cdot (y + g) = H \cdot (m + b + g)$  
  $= H \cdot b + H \cdot (m + g) + 0$  
  $= H \cdot A \cdot x$
- The same R4 quick-scan technique can also be used here
- To get enough equations for a unique solutions, multiple frames are needed
A better attack

Results

- Known-plaintext variant
  - Requires between 4 and 8 bursts depending on alignment
  - Space: 50 Mb
  - Time: 500 ms

- Ciphertext-only variant
  - Requires 8 consecutive bursts belonging to 2 FACCH3 L2 frames
  - Space: 5 Gb
  - Time: 1 s
Final words
Future

- C-band
- Packet Data (GmPRS)
- Upper layers implementation
- CSN.1 and 04.008 code generators
- TX side

Help welcome :)
Other satellite phone systems

- We choose Thuraya because:
  - Visible from Europe
  - Cheapest sat phone on ebay
  - Specifications mostly available

- Don’t think other are better without proof
  - Availability of commercial intercepts tend to say otherwise
Thanks to anyone who contributed to this projects and related ones. Most notably:

- Dimitri "horizon" Stolnikov
- Dieter Spaar
- RUB team
Further reading

- GMR-1 in general
  - 28C3 talk  [http://gmr.osmocom.org/trac/blog/28c3-recording](http://gmr.osmocom.org/trac/blog/28c3-recording)
  - GMR1 Specs  [http://www.etsi.org/standards-search](http://www.etsi.org/standards-search)

- AMBE Codec

- GMR-1 Cipher
  - RUB GMR page  [http://gmr.crypto.rub.de/](http://gmr.crypto.rub.de/)