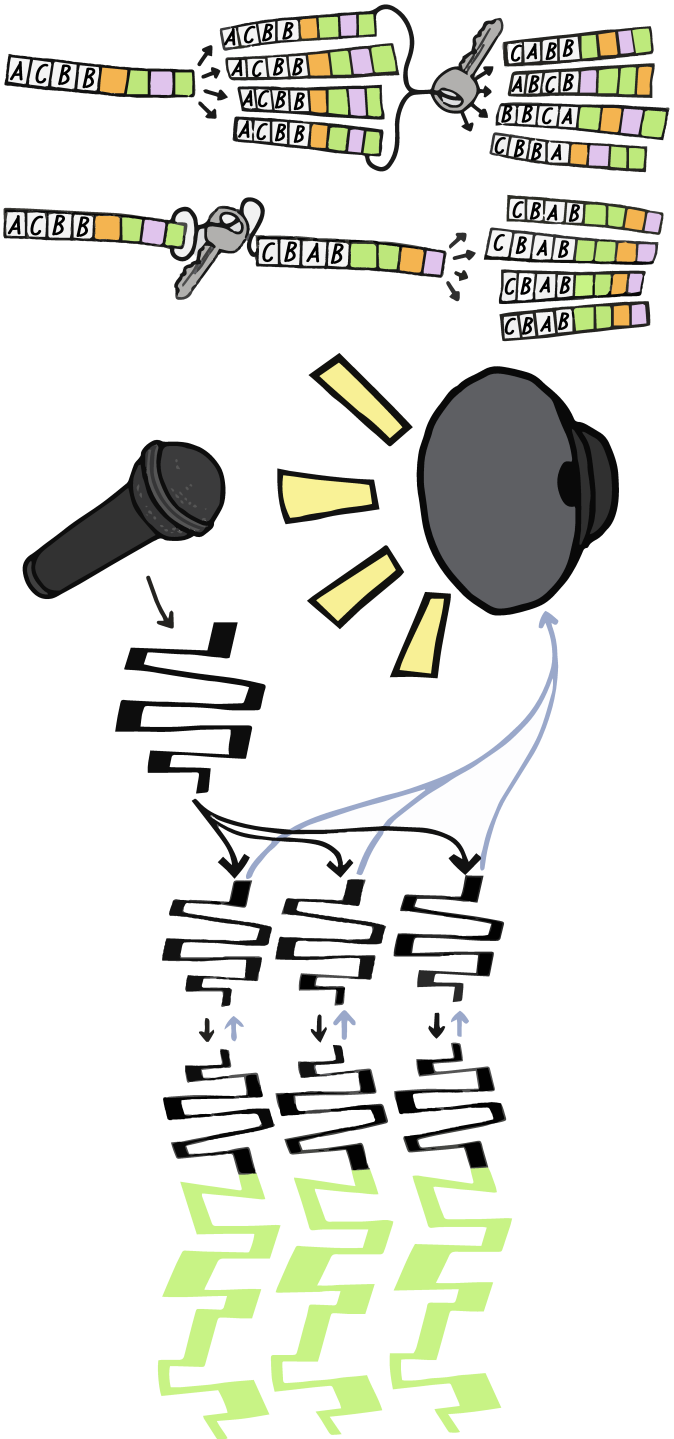
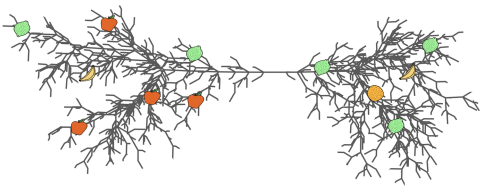
You Would Like to Order First



Top (Multiplex → Encrypt): The same message (ACBB) is first combined with different color-coded templates through multiplexing. Each variant is then independently encrypted using a shared key. This leads to distinct ciphertexts, but if the templates are known or public, attackers can reverse the multiplexing process and end up with multiple ciphertexts of the same underlying message, enabling algebraic attacks that exploit the known relationships between the variants.
Bottom (Encrypt → Multiplex): The message is first encrypted (producing, for example, CBAB from the original message), and the resulting ciphertext is then duplicated and wrapped in different color-coded templates. Because all copies are cryptographically identical before templating, multiplexing adds no diversity to the encrypted content. This offers attackers fewer opportunities for correlation-based attacks.
Signal Flow (Communication Path): The bottom illustration shows the complete communication path: microphone input → multiplexing → encryption → transmission → decryption → demultiplexing → speaker output.

You Would Like to Order First

GSM mobile communications used stream ciphers to protect call privacy, but protocol design left them vulnerable. Encryption was applied only after error correction and formatting, so fixed training sequences and redundant coding produced predictable ciphertext patterns. These leaks allowed attackers to recover session keys with modest effort, demonstrating that security depended less on theoretical cipher strength than on overall system design.



GSM TDMA Structure ◦ Fixed Burst Format ◦ Processing Order Vulnerability ◦ Stream Cipher XOR ◦ Predictable Plaintext ◦ Convolutional Codes ◦ A5/2 Algebraic Attack ◦ Cipher Downgrade Attack ◦ Session Key Reuse ◦ Biham-Barkan-Keller ◦ Protocol Layer Weakness

"I just found out that my cellular telephone was a lemon."

— Tobias Funke, 2004

You Would Like to Order First

> The history of secure communication traces back to the intersection of two twentieth-century revolutions: cryptography and wireless technology. For much of the century, radio systems prioritized reach and reliability over confidentiality. During World War II, breakthroughs such as the German Enigma and Allied SIGSALY systems introduced large-scale encryption in radio, but these were bespoke wartime inventions, not standardized infrastructure.
>
> In the postwar decades, civilian telecommunication networks expanded rapidly, but radio links remained analog and vulnerable. First-generation mobile systems (1G), such as AMPS in the United States and NMT in Scandinavia, used frequency modulation without encryption. Voice data was transmitted as analog waveforms, easily intercepted by anyone with a scanner. This vulnerability, while tolerated during the early novelty of mobile phones, became untenable as usage grew.
>
> By the 1980s, Europe pursued a unified digital cellular standard under the banner of the Groupe Spécial Mobile (GSM). The aim was not only cross-border interoperability, but also improved spectrum efficiency and — critically — built-in security. The digital transition allowed for integration of error correction, time-multiplexing, and cryptography into the protocol stack. Unlike analog predecessors, GSM was designed from the outset to offer some degree of confidentiality on the radio link.

Digital cellular networks are designed to carry simultaneous conversations — across a limited radio spectrum. Each call consists of two independent data streams, one uplink and one downlink, connecting handset and base station. These streams must be separated both by directionality and from the traffic of other users occupying the same band.

So we have duplexing and multiplexing. Duplexing separates the uplink (phone to tower) from the downlink (tower to phone). In frequency division duplexing (FDD), each direction is assigned their own frequency band, allowing simultaneous transmission and reception. In time division duplexing (TDD), both directions share a common band but alternate in fixed, synchronized time slots.

Multiplexing separates users sharing the same physical channel. In frequency division multiple access (FDMA), the spectrum is divided into separate frequency bands, each assigned to a different user. This isolates signals but requires fixed bandwidth allocation and limits how flexibly users can be added or removed. In time division multiple access (TDMA), users transmit in alternating time slots within a repeating frame structure. Each user has exclusive access to the channel during its assigned slot. This improves spectral efficiency, but requires strict global timing to keep transmissions aligned. In code division multiple access (CDMA), all users transmit simultaneously over the same frequency band, but each encodes its data using a unique pseudorandom spreading code. The receiver uses correlation to extract the intended signal. This allows full-time transmission with statistical multiplexing, but demands signal separation. All three approaches require that each user's transmission be confined to a fixed envelope, a burst, with predictable alignment and duration.

©2026 David H. Silver, CC BY-NC 4.0

These requirements propagate upward through the entire transmission stack. Each burst must arrive in its designated slot, with precise size and timing. Modulation, equalization, and error correction depend on this regularity. As a result, every upstream layer, from speech encoding to encryption, must preserve the burst format.

To transmit speech, the analog signal is sampled at regular intervals (Shannon, 1949) and each sample is encoded as a digital number. This raw bitstream is then compressed using a speech codec, a specialized algorithm that reduces bandwidth by representing only perceptually important features. As a toy example, consider a 20 ms segment of audio. Uncompressed, this might require over 2,000 bits. A codec might instead describe it using only pitch, volume, and phoneme class, reducing the bitrate by an order of magnitude.

GSM (the Global System for Mobile Communications) was developed as a pan-European standard for digital cellular networks in the early 1990s. It replaced earlier analog systems with a structured, time-synchronized digital stack designed for interoperability, moderate confidentiality, and efficient spectrum use. The GSM radio interface is based on TDMA: each 200 kHz carrier is divided into repeating time frames of eight slots, with each user assigned one slot per frame. Each slot (or burst) carries 114 bits of payload, framed by synchronization and guard bits.

Voice is transmitted as a sequence of such bursts. Every 20 milliseconds of speech is compressed into a 260-bit frame, which after coding and interleaving is split across multiple 114-bit radio bursts for transmission. These bits are divided into classes by perceptual importance. The most critical will later be protected with more redundancy. Each frame is processed independently and must be transmitted in order, aligned to the caller's assigned slot. From this point forward, it is treated as a fixed-length atomic unit: encoded, encrypted, and modulated as a whole. Control channels such as SACCH (Slow Associated Control Channel) follow a similar pattern, expanding 184-bit messages to 456 bits after error correction coding.

Before transmission, the frame is convolutionally encoded (which is a type of error correction (Elias, 1955) coding). This adds redundancy by producing each output bit as a function of the current and previous inputs. The goal is to enable error correction at the receiver without retransmission. After encoding, the output is interleaved. It is reordered across time so that localized bit corruption does not overwhelm any one frame. These operations are deterministic and standardized. Their result is a longer, structured bitstream with predictable relationships between positions.

At this point, the data must be encrypted, but without affecting their size or timing. Each burst has a fixed payload size, and must be transmitted precisely at its assigned interval. This rules out modes that expand input or require buffering; encryption must operate in place with no change to length or alignment. GSM therefore uses a stream cipher (Vernam, 1926): a keystream is generated and extttXORed with the data bit-for-bit, producing ciphertext of equal length and immediate readiness for modulation.

GSM fixes the processing order as: compression $\rightarrow$ error correction $\rightarrow$ interleaving $\rightarrow$ encryption. This sequencing is a deliberate engineering decision. By placing encryption at the end of the stack, the system isolates cryptographic logic from earlier processing

stages. Each module performs a self-contained transformation. This design simplifies implementation — but, as we will see, introduces a vulnerability.

By the time the bitstream reaches the cipher, it is no longer raw data. It has been processed into a rigid format defined by the protocol. Within the 114 ciphered bits per burst this includes:

- *Padding and known link-layer fields:* deterministic bits that fill or structure payloads on certain channels.
- *Error-correction codes:* parity bits computed from public polynomials.
- *Interleaving:* a known permutation applied identically to each block.

Each 114-bit burst contains payload data bracketed by tail, training, and guard intervals of fixed length; those bracket fields are not ciphered. Within the ciphered payload, the bitstream is heavily preprocessed prior to encryption. Bit patterns arising from coding, interleaving, and protocol padding are defined explicitly by the standard and repeat across sessions. The plaintext entering the encryption algorithm is therefore predictable at specific locations. It is drawn from a constrained distribution with high predictability and low entropy in fixed subregions. A passive observer capturing encrypted GSM traffic receives ciphertext derived from partially labeled inputs whose positions and formats are specified in advance by the protocol.

GSM's stream cipher preserves structure in a way that a block cipher with diffusion would not. Because encryption is bitwise XOR, linear relations introduced by coding survive intact in the ciphertext. Consider a concrete example: suppose the channel coding introduces a parity check — a known XOR relation among data bits. After encryption, the corresponding ciphertext bits satisfy the same parity relation among their respective keystream values. An attacker can deduce this constraint without knowing the underlying data.

By collecting multiple ciphertext samples, each reflecting similar patterns but different keystream realizations, the attacker builds a system of equations that gradually reduces the candidate key space. GSM compounds this vulnerability: voice frames are transmitted redundantly across multiple bursts, providing numerous ciphertext instances derived from aligned inputs. Repeated encipherment of predictable structure with the same key makes the keystream a target for mathematical reconstruction.

This vulnerability was exploited explicitly in the work of Eli Biham, Elad Barkan, and Nathan Keller. In 2003, they demonstrated a ciphertext-only attack against A5/2 capable of recovering the full 64-bit session key in under one second, using multiple frames of intercepted communication from control channels such as SACCH. The attack made no assumptions about plaintext content beyond its adherence to GSM's format. The weakness resulted from applying error correction and interleaving before encryption, allowing algebraic methods to exploit the resulting regularity. The attack combined brute-force enumeration of the cipher's R4 register (2^{16} possibilities) with solving overdetermined systems of linear equations derived from keystream parity constraints. This required hours of preprocessing and gigabytes of storage but was tractable on standard computing hardware.

In the same year, the authors presented an active attack that used this weakness in A5/2 to compromise A5/1 (a stronger cipher that GSM uses by default). GSM allows the base station to select the cipher for communication. A rogue station can impersonate a valid tower and request a downgrade to A5/2 from a handset that supports it. Once the device complies, the attacker captures the A5/2-encrypted exchange, recovers the session key, and then uses that key to decrypt subsequent bursts sent using A5/1. This is possible because GSM reuses the session key across ciphers during a session. The presence of A5/2 in the cipher suite thus undermines A5/1, regardless of whether the latter is ever explicitly requested by the attacker. Any device that implements A5/2 inherits its vulnerabilities and propagates them to the stronger cipher via shared key state.

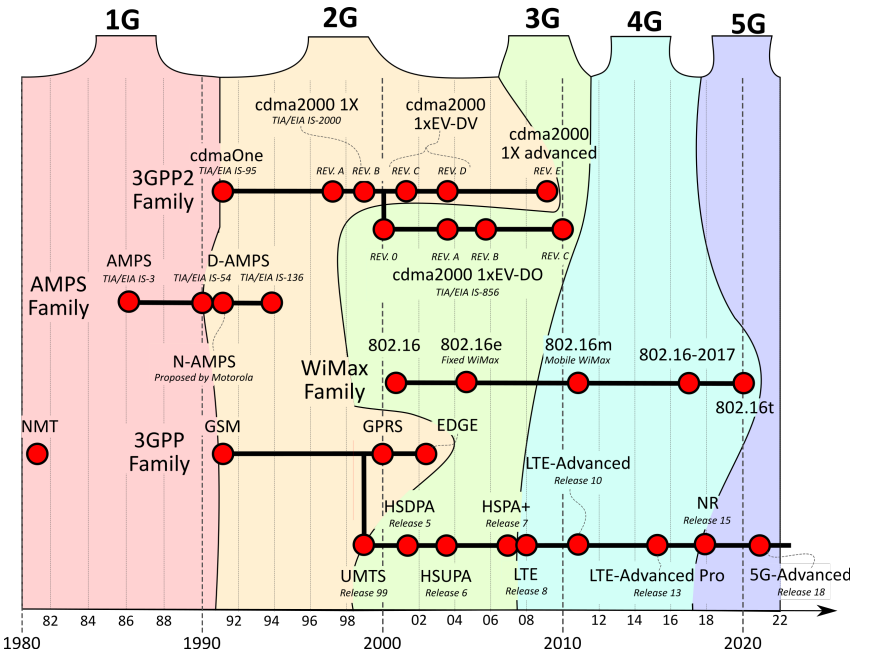
Barkan and Biham continued to refine their attacks. In 2005, they improved known plaintext techniques against A5/1, specifically targeting its irregular initialization procedure. This reduced the computational burden of recovering internal state, particularly in scenarios with limited plaintext exposure. However, their most significant advance came in 2006, when they extended ciphertext-only techniques to A5/1 itself. The approach required far more ciphertext and offline preprocessing than the attack on A5/2, but the principle was similar. By leveraging the publicly known convolutional codes used before encryption, the attackers extracted algebraic relations between ciphertext bits and the keystream. These relations were then used to filter candidate internal states of the cipher's LFSRs (Linear Feedback Shift Registers), narrowing the search space (Golomb, 1955) to feasible dimensions. The complexity of the attack remained high, but it fell within the capabilities of a moderately resourced organization with access to terabyte-scale storage and standard computational infrastructure.

The attack on A5/1 demonstrated that GSM's vulnerability was a result of the interplay between encryption placement, channel configuration, and cipher reuse. GSM's decision to support multiple ciphers without enforcing mutual isolation of key state allowed one weak algorithm to compromise the integrity of the entire suite. Because GSM does not authenticate base stations, handsets cannot verify that cipher selection is legitimate. Any device supporting A5/2 remains exposed to downgrade. Once the session key is recovered through a break of A5/2 — whether using algebraic decoding or parity-based keystream reconstruction — that same key grants access to A5/1-protected content. GSM's cipher suite is therefore not modular. Its effective security is determined not by the strongest cipher in use, but by the weakest that is supported. A5/2's inclusion rendered A5/1 susceptible by transitive failure.

Protocol Assumptions and Personal Entry Point

The weakest points in deployed cryptographic systems are rarely in the mathematics. They are in the layers that surround it: in protocol assumptions, state handling, framing conventions, or timing logic. This is why cryptographic standards are slow to change — not because better ciphers are unavailable, but because known, tested flaws are often safer than untested replacements. The defensive posture of a system is not just algorithmic strength, but mostly accumulated knowledge of how it fails.

I first encountered this issue in a lecture by Eli Biham around 2003. He outlined the GSM vulnerability using nothing but XOR equations, known plaintext segments, and short recurrence relations. This attack did not require knowing the full formalism of block cipher construction or number theory. It showed that security could collapse under regularity exposed by the protocol — and that the analysis of "where" in a system encryption occurs mattered as much as "how."



Cellular network standards and generation timeline, Wikimedia Commons, CC BY-SA 4.0

Toy GSM-Style Frame: How Post-Encoding XOR Leaks Keystream

Frame Layout

We model a simplified SACCH-style control message with *32 information bits* $s_1, \dots, s_{32}$. GSM inserts fixed training bits, padding, and applies forward-error correction; we model this with a minimal layout:

$$\underbrace{1100101101100101}_{\text{training}}\ \underbrace{s_1\ \dots\ s_8}_{\text{data}}$$
$$\underbrace{01101001}_{\text{pad}}\ \underbrace{s_9\ \dots\ s_{16}}_{\text{data}}$$
$$\underbrace{11001011}_{\text{parity}}\ \underbrace{s_{17}\ \dots\ s_{24}}_{\text{data}}\ \underbrace{s_{25}\ \dots\ s_{32}}_{\text{data}}. \quad (1)$$

Exactly **64** bits form the encoder input: 32 unknown information bits and 32 deterministic bits known to the attacker. This scales toward realistic GSM processing where SACCH messages expand from 184 bits to 456 bits.

Redundancy (Mini-Convolutional Code)

Each input bit x_i passes through a toy $(1, 1/2)$ convolutional code with generator polynomials $(1,\ 1 + D)$:

$$y_{i,0} = x_i, \qquad y_{i,1} = x_i \oplus x_{i-1}.$$

Assume zero-state initialization so the bit before each known block is defined. This yields 128 output bits $Y = [Y_0, \dots, Y_{127}]$ where $Y_k = y_{\lfloor k/2 \rfloor, k \bmod 2}$. Every pair satisfies

$$y_{i,1} \ \oplus\ y_{i-1,0} = y_{i,0} \quad \forall i \geq 1, \qquad (2)$$

a parity relation that survives encryption as a linear constraint tying three keystream bits together.

Interleaver

A fixed block interleaver permutes the 128 bits:

$$\pi(i) = (i \bmod 4) \cdot 32 + \left\lfloor \frac{i}{4} \right\rfloor, \qquad (3)$$

a public mapping known to attacker and receiver.

Encryption After Coding

Encryption XORs a keystream $K_0, \dots, K_{127}$ with the permuted code bits:

$$C_i = Y_{\pi(i)} \oplus K_i\ ., \qquad (4)$$

Because XOR preserves length and position, all structure in Y remains in masked form in C.

Ciphertext-Only Attack Sketch

Training leakage: From 32 known training/-pad/parity input bits, the rate 1/2 encoder produces 64 known coded bits (32 from $y_{\cdot,0}$ and 32 from $y_{\cdot,1}$ with state assumption), so the attacker obtains $K_i = C_i \oplus Y_{\pi(i)}$ for 64 positions.

Parity recursion: Using Equation (2), each parity relation after interleaving becomes a linear equation in three ciphertext bits and three keystream bits. The known K_i values seed a sparse linear system over $\mathbb{F}_2$ that propagates to many additional K_j. With *multiple frames*, the system typically determines the full keystream through solving a large linear system or using precomputed tables.

Information bit recovery: With keystream recovered, the attacker inverts the interleaver and convolutional code to extract $s_1, \dots, s_{32}$ from each frame.

Why Encrypt-First Stops the Leak

If encryption preceded coding, the encoder would process $X \oplus K'$ rather than X. Parity relation (2) would then bind unknown values, blocking the attack. Fixed fields would reveal nothing until after decryption.

References:

Barkan, E., Biham, E., Keller, N. (2008). Instant ciphertext-only cryptanalysis of GSM encrypted communication. *J. Cryptology* 21(3):392–429.