RUNSAFE SECURITY

What CWEs Does RunSafe *Stop* the Exploit Of?

Embedded software is highly vulnerable to memory safety flaws that allow attackers to corrupt memory and hijack legitimate software execution. Memory safety vulnerabilities consistently dominate the CWE Top 25 Most Dangerous Software Weaknesses, and they are responsible for many of the most damaging CVEs exploited in the wild.

RunSafe protects applications by preventing exploitation of memory corruption vulnerabilities at runtime without requiring source code, recompilation, or developer rewrites.

EXAMPLES OF CWES RUNSAFE MITIGATES

Core Memory Corruption Vulnerabilities

- **CWE-787** – Out-of-Bounds Write
- **CWE-119** – Improper Restriction of Operations within the Bounds of a Memory Buffer
- **CWE-120** – Classic Buffer Overflow
- **CWE-121** – Stack-based Buffer Overflow
- **CWE-122** – Heap-based Buffer Overflow
- **CWE-124** – Buffer Underwrite (Buffer Underflow)
- **CWE-193** – Off-by-One Error

Why they matter:
These flaws allow attackers to overwrite memory, corrupt control flow, and redirect execution, often leading directly to remote code execution (RCE).

Memory Lifecycle & State Errors

- **CWE-416 –** Use After Free
- **CWE-128 –** Wrap-around Error
- **CWE-190 –** Integer Overflow or Wraparound

Why they matter:
Attackers exploit incorrect memory reuse or arithmetic errors to manipulate memory layout, bypass checks, and gain control over execution paths.

Execution & Input-Driven Exploits

- **CWE-094 –** Improper Control of Code Generation (Code Injection)
- **CWE-020 –** Improper Input Validation

Why they matter:
When untrusted input influences memory or execution flow, attackers can inject payloads that trigger memory corruption and code execution.

Concurrency & Timing Flaws

- **CWE-367 –** Time-of-Check Time-of-Use (TOCTOU) Race Condition

Why they matter:
Race conditions can invalidate security assumptions and expose memory in unintended states and are often chained with memory corruption bugs for full exploitation.

WHAT THIS MEANS FOR EMBEDDED TEAMS

Many high-profile CVEs—across operating systems, embedded firmware, networking software, and industrial systems—trace back to the CWEs listed above.

RunSafe stops exploitation of these CVEs by:
- Preventing memory corruption from altering execution flow
- Blocking attacker-controlled code execution
- Neutralizing exploit techniques even when the vulnerability remains unpatched

This protection applies even when:
- Source code is unavailable
- Patches are delayed or impossible
- Software is legacy, embedded, or third-party