Encoded Pointer Memory Analysis Pattern

Intent

To recognize situations where a pointer stored in memory is not directly usable: its value must be interpreted or transformed before it can be resolved to a valid code or data address.

Context

Stack Trace, Execution Residue, Context Pointer, Historical Information

Common environments:

- Tagged pointers
- ARM64 pointer authentication (PAC)
- Top-Byte-Ignore tagging (AArch64)
- ASLR and relocations that have not yet been applied in the captured memory
- Managed space compressed and metadata-embedded GC pointers
- Objective-C and Swift tagged ISA pointers
- Sanitizers or checking runtimes that add metadata bits

Problem

A pointer in the dump visually appears to be an address, but fails to resolve using normal symbolic or spatial checks; dereferencing its raw value yields an incorrect memory address or a memory error.

Forces

- Performance/security constraints favor encoded pointer formats
- Debugger views often show raw stack/heap
- Encoding schemes vary by platform and compiler
- Hardware PAC may prevent guessing the correct pointer form without a proper decode context

Symptoms

- Pointer value not inside any loaded module or valid virtual address range
- Symbol resolution differs
- Adjacent stack slots look pointer-like, but this one does not
- Backwards disassembly shows an incorrect frame

Resolution Strategies

- Decode PAC
- Canonicalize upper bits
- Strip tags
- Expand bits
- Apply relocation deltas
- Mask metadata

Resulting Context

After correct interpretation, the pointer becomes:

- Resolvable to a target symbol
- Walkable for call-stack reconstruction
- Safe for dereferencing in analysis context
- Enables further analysis

Examples

Software Diagnostics Library

Related Patterns

Invalid Pointer