# CWE Detail – CWE-1299

## Description

The lack of protections on alternate paths to access  
 control-protected assets (such as unprotected shadow registers  
 and other external facing unguarded interfaces) allows an  
 attacker to bypass existing protections to the asset that are  
 only performed against the primary path.

## Extended Description

An asset inside a chip might have access-control  
 protections through one interface. However, if all paths to  
 the asset are not protected, an attacker might compromise  
 the asset through alternate paths. These alternate paths  
 could be through shadow or mirror registers inside the IP  
 core, or could be paths from other external-facing  
 interfaces to the IP core or SoC. Consider an SoC with various interfaces such as UART,  
 SMBUS, PCIe, USB, etc. If access control is implemented for  
 SoC internal registers only over the PCIe interface, then  
 an attacker could still modify the SoC internal registers  
 through alternate paths by coming through interfaces such  
 as UART, SMBUS, USB, etc. Alternatively, attackers might be able to bypass  
 existing protections by exploiting unprotected, shadow  
 registers. Shadow registers and mirror registers typically  
 refer to registers that can be accessed from multiple  
 addresses. Writing to or reading from the aliased/mirrored  
 address has the same effect as writing to the address of  
 the main register. They are typically implemented within an  
 IP core or SoC to temporarily hold certain data. These data  
 will later be updated to the main register, and both  
 registers will be in synch. If the shadow registers are not  
 access-protected, attackers could simply initiate  
 transactions to the shadow registers and compromise system  
 security.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2022-38399: Missing protection mechanism on serial connection allows for arbitrary OS command execution.

**•** CVE-2020-9285: Mini-PCI Express slot does not restrict direct memory access.

**•** CVE-2020-8004: When the internal flash is protected by blocking access on the Data Bus (DBUS), it can still be indirectly accessed through the Instruction Bus (IBUS).

**•** CVE-2017-18293: When GPIO is protected by blocking access  
 to corresponding GPIO resource registers,  
 protection can be bypassed by writing to the  
 corresponding banked GPIO registers instead.

**•** CVE-2020-15483: monitor device allows access to physical UART debug port without authentication

## Related Attack Patterns (CAPEC)

* CAPEC-457
* CAPEC-554

## Attack TTPs

**•** T1091: Replication Through Removable Media (Tactics: lateral-movement, initial-access)

**•** T1092: Communication Through Removable Media (Tactics: command-and-control)

## Modes of Introduction

**•** Architecture and Design: N/A

**•** Implementation: N/A

## Common Consequences

**•** Impact: Modify Memory, Read Memory, DoS: Resource Consumption (Other), Execute Unauthorized Code or Commands, Gain Privileges or Assume Identity, Alter Execution Logic, Bypass Protection Mechanism, Quality Degradation — Notes:

## Potential Mitigations

**•** Requirements: Protect assets from accesses against all potential interfaces and alternate paths. (Effectiveness: Defense in Depth)

**•** Architecture and Design: Protect assets from accesses against all potential interfaces and alternate paths. (Effectiveness: Defense in Depth)

**•** Implementation: Protect assets from accesses against all potential interfaces and alternate paths. (Effectiveness: Defense in Depth)

## Applicable Platforms

**•** None (Class: Not Language-Specific, Prevalence: Undetermined)

## Demonstrative Examples

**•** The bugged line of code is repeated in the Bad  
 example above. The weakness arises from the fact that the  
 SECURE\_ME register can be modified by writing to the  
 shadow register COPY\_OF\_SECURE\_ME. The address of  
 COPY\_OF\_SECURE\_ME should also be included in the check.  
 That buggy line of code should instead be replaced as  
 shown in the Good Code Snippet below.