Req ID	Requirement name	Supported by CIP	Need application support	Need HW solution	Status if supported by CIP	IEC-62443-4-2 tests reference	CIP recommendation
CR-4.1	Information confidentiality - at rest	TRUE	TRUE	FALSE	In-progress to add packages in progress	None Investigation	CIP platform complies to this requirementCIP users need to utilize the capabilities provided by CIP and encrypt data at rest
CR-4.2	Information persistence	FALSE	TRUE	FALSE	N.A.	None	CIP platform does not support this requirementCIP member needs to provide application which can achieve information persistance, CIP provides underneath packages to meet this requirement
CR-4.2 RE(1)	Erase of shared memory resources	TRUE	TRUE	FALSE	Completed added package ipcrm	Added https://gitlab.com/cip-project/cip-testing/cip-security-tests/-/tree/master/iec-security-tests/singlenode-testcases/TC_CR4.2-RE1_1	CIP meets this requirement by adding the packages.CIP users should use CIP capabilities and make sure volatile memory contents are deleted as well as any data buffer allocated by application is reset before deleting or allocating to other processes.
CR-4.2 RE(2)	Erase verification	TRUE	TRUE	FALSE	Completed added package ipcrm	Added https://gitlab.com/cip-project/cip-testing/cip-security-tests/-/tree/master/iec-security-tests/singlenode-testcases/TC_CR4.2-RE2_1	CIP users should support to confirm and acknowledge the erased contents are not accessible.
CR-4.3	Use of cryptography	TRUE	FALSE	FALSE	Completed added openssl package	Refer openssl tests for CR-1.9	Refer details of this requirement in Use of Cryptography document (https://gitlab.com/cip-project/cip-documents/-/blob/master/process/use_of_cryptography-cr-4.3.md)

Additional notes

Regarding CR-4.2 RE(1) For in-memory applications, APIs can be used, e.g.

- from OpenSSL `OPENSSL_cleanse` This function does not return any success status and is guaranteed to work.

If users are required to store sensitive data in files, the following approaches shall be chosen if possible:

- Anonymous files (`memfd_create`) shall be used so that a file with secret content will never exist os disk. The deletion is guaranteed by the kernel.
- Secrets shall only be stored in a specially mounted `ramfs` so that secrets will never be written to persistent memory. (`tmpfs` might be written to swap, so it is not desired here.).

If storing sensitive data on persistent storage is inevitable, `shred` shall be used for deletion. Note that this tool is only appropriate for (non-SMR) hard disks but not for managed flash. No sure approach for SSD and SD cards exists, unmanaged flash could allow it, but no software solution currently exists. `shred` gives a return value showing if the operation was successful or not.

An extended file system flag `FS_SECRM_FL` exists which should mark files as to be deleted in a secure manner, but it is nowhere implemented in the kernel. Implementing it for ramfs and ubifs could be beneficial.

For the meanwhile, if sensitive data is written to persistent memory, it shall only be written in an encrypted manner so that deleting the key would be sufficient to render the data unreadable. Note that the key itself must be stored on a volume that can be deleted securely.

Memory provided provided by the kernel is always zero-initialized (unless especially enabled by e.g. `CONFIG_MMAP_ALLOW_UNINITIALIZED`) and thus, is no concern of information leakage from one program to another.