New issue

Plausible deniability #1215

Open **Plausible deniability** #1215

I found an article that describes VeraCrypt volume detection. Maybe some attention should be paid to the possibility of plausible deniability?

https://www.raedts.biz/forensics/detecting-truecrypt-veracrypt-volumes/

The points about entropy and lack of file signature are non-issues. There is no way to believably fake a file signature of another filetype without actually embedding your container into another file (see Steganography with TC/VC volumes).

The file container being divisible by 1024 is a valid criticism, and I don't know if it is an actual limitation of how the encryption works with block sizes, or rather just a side effect of kilobytes being the smallest unit you can give the size as. I tried a workaround and appended random bytes at the end of the file container, then changed both outer and hidden container passwords to force rewrite of embedded backup header. The container seemed to work fine and was no longer divisible by 1024. Seemed to work for this case, but I can't guarantee anything.

👍 1

Thank you both for your detailed feedback and the insights you've shared.

First, let me clarify the objective of VeraCrypt. The primary goal of VeraCrypt is not to hide the presence of encryption, but to protect the encrypted content and ensure its integrity. Plausible deniability in the context of VeraCrypt file containers pertains mainly to the presence of hidden volumes within file containers, not to the presence of encryption itself. When dealing with entire disks or partitions, plausible deniability can extend to encryption, since one can argue that the disk or partition has been wiped and is not encrypted.

Steganography, as mentioned by **@Jertzukka**, is indeed an approach to conceal the presence of encrypted data. However, this is outside the core purpose of VeraCrypt. It's a separate field of data protection.

Regarding the container size being divisible by 1024: The size of a file container itself doesn't pose a restriction for VeraCrypt, as the volume header encodes the effective volume size that will be utilized. Technically, appending arbitrary data to a file container will not impede its usability. But it's essential to recreate the backup header after such a modification, as **@Jertzukka** correctly did. On Windows, however, there's a nuance. We access files using offsets aligned to disk sector sizes. If the offset of the backup header is not a multiple of 512 bytes, it will introduce errors.

The 1024-byte multiple constraint is mostly a result of the user interface design, which uses KiB as the smallest selectable unit. In Linux, for instance, it is already possible to create a file container whose size isn't a multiple of 1024 through the command line. For example:
`veracrypt -t -c --size=123127777 --password=test --volume-type=normal --filesystem=FAT --pim=0 --hash=sha256 --encryption=aes --non-interactive test.hc`
Here the size is 123127777 bytes which is not a multiple of 1024.

A potential enhancement could involve appending a random number of bytes, ranging from 0 to 1023, to the file container during its creation. Yet, to ensure compatibility with Windows, we will have to think about how we handle the alignment requirement for the backup header offset. One idea is to always create the backup header at an aligned offset, regardless of the actual file container size. I will consider this suggestion and its implications.

👍 3

idrassi added **enhancement** on Oct 1, 2023

idrassi self-assigned this on Oct 1, 2023

Veracrypt can be loaded via a backup header embedded in the volume;
Therefore, I tried to replace the first 128KB of the encrypted volume, disguised as mp4, iso, etc., which can also be done as a reasonable prevarication;
Choose to embed backup header when loading, although troublesome, can the author add this feature (disguise encryption header)? **@idrassi**

Metadata

Assignees

idrassi

Labels

enhancement