**GnuTLS**

GnuTLS is a secure communications library that implements and supports the SSL, TLS, and DTLS protocols and technologies. It includes a simple C language application programming interface (API) for accessing secure communication protocols, as well as APIs for parsing and writing X. 509, PKCS #12, and other necessary structures. GnuTLS was designed to provide security against attacks like eavesdropping, tampering and damaging the message integrity. It provides secure communication for the backend and works with all base linux libraries.

GnuTLS is made up of 3 independent parts, TLS protocol part, Certificate part and cryptographic backend part. The "TLS protocol portion" is the actual protocol implementation, which is completed entirely within the GnuTLS library. Verification functionality and certificate parsing is provided by certificate part, which gets its functionality from libtasn1 library. Data confidentiality and integrity is provided by cryptographic part which obtains its functionality from nettle and gmplib libraries. Following are some of the important features of GnuTLS library:

* Support for TLS 1.3, TLS 1.2, TLS 1.1, TLS 1.0 and optionally SSL 3.0 protocols.
* Support for Datagram TLS 1.0 and 1.2.
* Support for handling and verification of X.509 certificates.
* Support for password authentication using TLS-SRP

GnuTLS follows a development cycle in which even minor version numbers represent a stable release and odd minor version numbers represent a development release. Nettle and gmplib are the base libraries for installation of GnuTLS, before installing GnuTLS installation of libraries is necessary.

Following are some of the useful commands:

* Enable/disable-srp-authentication
* Enable/disable -psk-authentication
* Enable/disable -anon-authentication
* Enable/disable -dhe
* Enable/disable -ecdhe
* Enable/disable -openssl-compatibility
* Enable/disable -dtls-srtp-support
* Enable/disable -alpn-support
* Enable/disable -heartbeat-support

**Functionality of GunTLS:**

Diagram

Description automatically generated

There is a global state for read-only that starts once the global initialization function starts. This global structure contains memory allocation functions, ASN.1 parser structures, and, depending on the system's CPU, references to hardware accelerated encryption functions. There is no major change in global GnuTLS function except one change and that is decentralization function which must be called once all the programs are permanently furnished to free the allocated memory.

Authentication methords like certificate authentication are verified through credentials structure. These structure stores the parameter like private keys, certificates and other information that is used to proof the identity of user.

Session state and all other relevant information about the sessions are stored in session database. Each session has unique session id which is used for session identification purpose. After successful session handshake, each GnuTLS session call the backend function to perform the appropriate functionality. The server examines the session database immediately after receiving the client hello, and if the session ID supplied by the client matches a stored session, the saved session is retrieved, and the new session is a resumed one with the same session ID as the previous one.

By design, the GnuTLS library is thread safe, which means that library objects, such as TLS sessions, can be safely partitioned among threads as long as a single thread accesses a single object. This is enough to support a server that supports many sessions per thread. Thread-safe read-only access to objects, such as credentials holding structures, is also possible.

**Error Codes and Descriptions**

The following are some of the error codes with description:

|  |  |  |
| --- | --- | --- |
| **Error code** | **Error** | **Description** |
| 0 | GNUTLS\_E\_SUCCESS | Success |
| -3 | GNUTLS\_E\_UNKNOWN\_COMPRESSION\_ALGORITHM | Unable to negotiate the supported compression method |
| -6 | GNUTLS\_E\_UNKNOWN\_CIPHER\_TYPE | Incompatible cipher type |
| -7 | GNUTLS\_E\_LARGE\_PACKET | Large packet size |
| -8 | GNUTLS\_E\_UNSUPPORTED\_VERSION\_PACKET | Packet with illegal or unsupported version was received |
| -9 | GNUTLS\_E\_UNEXPECTED\_PACKET\_LENGTH | Error decoding received TLS packet |
| -10 | GNUTLS\_E\_INVALID\_SESSION | The specified session has been invalidated for some reason |

**Vulnerabilities in GnuTLS:**

The following are some of the common GnuTLS vulnerabilities:

1. CVE-2022-2509 - Double Free:

Operating system allocates some memory to store data temporarily and free it after some time and the same memory block is allocated by operating system to programmer to for temporary storage purposes. Ideally, the memory allocation software will detect that the block no longer belongs to the section of the program that is "returning" it, will recognize that the block has already been recycled, and will refuse to deallocate it a second time, avoiding the hazards of "freeing" it again. But if the memory allocator is not properly handled and it hands out the same memory spot to an other program at exactly the same loaned out spot then this causes great problem. Attacker can take advantage of this bug and you could end up with same memory chunk exploited with the two parts of same program. This vulnerability allow attacker to run malicious code on the target system This bug existed in certificate verification code. The solution to mitigate this vulnerability is to upgrade to latest version of GnuTLS

1. CVE-2020-11501 - Incorrect Cryptographic Function:

Before GnuTLS version 3.6.13, version used incorrect cryptographic function in DTLS. Instead of using random value DTLS client were using 32 '\0' bytes value, this made easier for attacker to run brute force as there was no randomness in in DTLS negotiation. If the attacker gets successful in getting this number they can significantly damage the system files and corrupt the system. This vulnerability could be mitigated by updating the GnuTLS version to any version after 3.6.13

**References**:

<https://gnutls.org/manual/gnutls.html#Introduction-to-GnuTLS>

<https://gnutls.org/manual/gnutls.html#How-to-use-GnuTLS-in-applications>

<https://nakedsecurity.sophos.com/2022/08/01/gnutls-patches-memory-mismanagement-bug-update-now/>

<https://gnutls.org/security-new.html>

<https://www.cvedetails.com/cve/CVE-2020-11501/>

<https://www.debian.org/security/2020/dsa-4652>

<https://access.redhat.com/security/cve/cve-2019-3836>