PitStop Library Container 2319

Release notes

What’s new

In this version, some major improvements for your production are delivered. Transaction queuing makes sure your production environment keeps on processing jobs without being affected by server upgrades. We also fixed a lot of vulnerability issues, as security is very important for our customers. We also fixed some bugs, which could arise in high volume environments. So great improvements and we are already looking forward to our next release in 6 weeks (June 20th).

1. Transaction queuing.

When there is a license server upgrade, it could happen that the license server is down for a short period of time (+/- 10 minutes). In the past, this causes PitStop Library Container to let jobs that were processed during this period fail.
In this version, countermeasures are implemented to let these jobs succeed, queue the pending transaction and send the transactions when the license server is back online. As such, server downtime has no impact on PitStop Library Container processing anymore.

2. Vulnerability issues fixed

In this version, a series of vulnerability issues were fixed. You can find the complete list here:

High	A vulnerability exists in curl <7.87.0 HSTS check that could be bypassed to trick it to keep using HTTP. Using its HSTS support, curl can be instructed to use HTTPS instead of using an insecure clear-text HTTP step even when HTTP is provided in the URL. However, the HSTS mechanism could be bypassed if the host name in the given URL first uses IDN characters that get replaced to ASCII counterparts as part of the IDN conversion. Like using the character UTF-8 U+3002 (IDEOGRAPHIC FULL STOP) instead of the common ASCII full stop (U+002E) `.`. Then in a subsequent request, it does not detect the HSTS state and makes a clear text transfer. Because it would store the info IDN encoded but look for it IDN decoded.
Medium	A use after free vulnerability exists in curl <7.87.0. Curl can be asked to *tunnel* virtually all protocols it supports through an HTTP proxy. HTTP proxies can (and often do) deny such tunnel operations. When getting denied to tunnel the specific protocols SMB or TELNET, curl would use a heap-allocated struct after it had been freed, in its transfer shutdown code path.
High	libexpat before 2.4.9 has a use-after-free in the doContent function in xmlparse.c.
High	In libexpat through 2.4.9, there is a use-after free caused by overeager destruction of a shared DTD in XML_ExternalEntityParserCreate in out-of-memory situations.
High	Heimdal before 7.7.1 allows attackers to cause a NULL pointer dereference in a SPNEGO acceptor via a preferred_mech_type of GSS_C_NO_OID and a nonzero initial_response value to send_accept.
Medium	A heap-based buffer overflow vulnerability was found in Samba within the GSSAPI unwrap_des() and unwrap_des3() routines of Heimdal. The DES and Triple-DES decryption routines in the Heimdal GSSAPI library allow a length-limited write buffer overflow on malloc() allocated memory when presented with a maliciously small packet. This flaw allows a remote user to send specially crafted malicious data to the application, possibly resulting in a denial of service (DoS) attack.

High	PAC parsing in MIT Kerberos 5 (aka krb5) before 1.19.4 and 1.20.x before 1.20.1 has integer overflows that may lead to remote code execution (in KDC, kadmind, or a GSS or Kerberos application server) on 32-bit platforms (which have a resultant heap-based buffer overflow), and cause a denial of service on other platforms. This occurs in krb5_pac_parse in lib/krb5/krb/pac.c. Heimdal before 7.7.1 has "a similar bug."
Critical	Heimdal before 7.7.1 allows remote attackers to execute arbitrary code because of an invalid free in the ASN.1 codec used by the Key Distribution Center (KDC).
High	A vulnerability exists in curl <7.87.0 HSTS check that could be bypassed to trick it to keep using HTTP. Using its HSTS support, curl can be instructed to use HTTPS instead of using an insecure clear-text HTTP step even when HTTP is provided in the URL. However, the HSTS mechanism could be bypassed if the host name in the given URL first uses IDN characters that get replaced to ASCII counterparts as part of the IDN conversion. Like using the character UTF-8 U+3002 (IDEOGRAPHIC FULL STOP) instead of the common ASCII full stop (U+002E) `.`. Then in a subsequent request, it does not detect the HSTS state and makes a clear text transfer. Because it would store the info IDN encoded but look for it IDN decoded.
Medium	A use after free vulnerability exists in curl <7.87.0. Curl can be asked to *tunnel* virtually all protocols it supports through an HTTP proxy. HTTP proxies can (and often do) deny such tunnel operations. When getting denied to tunnel the specific protocols SMB or TELNET, curl would use a heap-allocated struct after it had been freed, in its transfer shutdown code path.
High	libexpat before 2.4.9 has a use-after-free in the doContent function in xmlparse.c.
High	In libexpat through 2.4.9, there is a use-after free caused by overeager destruction of a shared DTD in XML_ExternalEntityParserCreate in out-of-memory situations.
Medium	It was discovered that Kerberos incorrectly handled certain S4U2Self requests. An attacker could possibly use this issue to cause a denial of service.
High	PAC parsing in MIT Kerberos 5 (aka krb5) before 1.19.4 and 1.20.x before 1.20.1 has integer overflows that may lead to remote code execution (in KDC, kadmind, or a GSS or Kerberos application server) on 32-bit platforms (which have a resultant heap-based buffer overflow), and cause a denial of service on other platforms. This occurs in krb5_pac_parse in lib/krb5/krb/pac.c. Heimdal before 7.7.1 has "a similar bug."
High	Heimdal before 7.7.1 allows attackers to cause a NULL pointer dereference in a SPNEGO acceptor via a preferred_mech_type of GSS_C_NO_OID and a nonzero initial_response value to send_accept.
Medium	A heap-based buffer overflow vulnerability was found in Samba within the GSSAPI unwrap_des() and unwrap_des3() routines of Heimdal. The DES and Triple-DES decryption routines in the Heimdal GSSAPI library allow a length-limited write buffer overflow on malloc() allocated memory when presented with a maliciously small packet. This flaw allows a remote user to send specially crafted malicious data to the application, possibly resulting in a denial of service (DoS) attack.
High	PAC parsing in MIT Kerberos 5 (aka krb5) before 1.19.4 and 1.20.x before 1.20.1 has integer overflows that may lead to remote code execution (in KDC, kadmind, or a GSS or Kerberos application server) on 32-bit platforms (which have a resultant heap-based buffer overflow), and cause a denial of service on other platforms. This occurs in krb5_pac_parse in lib/krb5/krb/pac.c. Heimdal before 7.7.1 has "a similar bug."
Critical	Heimdal before 7.7.1 allows remote attackers to execute arbitrary code because of an invalid free in the ASN.1 codec used by the Key Distribution Center (KDC).
High	Heimdal before 7.7.1 allows attackers to cause a NULL pointer dereference in a SPNEGO acceptor via a preferred_mech_type of GSS_C_NO_OID and a nonzero initial_response value to send_accept.
Medium	A heap-based buffer overflow vulnerability was found in Samba within the GSSAPI unwrap_des() and unwrap_des3() routines of Heimdal. The DES and Triple-DES decryption routines in the Heimdal GSSAPI library allow a length-limited write buffer overflow on malloc() allocated memory when presented with a

	maliciously small packet. This flaw allows a remote user to send specially crafted malicious data to the application, possibly resulting in a denial of service (DoS) attack.
High	PAC parsing in MIT Kerberos 5 (aka krb5) before 1.19.4 and 1.20.x before 1.20.1 has integer overflows that may lead to remote code execution (in KDC, kadmind, or a GSS or Kerberos application server) on 32-bit platforms (which have a resultant heap-based buffer overflow), and cause a denial of service on other platforms. This occurs in krb5_pac_parse in lib/krb5/krb/pac.c. Heimdal before 7.7.1 has "a similar bug."
Critical	Heimdal before 7.7.1 allows remote attackers to execute arbitrary code because of an invalid free in the ASN.1 codec used by the Key Distribution Center (KDC).
High	Heimdal before 7.7.1 allows attackers to cause a NULL pointer dereference in a SPNEGO acceptor via a preferred_mech_type of GSS_C_NO_OID and a nonzero initial_response value to send_accept.
Medium	A heap-based buffer overflow vulnerability was found in Samba within the GSSAPI unwrap_des() and unwrap_des3() routines of Heimdal. The DES and Triple-DES decryption routines in the Heimdal GSSAPI library allow a length-limited write buffer overflow on malloc() allocated memory when presented with a maliciously small packet. This flaw allows a remote user to send specially crafted malicious data to the application, possibly resulting in a denial of service (DoS) attack.
High	PAC parsing in MIT Kerberos 5 (aka krb5) before 1.19.4 and 1.20.x before 1.20.1 has integer overflows that may lead to remote code execution (in KDC, kadmind, or a GSS or Kerberos application server) on 32-bit platforms (which have a resultant heap-based buffer overflow), and cause a denial of service on other platforms. This occurs in krb5_pac_parse in lib/krb5/krb/pac.c. Heimdal before 7.7.1 has "a similar bug."
Critical	Heimdal before 7.7.1 allows remote attackers to execute arbitrary code because of an invalid free in the ASN.1 codec used by the Key Distribution Center (KDC).
Critical	Libksba before 1.6.3 is prone to an integer overflow vulnerability in the CRL signature parser.
Critical	The Linux-PAM package before 1.5.2-6.1 for openSUSE Tumbleweed allows authentication bypass for SSH logins. The pam_access.so module doesn't correctly restrict login if a user tries to connect from an IP address that is not resolvable via DNS. In such conditions, a user with denied access to a machine can still get access. NOTE: the relevance of this issue is largely limited to openSUSE Tumbleweed and openSUSE Factory; it does not affect Linux-PAM upstream.
High	SQLite 1.0.12 through 3.39.x before 3.39.2 sometimes allows an array-bounds overflow if billions of bytes are used in a string argument to a C API.
Informational	A timing based side channel exists in the OpenSSL RSA Decryption implementation which could be sufficient to recover a plaintext across a network in a Bleichenbacher style attack. To achieve a successful decryption an attacker would have to be able to send a very large number of trial messages for decryption. The vulnerability affects all RSA padding modes: PKCS#1 v1.5, RSA-OEAP and RSASVE. For example, in a TLS connection, RSA is commonly used by a client to send an encrypted pre-master secret to the server. An attacker that had observed a genuine connection between a client and a server could use this flaw to send trial messages to the server and record the time taken to process them. After a sufficiently large number of messages the attacker could recover the pre-master secret used for the original connection and thus be able to decrypt the application data sent over that connection.
Informational	The function PEM_read_bio_ex() reads a PEM file from a BIO and parses and decodes the "name" (e.g. "CERTIFICATE"), any header data and the payload data. If the function succeeds then the "name_out", "header" and "data" arguments are populated with pointers to buffers containing the relevant decoded data. The caller is responsible for freeing those buffers. It is possible to construct a PEM file that results in 0 bytes of payload data. In this case PEM_read_bio_ex() will return a failure code but will populate the header argument with a pointer to a buffer that has already been freed. If the caller also frees this buffer then a double free will occur. This will most likely lead to a crash. This could be exploited by an attacker who has the ability to supply malicious PEM files for parsing to achieve a denial of service attack. The functions

	PEM_read_bio() and PEM_read() are simple wrappers around PEM_read_bio_ex() and therefore these functions are also directly affected.
Informational	The public API function BIO_new_NDEF is a helper function used for streaming ASN.1 data via a BIO. It is primarily used internally to OpenSSL to support the SMIME, CMS and PKCS7 streaming capabilities, but may also be called directly by end user applications. The function receives a BIO from the caller, prepends a new BIO_f_asn1 filter BIO onto the front of it to form a BIO chain, and then returns the new head of the BIO chain to the caller. Under certain conditions, for example if a CMS recipient public key is invalid, the new filter BIO is freed and the function returns a NULL result indicating a failure. However, in this case, the BIO chain is not properly cleaned up and the BIO passed by the caller still retains internal pointers to the previously freed filter BIO. If the caller then goes on to call BIO_pop() on the BIO then a use-after-free will occur. This will most likely result in a crash.
Informational	An invalid pointer dereference on read can be triggered when an application tries to load malformed PKCS7 data with the d2i_PKCS7(), d2i_PKCS7_bio() or d2i_PKCS7_fp() functions. The result of the dereference is an application crash which could lead to a denial of service attack. The TLS implementation in OpenSSL does not call this function however third party applications might call these functions on untrusted data.
Informational	An invalid pointer dereference on read can be triggered when an application tries to check a malformed DSA public key by the EVP_PKEY_public_check() function. This will most likely lead to an application crash. This function can be called on public keys supplied from untrusted sources which could allow an attacker to cause a denial of service attack. The TLS implementation in OpenSSL does not call this function but applications might call the function if there are additional security requirements imposed by standards such as FIPS 140-3.
Informational	There is a type confusion vulnerability relating to X.400 address processing inside an X.509 GeneralName. X.400 addresses were parsed as an ASN1_STRING but the public structure definition for GENERAL_NAME incorrectly specified the type of the x400Address field as ASN1_TYPE. This field is subsequently interpreted by the OpenSSL function GENERAL_NAME_cmp as an ASN1_TYPE rather than an ASN1_STRING. When CRL checking is enabled (i.e. the application sets the X509_V_FLAG_CRL_CHECK flag), this vulnerability may allow an attacker to pass arbitrary pointers to a memcmp call, enabling them to read memory contents or enact a denial of service. In most cases, the attack requires the attacker to provide both the certificate chain and CRL, neither of which need to have a valid signature. If the attacker only controls one of these inputs, the other input must already contain an X.400 address as a CRL distribution point, which is uncommon.
Informational	A NULL pointer can be dereferenced when signatures are being verified on PKCS7 signed or signedAndEnveloped data. In case the hash algorithm used for the signature is known to the OpenSSL library but the implementation of the hash algorithm is not available the digest initialization will fail. There is a missing check for the return value from the initialization function which later leads to invalid usage of the digest API most likely leading to a crash. The unavailability of an algorithm can be caused by using FIPS enabled configuration of providers or more commonly by not loading the legacy provider. PKCS7 data is processed by the SMIME library calls and also by the time stamp (TS) library calls. The TLS implementation in OpenSSL does not call these functions however third party applications would be affected if they call these functions to verify signatures on untrusted data.
High	Heimdal is an implementation of ASN.1/DER, PKIX, and Kerberos. Versions prior to 7.7.1 are vulnerable to a denial of service vulnerability in Heimdal's PKI certificate validation library, affecting the KDC (via PKINIT) and kinit (via PKINIT), as well as any third-party applications using Heimdal's libhx509. Users should upgrade to Heimdal 7.7.1 or 7.8. There are no known workarounds for this issue.
Medium	shadow: TOCTOU (time-of-check time-of-use) race condition when copying and removing directory trees
Critical	The Keccak XKCP SHA-3 reference implementation before fdc6fef has an integer overflow and resultant buffer overflow that allows attackers to execute arbitrary code or eliminate expected cryptographic properties. This occurs in the sponge function interface.

High	An issue was discovered in Python before 3.11.1. An unnecessary quadratic algorithm exists in one path when processing some inputs to the IDNA (RFC 3490) decoder, such that a crafted, unreasonably long name being presented to the decoder could lead to a CPU denial of service. Hostnames are often supplied by remote servers that could be controlled by a malicious actor; in such a scenario, they could trigger excessive CPU consumption on the client attempting to make use of an attacker-supplied supposed hostname. For example, the attack payload could be placed in the Location header of an HTTP response with status code 302. A fix is planned in 3.11.1, 3.10.9, 3.9.16, 3.8.16, and 3.7.16.

Creating a vulnerability report is now part of our standard work to ensure vulnerabilities are detected as soon as they are introduced.

Bugfixes

The following bugs have been reported and fixed in this version:

ENFCLOUD-826	PitStop reported job completed, but PLC was actually still working on it.
ENFCLOUD-827	PitStop Error: document is already being processed
ENFCLOUD-568	The job ID in the response of progress API is different than the job ID we sent in the request