esp-firmware Scan Report

Project Name	esp-firmware
Scan Start	Thursday, June 14, 2018 2:54:34 PM
Preset	OWASP TOP 10 - 2010
Scan Time	00h:01m:35s
Lines Of Code Scanned	2374
Files Scanned	8
Report Creation Time	Thursday, June 14, 2018 2:56:51 PM
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15
Team	CxServer
Checkmarx Version	8.7.0
Scan Type	Full
Source Origin	LocalPath
Density	4/100 (Vulnerabilities/LOC)
Visibility	Public

Filter Settings

Severity

Included: High, Medium, Low, Information

Excluded: None

Result State

Included: Confirmed, Not Exploitable, To Verify, Urgent, Proposed Not Exploitable

Excluded: None

Assigned to

Included: All

Categories

Included:

Uncategorized	All
Custom	All
PCI DSS v3.2	All
OWASP Top 10 2013	All
FISMA 2014	All
NIST SP 800-53	All
OWASP Top 10 2017	All
OWASP Mobile Top 10 2016	All

Excluded:

Uncategorized	None
Custom	None
PCI DSS v3.2	None
OWASP Top 10 2013	None
FISMA 2014	None

NIST SP 800-53	None
OWASP Top 10 2017	None
OWASP Mobile Top 10 2016	None

Results Limit

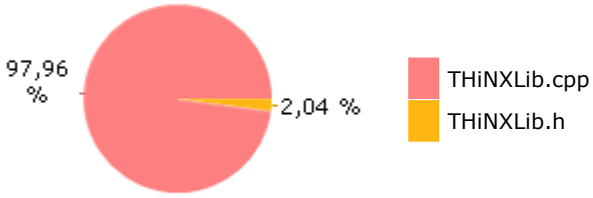
Results limit per query was set to 50

Selected Queries

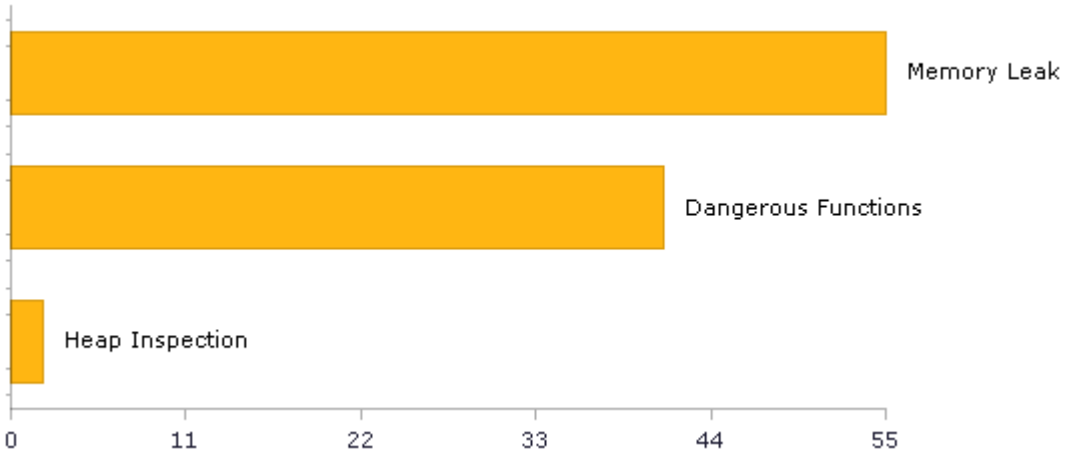
Selected queries are listed in Result Summary

Result Summary

Most Vulnerable Files



Top 5 Vulnerabilities



Scan Summary - OWASP Top 10 2017

Further details and elaboration about vulnerabilities and risks can be found at: OWASP Top 10 2017

Category	Threat Agent	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection*	App. Specific	EASY	COMMON	EASY	SEVERE	App. Specific	0	0
A2-Broken Authentication*	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A3-Sensitive Data Exposure*	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	2	2
A4-XML External Entities (XXE)	App. Specific	AVERAGE	COMMON	EASY	SEVERE	App. Specific	0	0
A5-Broken Access Control*	App. Specific	AVERAGE	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A6-Security Misconfiguration *	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A7-Cross-Site Scripting (XSS)	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A8-Insecure Deserialization	App. Specific	DIFFICULT	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A9-Using Components with Known Vulnerabilities*	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	MODERATE	App. Specific	41	41
A10-Insufficient Logging & Monitoring	App. Specific	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	App. Specific	0	0

* Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - OWASP Top 10 2013

Further details and elaboration about vulnerabilities and risks can be found at: OWASP Top 10 2013

Category	Threat Agent	Attack Vectors	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection*	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	AVERAGE	SEVERE	ALL DATA	0	0
A2-Broken Authentication and Session Management*	EXTERNAL, INTERNAL USERS	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	AFFECTED DATA AND FUNCTIONS	0	0
A3-Cross-Site Scripting (XSS)	EXTERNAL, INTERNAL, ADMIN USERS	AVERAGE	VERY WIDESPREAD	EASY	MODERATE	AFFECTED DATA AND SYSTEM	0	0
A4-Insecure Direct Object References*	SYSTEM USERS	EASY	COMMON	EASY	MODERATE	EXPOSED DATA	0	0
A5-Security Misconfiguration *	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	EASY	MODERATE	ALL DATA AND SYSTEM	0	0
A6-Sensitive Data Exposure*	EXTERNAL, INTERNAL, ADMIN USERS, USERS BROWSERS	DIFFICULT	UNCOMMON	AVERAGE	SEVERE	EXPOSED DATA	2	2
A7-Missing Function Level Access Control*	EXTERNAL, INTERNAL USERS	EASY	COMMON	AVERAGE	MODERATE	EXPOSED DATA AND FUNCTIONS	0	0
A8-Cross-Site Request Forgery (CSRF)	USERS BROWSERS	AVERAGE	COMMON	EASY	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0
A9-Using Components with Known Vulnerabilities*	EXTERNAL USERS, AUTOMATED TOOLS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	41	41
A10-Unvalidated Redirects and Forwards	USERS BROWSERS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0

***** Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - PCI DSS v3.2

Category	Issues Found	Best Fix Locations
PCI DSS (3.2) - 6.5.1 - Injection flaws - particularly SQL injection	0	0
PCI DSS (3.2) - 6.5.2 - Buffer overflows*	0	0
PCI DSS (3.2) - 6.5.3 - Insecure cryptographic storage	0	0
PCI DSS (3.2) - 6.5.4 - Insecure communications	0	0
PCI DSS (3.2) - 6.5.5 - Improper error handling*	0	0
PCI DSS (3.2) - 6.5.7 - Cross-site scripting (XSS)	0	0
PCI DSS (3.2) - 6.5.8 - Improper access control	0	0
PCI DSS (3.2) - 6.5.9 - Cross-site request forgery	0	0
PCI DSS (3.2) - 6.5.10 - Broken authentication and session management	0	0

***** Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - FISMA 2014

Category	Description	Issues Found	Best Fix Locations
Access Control*	Organizations must limit information system access to authorized users, processes acting on behalf of authorized users, or devices (including other information systems) and to the types of transactions and functions that authorized users are permitted to exercise.	0	0
Audit And Accountability*	Organizations must: (i) create, protect, and retain information system audit records to the extent needed to enable the monitoring, analysis, investigation, and reporting of unlawful, unauthorized, or inappropriate information system activity; and (ii) ensure that the actions of individual information system users can be uniquely traced to those users so they can be held accountable for their actions.	0	0
Configuration Management*	Organizations must: (i) establish and maintain baseline configurations and inventories of organizational information systems (including hardware, software, firmware, and documentation) throughout the respective system development life cycles; and (ii) establish and enforce security configuration settings for information technology products employed in organizational information systems.	0	0
Identification And Authentication*	Organizations must identify information system users, processes acting on behalf of users, or devices and authenticate (or verify) the identities of those users, processes, or devices, as a prerequisite to allowing access to organizational information systems.	0	0
Media Protection*	Organizations must: (i) protect information system media, both paper and digital; (ii) limit access to information on information system media to authorized users; and (iii) sanitize or destroy information system media before disposal or release for reuse.	2	2
System And Communications Protection	Organizations must: (i) monitor, control, and protect organizational communications (i.e., information transmitted or received by organizational information systems) at the external boundaries and key internal boundaries of the information systems; and (ii) employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational information systems.	0	0
System And Information Integrity*	Organizations must: (i) identify, report, and correct information and information system flaws in a timely manner; (ii) provide protection from malicious code at appropriate locations within organizational information systems; and (iii) monitor information system security alerts and advisories and take appropriate actions in response.	0	0

***** Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - NIST SP 800-53

Category	Issues Found	Best Fix Locations
AC-12 Session Termination (P2)	0	0
AC-3 Access Enforcement (P1)*	0	0
AC-4 Information Flow Enforcement (P1)	0	0
AC-6 Least Privilege (P1)	0	0
AU-9 Protection of Audit Information (P1)*	0	0
CM-6 Configuration Settings (P2)	0	0
IA-5 Authenticator Management (P1)	0	0
IA-6 Authenticator Feedback (P2)	0	0
IA-8 Identification and Authentication (Non-Organizational Users) (P1)	0	0
SC-12 Cryptographic Key Establishment and Management (P1)	0	0
SC-13 Cryptographic Protection (P1)*	0	0
SC-17 Public Key Infrastructure Certificates (P1)	0	0
SC-18 Mobile Code (P2)	0	0
SC-23 Session Authenticity (P1)*	0	0
SC-28 Protection of Information at Rest (P1)*	0	0
SC-4 Information in Shared Resources (P1)*	2	2
SC-5 Denial of Service Protection (P1)*	55	55
SC-8 Transmission Confidentiality and Integrity (P1)*	0	0
SI-10 Information Input Validation (P1)*	0	0
SI-11 Error Handling (P2)*	0	0
SI-15 Information Output Filtering (P0)	0	0
SI-16 Memory Protection (P1)	0	0

***** Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - OWASP Mobile Top 10 2016

Category	Description	Issues Found	Best Fix Locations
M1-Improper Platform Usage	This category covers misuse of a platform feature or failure to use platform security controls. It might include Android intents, platform permissions, misuse of TouchID, the Keychain, or some other security control that is part of the mobile operating system. There are several ways that mobile apps can experience this risk.	0	0
M2-Insecure Data Storage	This category covers insecure data storage and unintended data leakage.	0	0
M3-Insecure Communication	This category covers poor handshaking, incorrect SSL versions, weak negotiation, cleartext communication of sensitive assets, etc.	0	0
M4-Insecure Authentication	This category captures notions of authenticating the end user or bad session management. This can include: -Failing to identify the user at all when that should be required -Failure to maintain the user's identity when it is required -Weaknesses in session management	0	0
M5-Insufficient Cryptography	The code applies cryptography to a sensitive information asset. However, the cryptography is insufficient in some way. Note that anything and everything related to TLS or SSL goes in M3. Also, if the app fails to use cryptography at all when it should, that probably belongs in M2. This category is for issues where cryptography was attempted, but it wasnt done correctly.	0	0
M6-Insecure Authorization	This is a category to capture any failures in authorization (e.g., authorization decisions in the client side, forced browsing, etc.). It is distinct from authentication issues (e.g., device enrolment, user identification, etc.). If the app does not authenticate users at all in a situation where it should (e.g., granting anonymous access to some resource or service when authenticated and authorized access is required), then that is an authentication failure not an authorization failure.	0	0
M7-Client Code Quality	This category is the catch-all for code-level implementation problems in the mobile client. That's distinct from server-side coding mistakes. This would capture things like buffer overflows, format string vulnerabilities, and various other code-level mistakes where the solution is to rewrite some code that's running on the mobile device.	0	0
M8-Code Tampering	This category covers binary patching, local resource modification, method hooking, method swizzling, and dynamic memory modification. Once the application is delivered to the mobile device, the code and data resources are resident there. An attacker can either directly modify the code, change the contents of memory dynamically, change or replace the system APIs that the application uses, or	0	0

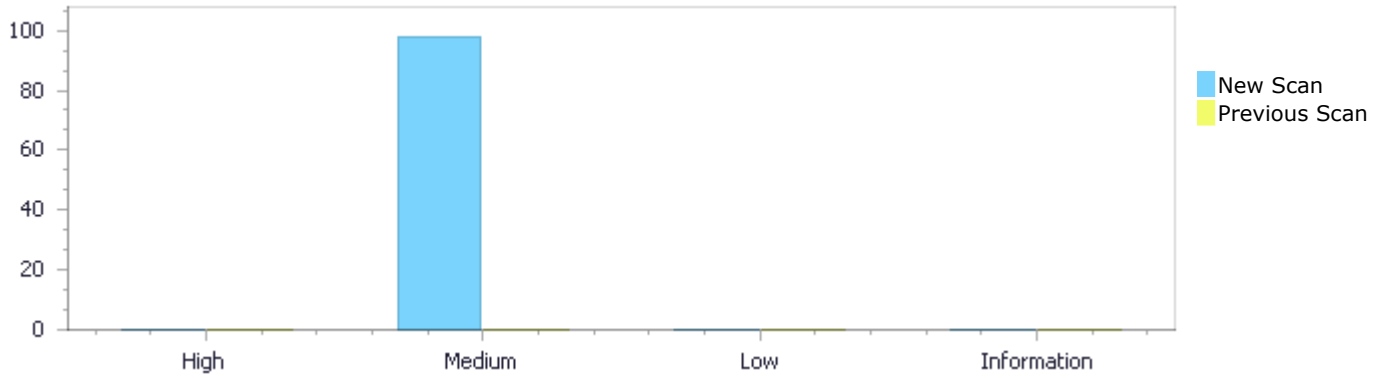
	modify the application's data and resources. This can provide the attacker a direct method of subverting the intended use of the software for personal or monetary gain.		
M9-Reverse Engineering	This category includes analysis of the final core binary to determine its source code, libraries, algorithms, and other assets. Software such as IDA Pro, Hopper, otool, and other binary inspection tools give the attacker insight into the inner workings of the application. This may be used to exploit other nascent vulnerabilities in the application, as well as revealing information about back end servers, cryptographic constants and ciphers, and intellectual property.	0	0
M10-Extraneous Functionality	Often, developers include hidden backdoor functionality or other internal development security controls that are not intended to be released into a production environment. For example, a developer may accidentally include a password as a comment in a hybrid app. Another example includes disabling of 2-factor authentication during testing.	0	0

Scan Summary - Custom

Category	Issues Found	Best Fix Locations
Must audit	0	0
Check	0	0
Optional	0	0

Results Distribution By Status First scan of the project

	High	Medium	Low	Information	Total
New Issues	0	98	0	0	98
Recurrent Issues	0	0	0	0	0
Total	0	98	0	0	98
Fixed Issues	0	0	0	0	0



Results Distribution By State

	High	Medium	Low	Information	Total
Confirmed	0	0	0	0	0
Not Exploitable	0	0	0	0	0
To Verify	0	98	0	0	98
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	0	98	0	0	98

Result Summary

Vulnerability Type	Occurrences	Severity
Memory Leak	55	Medium
Dangerous Functions	41	Medium
Heap Inspection	2	Medium

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
src/THiNXLib.cpp	96
src/THiNXLib.h	2

Scan Results Details

Memory Leak

Query Path:
CPP\Cx\CPP Medium Threat\Memory Leak Version:1

Categories

NIST SP 800-53: SC-5 Denial of Service Protection (P1)

Description

Memory Leak\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=44
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	74	74
Object	api_key_param	api_key_param

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
74.    api_key_param = new WiFiManagerParameter("apikey", "API Key",
thinx_api_key, 64);
```

Memory Leak\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=45
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	76	76
Object	owner_param	owner_param

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
76.    owner_param = new WiFiManagerParameter("owner", "Owner ID",
thinx_owner_key, 64);
```

Memory Leak\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=46
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1069	1069
Object	mqtt_client	mqtt_client

Code Snippet
File Name src/THiNXLib.cpp
Method bool THiNX::start_mqtt() {

```
....
1069.      mqtt_client = new PubSubClient(thx_wifi_client,
thinx_mqtt_url);
```

Memory Leak\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=47
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1074	1074
Object	mqtt_client	mqtt_client

Code Snippet
File Name src/THiNXLib.cpp
Method bool THiNX::start_mqtt() {

```
....
1074.        mqtt_client = new PubSubClient(https_client,
thinx_mqtt_url);
```

Memory Leak\Path 5:

Severity	Medium
Result State	To Verify

Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=48
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1147	1147
Object	mqtt_client	mqtt_client

Code Snippet
File Name src/THiNXLib.cpp
Method bool THiNX::start_mqtt() {

```
....
1147.            mqtt_client = new PubSubClient(thx_wifi_client,
thinx_mqtt_url);
```

Memory Leak\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=49
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	940	940
Object	format	format

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::thinx_time(const char* optional_format) {

```
....
940.    char *format = strdup(time_format);
```

Memory Leak\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=50
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	957	957
Object	format	format

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::thinx_date(const char* optional_format) {

```
....
957.    char *format = strdup(date_format);
```

Memory Leak\Path 8:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=51
Status New

	Source	Destination
File	src/THiNXLib.h	src/THiNXLib.h
Line	178	178
Object	thx_revision	thx_revision

Code Snippet
File Name src/THiNXLib.h
Method const char* thx_revision = strdup(String(THX_REVISION).c_str());

```
....
178.      const char* thx_revision =
strdup(String(THX_REVISION).c_str());
```

Memory Leak\Path 9:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=52
Status New

	Source	Destination
File	src/THiNXLib.h	src/THiNXLib.h
Line	184	184
Object	thx_commit_id	thx_commit_id

Code Snippet
File Name src/THiNXLib.h
Method const char* thx_commit_id = strdup(THX_COMMIT_ID);

```
....
184.      const char* thx_commit_id = strdup(THX_COMMIT_ID);
```

Memory Leak\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=53
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	89	89
Object	thinx_commit_id	thinx_commit_id

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
89.      thinx_commit_id = strdup(thx_commit_id); // -D build env var
```

Memory Leak\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=54
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	91	91
Object	thinx_commit_id	thinx_commit_id

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
91.      thinx_commit_id = strdup(THINX_COMMIT_ID); // value from
thinx.h
```

Memory Leak\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=55
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp

Line	113	113
Object	app_version	app_version

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
113.    app_version = strdup("");
```

Memory Leak\Path 13:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=56
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	114	114
Object	available_update_url	available_update_url

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
114.    available_update_url = strdup("");
```

Memory Leak\Path 14:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=57
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	115	115
Object	thinx_cloud_url	thinx_cloud_url

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
115.    thinx_cloud_url = strdup("thinx.cloud");
```

Memory Leak\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=58
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	117	117
Object	thinx_firmware_version_short	thinx_firmware_version_short

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
117.    thinx_firmware_version_short = strdup("");
```

Memory Leak\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=59
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	118	118
Object	thinx_firmware_version	thinx_firmware_version

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
118.    thinx_firmware_version = strdup("");
```

Memory Leak\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=60
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	119	119
Object	thinx_mqtt_url	thinx_mqtt_url

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
119.    thinx_mqtt_url = strdup("thinx.cloud");
```

Memory Leak\Path 18:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=61
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	120	120
Object	thinx_version_id	thinx_version_id

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
120.    thinx_version_id = strdup("");
```

Memory Leak\Path 19:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=62
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	121	121
Object	thinx_api_key	thinx_api_key

Code Snippet
File Name src/THiNXLib.cpp

Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
121.    thinx_api_key = strdup("");
```

Memory Leak\Path 20:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=63
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	130	130
Object	thinx_owner	thinx_owner

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
130.      thinx_owner = strdup("");
```

Memory Leak\Path 21:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=64
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	152	152
Object	thinx_api_key	thinx_api_key

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
152.      thinx_api_key = strdup(__apikey);
```

Memory Leak\Path 22:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=65

Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	164	164
Object	thinx_owner	thinx_owner

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
164.      thinx_owner = strdup(__owner_id);
```

Memory Leak\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=66
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	195	195
Object	thinx_api_key	thinx_api_key

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::init_with_api_key(const char * __apikey) {

```
....
195.      thinx_api_key = strdup(__apikey);
```

Memory Leak\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=67
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	626	626
Object	thinx_udid	thinx_udid

Code Snippet

File Name	src/THiNXLib.cpp
Method	void THiNX::parse(String payload) {

```
....
626.          thinx_udid = strdup(udid.c_str());
```

Memory Leak\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=68
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	640	640
Object	available_update_url	available_update_url

Code Snippet

File Name	src/THiNXLib.cpp
Method	void THiNX::parse(String payload) {

```
....
640.          available_update_url = strdup("");
```

Memory Leak\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=69
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	686	686
Object	expected_hash	expected_hash

Code Snippet

File Name	src/THiNXLib.cpp
Method	void THiNX::parse(String payload) {

```
....
686.            expected_hash = strdup(hash.c_str());
```

Memory Leak\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=

	15&pathid=70
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	692	692
Object	expected_md5	expected_md5

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
692.            expected_md5 = strdup(md5.c_str());
```

Memory Leak\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=71
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	769	769
Object	thinx_alias	thinx_alias

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
769.            thinx_alias = strdup(alias.c_str());
```

Memory Leak\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=72
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	774	774
Object	thinx_owner	thinx_owner

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
774.            thinx_owner = strdup(owner.c_str());
```

Memory Leak\Path 30:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=73
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	779	779
Object	thinx_udid	thinx_udid

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
779.            thinx_udid = strdup(udid.c_str());
```

Memory Leak\Path 31:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=74
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	807	807
Object	thinx_udid	thinx_udid

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
807.            thinx_udid = strdup(udid.c_str());
```

Memory Leak\Path 32:

Severity Medium
Result State To Verify

Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=75
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	849	849
Object	expected_hash	expected_hash

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
849.            expected_hash = strdup(hash.c_str());
```

Memory Leak\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=76
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	855	855
Object	expected_md5	expected_md5

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
855.            expected_md5 = strdup(md5.c_str());
```

Memory Leak\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=77
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	942	942
Object	format	format

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::thinx_time(const char* optional_format) {

```
....
942.        format = strdup(optional_format);
```

Memory Leak\Path 35:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=78
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	959	959
Object	format	format

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::thinx_date(const char* optional_format) {

```
....
959.        format = strdup(optional_format);
```

Memory Leak\Path 36:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=79
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1239	1239
Object	thinx_alias	thinx_alias

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::restore_device_info() {

```
....
1239.         thinx_alias = strdup(config["alias"]);
```

Memory Leak\Path 37:

Severity	Medium

Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=80
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1245	1245
Object	thinx_udid	thinx_udid

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::restore_device_info() {

```
....
1245.           thinx_udid = strdup(udid);
```

Memory Leak\Path 38:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=81
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1247	1247
Object	thinx_udid	thinx_udid

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::restore_device_info() {

```
....
1247.           thinx_udid = strdup(THINX_UDID);
```

Memory Leak\Path 39:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=82
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1250	1250

Object	thinx_udid	thinx_udid

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::restore_device_info() {

```
....
1250.        thinx_udid = strdup(THINX_UDID);
```

Memory Leak\Path 40:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=83
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1254	1254
Object	thinx_api_key	thinx_api_key

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::restore_device_info() {

```
....
1254.        thinx_api_key = strdup(config["apikey"]);
```

Memory Leak\Path 41:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=84
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1258	1258
Object	thinx_owner	thinx_owner

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::restore_device_info() {

```
....
1258.        thinx_owner = strdup(config["owner"]);
```

Memory Leak\Path 42:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=85
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1262	1262
Object	available_update_url	available_update_url

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::restore_device_info() {

```
....
1262.         available_update_url = strdup(config["ott"]);
```

Memory Leak\Path 43:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=86
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1402	1402
Object	thinx_api_key	thinx_api_key

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1402.       thinx_api_key = strdup(THINX_API_KEY);
```

Memory Leak\Path 44:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=87
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp

Line	1406	1406
Object	thinx_udid	thinx_udid

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1406.       thinx_udid = strdup(THINX_UDID);
```

Memory Leak\Path 45:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=88
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1408	1408
Object	thinx_udid	thinx_udid

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1408.       thinx_udid = strdup("");
```

Memory Leak\Path 46:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=89
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1413	1413
Object	thinx_commit_id	thinx_commit_id

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1413.    thinx_commit_id = strdup(thx_commit_id);
```

Memory Leak\Path 47:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=90
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1418	1418
Object	thinx_mqtt_url	thinx_mqtt_url

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1418.    thinx_mqtt_url = strdup(THINX_MQTT_URL);
```

Memory Leak\Path 48:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=91
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1419	1419
Object	thinx_cloud_url	thinx_cloud_url

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1419.    thinx_cloud_url = strdup(THINX_CLOUD_URL);
```

Memory Leak\Path 49:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=92
Status	New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1420	1420
Object	thinx_alias	thinx_alias

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1420.    thinx_alias = strdup(THINX_ALIAS);
```

Memory Leak\Path 50:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=93
Status New

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1421	1421
Object	thinx_owner	thinx_owner

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1421.    thinx_owner = strdup(THINX_OWNER);
```

Dangerous Functions

Query Path:
CPP\Cx\CPP Medium Threat\Dangerous Functions Version:1

Categories

OWASP Top 10 2013: A9-Using Components with Known Vulnerabilities
OWASP Top 10 2017: A9-Using Components with Known Vulnerabilities

Description

Dangerous Functions\Path 1:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=1
Status New

The dangerous function, sprintf, was found in use at line 1722 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1754	1754
Object	sprintf	sprintf

Code Snippet
File Name src/THiNXLib.cpp
Method bool THiNX::check_hash(char * filename, char * expected) {

```
....
1754.        sprintf(aes_text + 2 * i, "%02X", obuf[i]);
```

Dangerous Functions\Path 2:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=2
Status New

The dangerous function, sprintf, was found in use at line 918 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	919	919
Object	sprintf	sprintf

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::thinx_mqtt_channel() {

```
....
919.    sprintf(mqtt_device_channel, "/%s/%s", thinx_owner, thinx_udid);
```

Dangerous Functions\Path 3:

Severity Medium
Result State To Verify
Online Results http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=3
Status New

The dangerous function, sprintf, was found in use at line 923 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination

File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	924	924
Object	sprintf	sprintf

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::thinx_mqtt_channels() {

```
....
924.    sprintf(mqtt_device_channels, "/%s/%s/#", thinx_owner,
thinx_udid);
```

Dangerous Functions\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=4
Status	New

The dangerous function, sprintf, was found in use at line 928 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	929	929
Object	sprintf	sprintf

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::thinx_mqtt_status_channel() {

```
....
929.    sprintf(mqtt_device_status_channel, "/%s/%s/status",
thinx_owner, thinx_udid);
```

Dangerous Functions\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=5
Status	New

The dangerous function, sprintf, was found in use at line 976 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	977	977

Object	sprintf	sprintf

Code Snippet
File Name src/THiNXLib.cpp
Method const char * THiNX::thinx_mac() {

```
....
977.    sprintf(mac_string, "5CCF7F%6X", ESP.getChipId()); // ESP8266
only!
```

Dangerous Functions\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=6
Status	New

The dangerous function, strcpy, was found in use at line 40 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	43	43
Object	strcpy	strcpy

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::saveConfigCallback() {

```
....
43.    strcpy(thx_api_key, api_key_param->getValue());
```

Dangerous Functions\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=7
Status	New

The dangerous function, strcpy, was found in use at line 40 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	44	44
Object	strcpy	strcpy

Code Snippet

File Name	src/THiNXLib.cpp
Method	void THiNX::saveConfigCallback() {

```
....
44.    strcpy(thx_owner_key, owner_param->getValue());
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=8
Status	New

The dangerous function, strlen, was found in use at line 67 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	88	88
Object	strlen	strlen

Code Snippet

File Name	src/THiNXLib.cpp
Method	THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
88.    if (strlen(thx_commit_id) > 8) {
```

Dangerous Functions\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=9
Status	New

The dangerous function, strlen, was found in use at line 67 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	129	129
Object	strlen	strlen

Code Snippet

File Name	src/THiNXLib.cpp
Method	THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
129.    if (strlen(__owner_id) == 0) {
```

Dangerous Functions\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=10
Status	New

The dangerous function, strlen, was found in use at line 67 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	150	150
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
150.    if (strlen(__apikey) > 4) {
```

Dangerous Functions\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=11
Status	New

The dangerous function, strlen, was found in use at line 67 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	154	154
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
154.        if (strlen(thinx_api_key) > 4) {
```

Dangerous Functions\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=12
Status	New

The dangerous function, strlen, was found in use at line 67 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	162	162
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
162.    if (strlen(__owner_id) > 4) {
```

Dangerous Functions\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=13
Status	New

The dangerous function, strlen, was found in use at line 67 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	166	166
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method THiNX::THiNX(const char * __apikey, const char * __owner_id) {

```
....
166.        if (strlen(thinx_owner) > 4) {
```

Dangerous Functions\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=14

Status	New

The dangerous function, strlen, was found in use at line 179 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	194	194
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::init_with_api_key(const char * __apikey) {

```
....
194.    if (strlen(__apikey) > 4) {
```

Dangerous Functions\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=15
Status	New

The dangerous function, strlen, was found in use at line 179 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	197	197
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::init_with_api_key(const char * __apikey) {

```
....
197.      if (strlen(thinx_api_key) < 4) {
```

Dangerous Functions\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=16
Status	New

The dangerous function, strlen, was found in use at line 215 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	233	233
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::connect() {

```
....
233.          if (strlen(THINX_ENV_SSID) > 2) {
```

Dangerous Functions\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=17
Status	New

The dangerous function, strlen, was found in use at line 350 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	357	357
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::checkin_body() {

```
....
357.    if (strlen(thinx_firmware_version) > 1) {
```

Dangerous Functions\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=18
Status	New

The dangerous function, strlen, was found in use at line 350 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	363	363

Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::checkin_body() {

```
....
363.    if (strlen(thinx_firmware_version_short) > 1) {
```

Dangerous Functions\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=19
Status	New

The dangerous function, strlen, was found in use at line 350 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	367	367
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::checkin_body() {

```
....
367.    if (strlen(thx_commit_id) > 1) {
```

Dangerous Functions\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=20
Status	New

The dangerous function, strlen, was found in use at line 350 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	371	371
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp

Method String THiNX::checkin_body() {

```
....
371.    if (strlen(thinx_owner) > 1) {
```

Dangerous Functions\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=21
Status	New

The dangerous function, strlen, was found in use at line 350 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	375	375
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::checkin_body() {

```
....
375.    if (strlen(thinx_alias) > 1) {
```

Dangerous Functions\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=22
Status	New

The dangerous function, strlen, was found in use at line 350 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	379	379
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method String THiNX::checkin_body() {

```
....
379.    if (strlen(thinx_udid) > 4) {
```

Dangerous Functions\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=23
Status	New

The dangerous function, strlen, was found in use at line 547 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	638	638
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
638.        if (strlen(available_update_url) > 5) {
```

Dangerous Functions\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=24
Status	New

The dangerous function, strlen, was found in use at line 547 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	731	731
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
731.            if (strlen(available_update_url) > 4) {
```

Dangerous Functions\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=

	15&pathid=25
Status	New

The dangerous function, strlen, was found in use at line 547 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	743	743
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
743.            if (strlen(available_update_url) > 4) {
```

Dangerous Functions\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=26
Status	New

The dangerous function, strlen, was found in use at line 547 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	885	885
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
885.        if ((strlen(ssid) > 2) && (strlen(pass) > 0)) {
```

Dangerous Functions\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=27
Status	New

The dangerous function, strlen, was found in use at line 547 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	885	885
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
885.        if ((strlen(ssid) > 2) && (strlen(pass) > 0)) {
```

Dangerous Functions\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=28
Status	New

The dangerous function, strlen, was found in use at line 1051 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1062	1062
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method bool THiNX::start_mqtt() {

```
....
1062.    if (strlen(thinx_udid) < 4) {
```

Dangerous Functions\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=29
Status	New

The dangerous function, strlen, was found in use at line 1051 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1083	1083

Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method bool THiNX::start_mqtt() {

```
....
1083.    if (strlen(thinx_api_key) < 5) {
```

Dangerous Functions\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=30
Status	New

The dangerous function, strlen, was found in use at line 1162 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1244	1244
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::restore_device_info() {

```
....
1244.        if ( strlen(udid) > 2 ) {
```

Dangerous Functions\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=31
Status	New

The dangerous function, strlen, was found in use at line 1306 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1313	1313
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp

Method void THiNX::deviceInfo() {

```
....
1313.    if (strlen(thinx_owner) > 1) {
```

Dangerous Functions\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=32
Status	New

The dangerous function, strlen, was found in use at line 1306 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1317	1317
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::deviceInfo() {

```
....
1317.    if (strlen(thinx_api_key) > 1) {
```

Dangerous Functions\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=33
Status	New

The dangerous function, strlen, was found in use at line 1306 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1321	1321
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::deviceInfo() {

```
....
1321.    if (strlen(thinx_udid) > 1) {
```

Dangerous Functions\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=34
Status	New

The dangerous function, strlen, was found in use at line 1306 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1326	1326
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::deviceInfo() {

```
....
1326.    if (strlen(available_update_url) > 1) {
```

Dangerous Functions\Path 35:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=35
Status	New

The dangerous function, strlen, was found in use at line 1398 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1401	1401
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1401.    if (strlen(thinx_api_key) < 4) {
```

Dangerous Functions\Path 36:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=

	15&pathid=36
Status	New

The dangerous function, strlen, was found in use at line 1398 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1405	1405
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::import_build_time_constants() {

```
....
1405.    if (strlen(THINX_UDID) > 2) {
```

Dangerous Functions\Path 37:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=37
Status	New

The dangerous function, strlen, was found in use at line 1473 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1475	1475
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::evt_save_api_key() {

```
....
1475.      if (strlen(thx_api_key) > 4) {
```

Dangerous Functions\Path 38:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=38
Status	New

The dangerous function, strlen, was found in use at line 1473 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1479	1479
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::evt_save_api_key() {

```
....
1479.       if (strlen(thx_owner_key) > 4) {
```

Dangerous Functions\Path 39:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=39
Status	New

The dangerous function, strlen, was found in use at line 1537 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1586	1586
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::loop() {

```
....
1586.       if (strlen(mqtt_device_channel) > 5) {
```

Dangerous Functions\Path 40:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=40
Status	New

The dangerous function, strlen, was found in use at line 1537 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1606	1606

Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::loop() {

```
....
1606.       if (strlen(thinx_udid) > 4) {
```

Dangerous Functions\Path 41:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=41
Status	New

The dangerous function, strlen, was found in use at line 1537 in src/THiNXLib.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1644	1644
Object	strlen	strlen

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::loop() {

```
....
1644.       if (strlen(thinx_api_key) > 4) {
```

Heap Inspection

Query Path:
CPP\Cx\CPP Medium Threat\Heap Inspection Version:1

Categories

OWASP Top 10 2013: A6-Sensitive Data Exposure
FISMA 2014: Media Protection
NIST SP 800-53: SC-4 Information in Shared Resources (P1)
OWASP Top 10 2017: A3-Sensitive Data Exposure

Description

Heap Inspection\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=42
Status	New

Method THiNX::parse at line 547 of src/THiNXLib.cpp defines pass, which is designated to contain user passwords. However, while plaintext passwords are later assigned to pass, this variable is never cleared from memory.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	882	882
Object	pass	pass

Code Snippet
File Name src/THiNXLib.cpp
Method void THiNX::parse(String payload) {

```
....
882.        const char *pass = configuration["THINX_ENV_PASS"];
```

Heap Inspection\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://SRVRV128/CxWebClient/ViewerMain.aspx?scanid=1000014&projectid=15&pathid=43
Status	New

Method THiNX::start_mqtt at line 1051 of src/THiNXLib.cpp defines pass, which is designated to contain user passwords. However, while plaintext passwords are later assigned to pass, this variable is never cleared from memory.

	Source	Destination
File	src/THiNXLib.cpp	src/THiNXLib.cpp
Line	1090	1090
Object	pass	pass

Code Snippet
File Name src/THiNXLib.cpp
Method bool THiNX::start_mqtt() {

```
....
1090.    const char* pass = thinx_api_key;
```

Dangerous Functions

Risk

What might happen

Use of dangerous functions may expose varying risks associated with each particular function, with potential impact of improper usage of these functions varying significantly. The presence of such functions indicates a flaw in code maintenance policies and adherence to secure coding practices, in a way that has allowed introducing known dangerous code into the application.

Cause

How does it happen

A dangerous function has been identified within the code. Functions are often deemed dangerous to use for numerous reasons, as there are different sets of vulnerabilities associated with usage of such functions. For example, some string copy and concatenation functions are vulnerable to Buffer Overflow, Memory Disclosure, Denial of Service and more. Use of these functions is not recommended.

General Recommendations

How to avoid it

- Deploy a secure and recommended alternative to any functions that were identified as dangerous.
    - If no secure alternative is found, conduct further researching and testing to identify whether current usage successfully sanitizes and verifies values, and thus successfully avoids the use-cases for whom the function is indeed dangerous
- Conduct a periodical review of methods that are in use, to ensure that all external libraries and built-in functions are up-to-date and whose use has not been excluded from best secure coding practices.

Source Code Examples

CPP

Buffer Overflow in gets()

```
int main()
{
	char buf[10];

	printf("Please enter your name: ");
	gets(buf); // veryveryverylongname
	if (buf == ACCEPTED_NAME)
	{
		// Do something
	}
	return 0;
}
```

Safe reading from user

```
int main()
{
	char buf[10];

	printf("Please enter your name: ");
	fgets(buf, sizeof(buf), stdin); //setting the amount of bytes to read
	if (buf == ACCEPTED_NAME)
	{
		//Do something
```

```
	}
	return 0;
}
```

Unsafe function for string copy

```
int main(int argc, char* argv[])
{
	char buf[10];
	strcpy(buf, argv[1]); // overflow occurs when len(argv[1]) > 10 bytes

	return 0;
}
```

Safe string copy

```
int main(int argc, char* argv[])
{
	char buf[10];
	strncpy(buf, argv[1], sizeof(buf));
	buf[9]= '\0'; //strncpy doesn't NULL terminates

	return 0;
}
```

Unsafe format string

```
int main(int argc, char* argv[])
{
	printf(argv[1]); // If argv[1] contains a format token, such as %s,%x or %d, will cause
an access violation
	return 0;
}
```

Safe format string

```
int main(int argc, char* argv[])
{
	printf("%s", argv[1]); // Second parameter is not a formattable string

	return 0;
}
```

Heap Inspection

Risk

What might happen

All variables stored by the application in unencrypted memory can potentially be retrieved by an unauthorized user, with privlieged access to the machine. For example, a privileged attacker could attach a debugger to the running process, or retrieve the process's memory from the swapfile or crash dump file.
Once the attacker finds the user passwords in memory, these can be reused to easily impersonate the user to the system.

Cause

How does it happen

String variables are immutable - in other words, once a string variable is assigned, its value cannot be changed or removed. Thus, these strings may remain around in memory, possibly in multiple locations, for an indefinite period of time until the garbage collector happens to remove it. Sensitive data, such as passwords, will remain exposed in memory as plaintext with no control over their lifetime.

General Recommendations

How to avoid it

Generic Guidance:

- Do not store senstiive data, such as passwords or encryption keys, in memory in plaintext, even for a short period of time.
- Prefer to use specialized classes that store encrypted memory.
- Alternatively, store secrets temporarily in mutable data types, such as byte arrays, and then promptly zeroize the memory locations.

Specific Recommendations - Java:

- Instead of storing passwords in immutable strings, prefer to use an encrypted memory object, such as SealedObject.

Specific Recommendations - .NET:

- Instead of storing passwords in immutable strings, prefer to use an encrypted memory object, such as SecureString or ProtectedData.

Source Code Examples

Java

Plaintext Password in Immutable String

```
class Heap_Inspection
{
  private string password;
```

```
  void setPassword()
 {
      password = System.console().readLine("Enter your password: ");
  }
}
```

Password Protected in Memory

```
class Heap_Inspection_Fixed
{
  private SealedObject password;

  void setPassword()
 {
      byte[] sKey = getKeyFromConfig();
      Cipher c = Cipher.getInstance("AES");
      c.init(Cipher.ENCRYPT_MODE, sKey);

      char[] input = System.console().readPassword("Enter your password: ");
      password = new SealedObject(Arrays.asList(input), c);

    //Zero out the possible password, for security.
    Arrays.fill(password, '0');
  }
}
```

CPP

Vulnerable C code

```
/* Vulnerable to heap inspection */

#include <stdio.h>


void somefunc(){
      printf("Yea, I'm just being called for the heap of it..\n");
}

void authfunc(){
        char* password = (char *) malloc(256);
        char ch;
        ssize_t k;
            int i=0;
        while(k = read(0, &ch, 1) > 0)
        {
                if (ch == '\n'){
                        password[i]='\0';
                        break;
                } else{
                        password[i++]=ch;
                        fflush(0);
                }
        }
        printf("Password: %s\n",&password[0]);
}
```

```
int main()
{
    printf("Please enter a password:\n");

      authfunc();
      printf("You can now dump memory to find this password!");
      somefunc();
      gets();

}
```

Safe C code

```
/* Pesumably safe heap */

#include <stdio.h>
#include <string.h>

#define STDIN_FILENO 0

void somefunc(){
        printf("Yea, I'm just being called for the heap of it..\n");
}

void authfunc(){
      char* password = (char*) malloc(256);
      int i=0;
      char ch;
      ssize_t k;
      while(k = read(STDIN_FILENO, &ch, 1) > 0)
      {
            if (ch == '\n'){
                  password[i]='\0';
                  break;
            } else{
                  password[i++]=ch;
                  fflush(0);
            }
      }
      i=0;
      memset(password,'\0',256);
}

int main()
{
      printf("Please enter a password:\n");
      authfunc();
      somefunc();
      char ch;
      while(read(STDIN_FILENO, &ch, 1) > 0)
      {
            if (ch == '\n')
                  break;
      }
}
```

Failure to Release Memory Before Removing Last Reference ('Memory Leak')

Weakness ID: 401 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The software does not sufficiently track and release allocated memory after it has been used, which slowly consumes remaining memory.

Extended Description

This is often triggered by improper handling of malformed data or unexpectedly interrupted sessions.

Terminology Notes

"memory leak" has sometimes been used to describe other kinds of issues, e.g. for information leaks in which the contents of memory are inadvertently leaked (CVE-2003-0400 is one such example of this terminology conflict).

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C

C++

Modes of Introduction

Memory leaks have two common and sometimes overlapping causes:

- Error conditions and other exceptional circumstances
- Confusion over which part of the program is responsible for freeing the memory

Common Consequences

Scope	Effect
Availability	Most memory leaks result in general software reliability problems, but if an attacker can intentionally trigger a memory leak, the attacker might be able to launch a denial of service attack (by crashing or hanging the program) or take advantage of other unexpected program behavior resulting from a low memory condition.

Likelihood of Exploit

Medium

Demonstrative Examples

Example 1

The following C function leaks a block of allocated memory if the call to read() fails to return the expected number of bytes:

(Bad Code)

Example Language: **C**

```
char* getBlock(int fd) {
char* buf = (char*) malloc(BLOCK_SIZE);
if (!buf) {
return NULL;
}
if (read(fd, buf, BLOCK_SIZE) != BLOCK_SIZE) {

return NULL;
}
```

```
return buf;
}
```

Example 2

Here the problem is that every time a connection is made, more memory is allocated. So if one just opened up more and more connections, eventually the machine would run out of memory.

(Bad Code)

Example Language: **C**

```
bar connection(){
foo = malloc(1024);
return foo;
}
endConnection(bar foo) {

free(foo);
}
int main() {

while(1) //thread 1
//On a connection
foo=connection(); //thread 2
//When the connection ends
endConnection(foo)
}
```

Observed Examples

Reference	Description
CVE-2005-3119	Memory leak because function does not free() an element of a data structure.
CVE-2004-0427	Memory leak when counter variable is not decremented.
CVE-2002-0574	Memory leak when counter variable is not decremented.
CVE-2005-3181	Kernel uses wrong function to release a data structure, preventing data from being properly tracked by other code.
CVE-2004-0222	Memory leak via unknown manipulations as part of protocol test suite.
CVE-2001-0136	Memory leak via a series of the same command.

Potential Mitigations

Pre-design: Use a language or compiler that performs automatic bounds checking.

Phase: Architecture and Design

Use an abstraction library to abstract away risky APIs. Not a complete solution.

Pre-design through Build: The Boehm-Demers-Weiser Garbage Collector or valgrind can be used to detect leaks in code. This is not a complete solution as it is not 100% effective.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	398	Indicator of Poor Code Quality	**Seven Pernicious Kingdoms (primary)700**
ChildOf	Category	399	Resource Management Errors	**Development Concepts (primary)699**
ChildOf	Category	633	Weaknesses that Affect Memory	**Resource-specific Weaknesses (primary)631**
ChildOf	Category	730	OWASP Top Ten 2004 Category A9 - Denial of Service	**Weaknesses in OWASP Top Ten (2004) (primary)711**
ChildOf	Weakness Base	772	Missing Release of Resource after Effective	**Research Concepts (primary)1000**

			Lifetime	
MemberOf	View	630	Weaknesses Examined by SAMATE	**Weaknesses Examined by SAMATE (primary)630**
CanFollow	Weakness Class	390	Detection of Error Condition Without Action	Research Concepts1000

Relationship Notes

This is often a resultant weakness due to improper handling of malformed data or early termination of sessions.

Affected Resources

- Memory

Functional Areas

- Memory management

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
PLOVER			Memory leak
7 Pernicious Kingdoms			Memory Leak
CLASP			Failure to deallocate data
OWASP Top Ten 2004	A9	CWE More Specific	Denial of Service

White Box Definitions

A weakness where the code path has:

1. start statement that allocates dynamically allocated memory resource

2. end statement that loses identity of the dynamically allocated memory resource creating situation where dynamically allocated memory resource is never relinquished

Where "loses" is defined through the following scenarios:

1. identity of the dynamic allocated memory resource never obtained

2. the statement assigns another value to the data element that stored the identity of the dynamically allocated memory resource and there are no aliases of that data element

3. identity of the dynamic allocated memory resource obtained but never passed on to function for memory resource release

4. the data element that stored the identity of the dynamically allocated resource has reached the end of its scope at the statement and there are no aliases of that data element

References

J. Whittaker and H. Thompson. "How to Break Software Security". Addison Wesley. 2003.

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	PLOVER		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-01		KDM Analytics	External
	added/updated white box definitions		
2008-08-15		Veracode	External
	Suggested OWASP Top Ten 2004 mapping		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, References, Relationship Notes, Taxonomy Mappings, Terminology Notes		
2008-10-14	CWE Content Team	MITRE	Internal
	updated Description		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Other Notes		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Name		
2009-07-17	KDM Analytics		External
	Improved the White Box Definition		

2009-07-27	CWE Content Team	MITRE	Internal
	updated White Box Definitions		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Modes of Introduction, Other Notes		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

Previous Entry Names

Change Date	Previous Entry Name
2008-04-11	Memory Leak
2009-05-27	Failure to Release Memory Before Removing Last Reference (aka 'Memory Leak')

BACK TO TOP

Scanned Languages

Language	Hash Number	Change Date
CPP	4541647240435660	6/11/2018
Common	0105849645654507	6/11/2018