Arduino_STM32 Scan Report

Project Name	Arduino_STM32
Scan Start	Friday, June 21, 2024 11:43:11 PM
Preset	Checkmarx Default
Scan Time	00h:14m:03s
Lines Of Code Scanned	3850
Files Scanned	9
Report Creation Time	Saturday, June 22, 2024 12:04:42 AM
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072
Team	CxServer
Checkmarx Version	8.7.0
Scan Type	Full
Source Origin	LocalPath
Density	2/100 (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

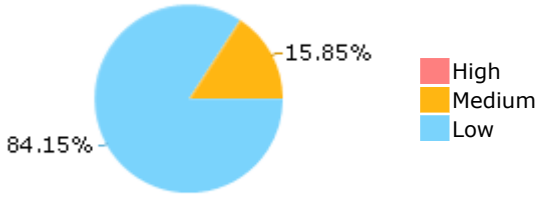
Results Limit

Results limit per query was set to 50

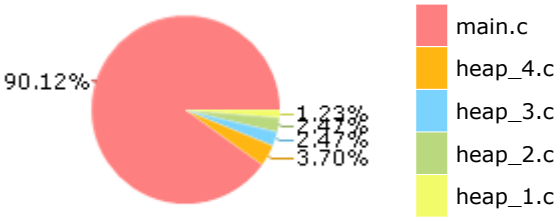
Selected Queries

Selected queries are listed in Result Summary

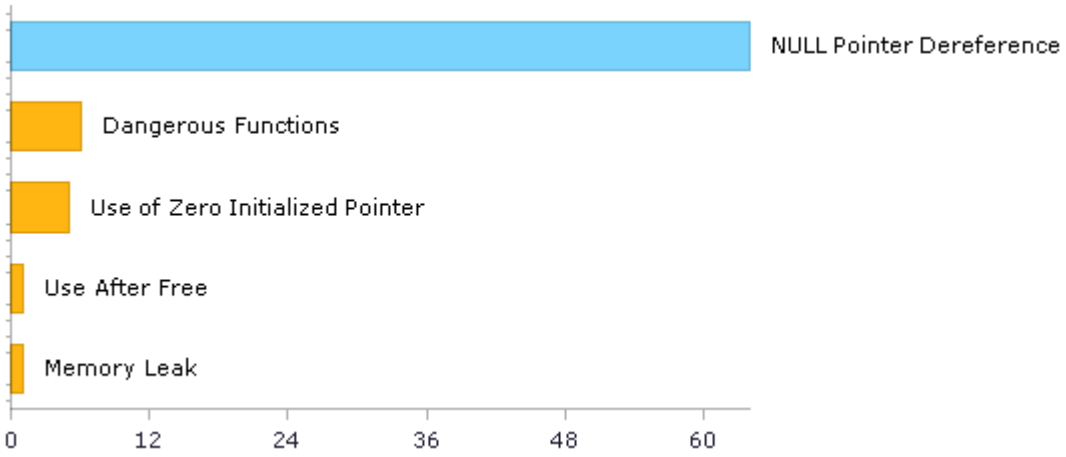
Result Summary



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	65	4
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	4	4
A3-Sensitive Data Exposure	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	0	0
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	6	6
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	0	0
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	6	6
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.	4	4
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.	0	0
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)	4	4
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)	0	0
SC-5 Denial of Service Protection (P1)*	71	10
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

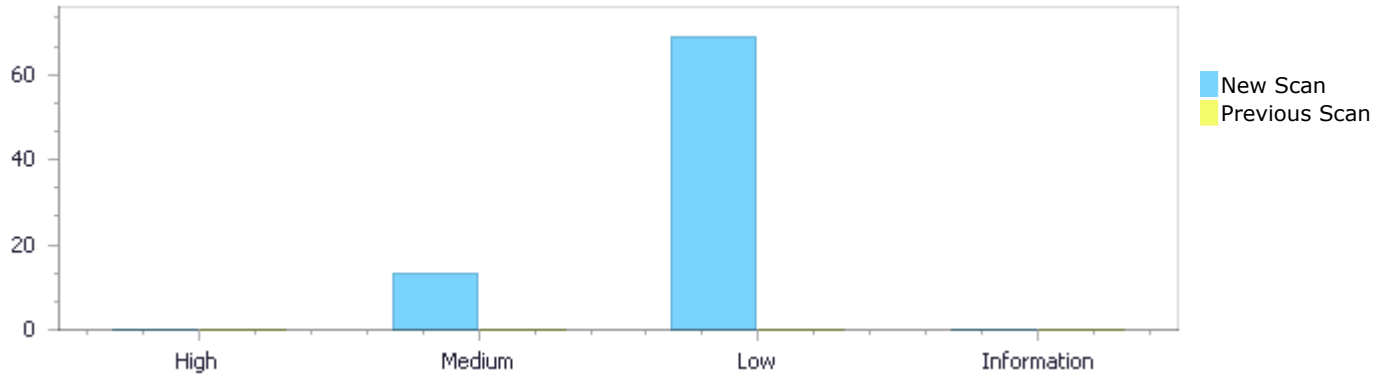
Scan Summary - Custom

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	13	69	0	82
Recurrent Issues	0	0	0	0	0
Total	0	13	69	0	82
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	13	69	0	82
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	0	13	69	0	82

Result Summary

Vulnerability Type	Occurrences	Severity
Dangerous Functions	6	Medium
Use of Zero Initialized Pointer	5	Medium
Memory Leak	1	Medium
Use After Free	1	Medium
NULL Pointer Dereference	64	Low

Improper Resource Access Authorization	4	Low
Inconsistent Implementations	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
Arduino_STM32/main.c	6
Arduino_STM32/heap_3.c	2
Arduino_STM32/heap_4.c	2
Arduino_STM32/heap_1.c	1
Arduino_STM32/heap_2.c	1
Arduino_STM32/heap_5.c	1

Scan Results Details

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://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=66
Status	New

The dangerous function, atoi, was found in use at line 237 in Arduino_STM32/main.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	286	286
Object	atoi	atoi

Code Snippet
File Name Arduino_STM32/main.c
Method int main(int argc, char **argv)

```
....
286.                    detach_delay = atoi(optarg);
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=67
Status	New

The dangerous function, atoi, was found in use at line 237 in Arduino_STM32/main.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	301	301

Object	atoi	atoi

Code Snippet
File Name Arduino_STM32/main.c
Method int main(int argc, char **argv)

```
....
301.                    match_config_index = atoi(optarg);
```

Dangerous Functions\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=68
Status	New

The dangerous function, atoi, was found in use at line 237 in Arduino_STM32/main.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	305	305
Object	atoi	atoi

Code Snippet
File Name Arduino_STM32/main.c
Method int main(int argc, char **argv)

```
....
305.                    match_iface_index = atoi(optarg);
```

Dangerous Functions\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=69
Status	New

The dangerous function, atoi, was found in use at line 237 in Arduino_STM32/main.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	319	319
Object	atoi	atoi

Code Snippet	
File Name	Arduino_STM32/main.c
Method	int main(int argc, char **argv)

```
....
319.                    transfer_size = atoi(optarg);
```

Dangerous Functions\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=70
Status	New

The dangerous function, atoi, was found in use at line 237 in Arduino_STM32/main.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	326	326
Object	atoi	atoi

Code Snippet	
File Name	Arduino_STM32/main.c
Method	int main(int argc, char **argv)

```
....
326.                    expected_size = atoi(optarg);
```

Dangerous Functions\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=71
Status	New

The dangerous function, atoi, was found in use at line 147 in Arduino_STM32/main.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	157	157
Object	atoi	atoi

Code Snippet	
File Name	Arduino_STM32/main.c
Method	static int resolve_device_path(char *path)

```
....
157.        match_bus = atoi(path);
```

Use of Zero Initialized Pointer

Query Path:
CPP\Cx\CPP Medium Threat\Use of Zero Initialized Pointer Version:1

Categories

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

Description

Use of Zero Initialized Pointer\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=74
Status	New

The variable declared in pucAlignedHeap at Arduino_STM32/heap_1.c in line 99 is not initialized when it is used by pucAlignedHeap at Arduino_STM32/heap_1.c in line 99.

	Source	Destination
File	Arduino_STM32/heap_1.c	Arduino_STM32/heap_1.c
Line	102	127
Object	pucAlignedHeap	pucAlignedHeap

Code Snippet
File Name Arduino_STM32/heap_1.c
Method void *pvPortMalloc(size_t xWantedSize)

```
....
102.  static uint8_t *pucAlignedHeap = NULL;
....
127.                    pvReturn = pucAlignedHeap + xNextFreeByte;
```

Use of Zero Initialized Pointer\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=75
Status	New

The variable declared in pvReturn at Arduino_STM32/heap_4.c in line 151 is not initialized when it is used by pvReturn at Arduino_STM32/heap_4.c in line 151.

	Source	Destination
File	Arduino_STM32/heap_4.c	Arduino_STM32/heap_4.c

Line	154	296
Object	pvReturn	pvReturn

Code Snippet
File Name Arduino_STM32/heap_4.c
Method void *pvPortMalloc(size_t xWantedSize)

```
....
154.  void *pvReturn = NULL;
....
296.        configASSERT( ( ( ( uint32_t ) pvReturn ) &
portBYTE_ALIGNMENT_MASK ) == 0 );
```

Use of Zero Initialized Pointer\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=76
Status	New

The variable declared in pxNextFreeBlock at Arduino_STM32/heap_2.c in line 154 is not initialized when it is used by pxNextFreeBlock at Arduino_STM32/heap_2.c in line 154.

	Source	Destination
File	Arduino_STM32/heap_2.c	Arduino_STM32/heap_2.c
Line	166	166
Object	pxNextFreeBlock	pxNextFreeBlock

Code Snippet
File Name Arduino_STM32/heap_2.c
Method void *pvPortMalloc(size_t xWantedSize)

```
....
166.                    prvHeapInit();
```

Use of Zero Initialized Pointer\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=77
Status	New

The variable declared in pxNextFreeBlock at Arduino_STM32/heap_4.c in line 367 is not initialized when it is used by pxNextFreeBlock at Arduino_STM32/heap_4.c in line 367.

	Source	Destination
File	Arduino_STM32/heap_4.c	Arduino_STM32/heap_4.c

Line	398	404
Object	pxNextFreeBlock	pxNextFreeBlock

Code Snippet
File Name Arduino_STM32/heap_4.c
Method static void prvHeapInit(void)

```
....
398.        pxEnd->pxNextFreeBlock = NULL;
....
404.        pxFirstFreeBlock->pxNextFreeBlock = pxEnd;
```

Use of Zero Initialized Pointer\Path 5:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=78
Status New

The variable declared in pxNextFreeBlock at Arduino_STM32/heap_5.c in line 431 is not initialized when it is used by pxNextFreeBlock at Arduino_STM32/heap_5.c in line 431.

	Source	Destination
File	Arduino_STM32/heap_5.c	Arduino_STM32/heap_5.c
Line	491	498
Object	pxNextFreeBlock	pxNextFreeBlock

Code Snippet
File Name Arduino_STM32/heap_5.c
Method void vPortDefineHeapRegions(const HeapRegion_t * const pxHeapRegions)

```
....
491.            pxEnd->pxNextFreeBlock = NULL;
....
498.            pxFirstFreeBlockInRegion->pxNextFreeBlock = pxEnd;
```

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://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=72

Status	New

	Source	Destination
File	Arduino_STM32/heap_3.c	Arduino_STM32/heap_3.c
Line	102	102
Object	pvReturn	pvReturn

Code Snippet
File Name Arduino_STM32/heap_3.c
Method void *pvPortMalloc(size_t xWantedSize)

```
....
102.              pvReturn = malloc( xWantedSize );
```

Use After Free

Query Path:
CPP\Cx\CPP Medium Threat\Use After Free Version:1

Categories

NIST SP 800-53: SC-5 Denial of Service Protection (P1)
OWASP Top 10 2017: A1-Injection

Description
Use After Free\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=73
Status	New

The pointer pv at Arduino_STM32/heap_3.c in line 121 is being used after it has been freed.

	Source	Destination
File	Arduino_STM32/heap_3.c	Arduino_STM32/heap_3.c
Line	127	128
Object	pv	pv

Code Snippet
File Name Arduino_STM32/heap_3.c
Method void vPortFree(void *pv)

```
....
127.                    free( pv );
128.                    traceFREE( pv, 0 );
```

NULL Pointer Dereference

Query Path:
CPP\Cx\CPP Low Visibility\NULL Pointer Dereference Version:1

Categories

NIST SP 800-53: SC-5 Denial of Service Protection (P1)
OWASP Top 10 2017: A1-Injection

Description

NULL Pointer Dereference\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=2
Status	New

The variable declared in 0 at Arduino_STM32/heap_2.c in line 154 is not initialized when it is used by xBlockSize at Arduino_STM32/heap_2.c in line 154.

	Source	Destination
File	Arduino_STM32/heap_2.c	Arduino_STM32/heap_2.c
Line	166	166
Object	0	xBlockSize

Code Snippet
File Name Arduino_STM32/heap_2.c
Method void *pvPortMalloc(size_t xWantedSize)

```
....
166.                    prvHeapInit();
```

NULL Pointer Dereference\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=3
Status	New

The variable declared in 0 at Arduino_STM32/heap_4.c in line 367 is not initialized when it is used by xBlockSize at Arduino_STM32/heap_4.c in line 367.

	Source	Destination
File	Arduino_STM32/heap_4.c	Arduino_STM32/heap_4.c
Line	389	389
Object	0	xBlockSize

Code Snippet
File Name Arduino_STM32/heap_4.c
Method static void prvHeapInit(void)

```
....
389.        xStart.xBlockSize = ( size_t ) 0;
```

NULL Pointer Dereference\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=4
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	403
Object	dfu_root	dev

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
403.        ret = libusb_open(dfu_root->dev, &dfu_root->dev_handle);
```

NULL Pointer Dereference\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=5
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by next at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	390

Object	dfu_root	next

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
390.         } else if (dfu_root->next != NULL) {
```

NULL Pointer Dereference\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=6
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	403
Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
403.         ret = libusb_open(dfu_root->dev, &dfu_root->dev_handle);
```

NULL Pointer Dereference\Path 6:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=7
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	404
Object	dfu_root	dev_handle

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
404.        if (ret || !dfu_root->dev_handle)
```

NULL Pointer Dereference\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=8
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by vendor at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	407
Object	dfu_root	vendor

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
407.        printf("ID %04x:%04x\n", dfu_root->vendor, dfu_root->product);
```

NULL Pointer Dereference\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=9
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by product at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	407
Object	dfu_root	product

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
407.        printf("ID %04x:%04x\n", dfu_root->vendor, dfu_root->product);
```

NULL Pointer Dereference\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=10

Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by func_dfu at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	410
Object	dfu_root	func_dfu

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
410.               libusb_le16_to_cpu(dfu_root-
>func_dfu.bcdDFUVersion));
```

NULL Pointer Dereference\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=11
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by flags at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	413
Object	dfu_root	flags

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
413.        if (!(dfu_root->flags & DFU_IFF_DFU)) {
```

NULL Pointer Dereference\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=12
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by vendor at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	420
Object	dfu_root	vendor

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
420.              runtime_vendor = dfu_root->vendor;
```

NULL Pointer Dereference\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=13
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by product at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	421
Object	dfu_root	product

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
421.              runtime_product = dfu_root->product;
```

NULL Pointer Dereference\Path 13:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=14
Status New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	424
Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
424.              if (libusb_claim_interface(dfu_root->dev_handle,
dfu_root->interface) < 0) {
```

NULL Pointer Dereference\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=15
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	424
Object	dfu_root	interface

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
424.              if (libusb_claim_interface(dfu_root->dev_handle,
dfu_root->interface) < 0) {
```

NULL Pointer Dereference\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=16
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c

Line	49	426
Object	dfu_root	interface

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
426.                          dfu_root->interface);
```

NULL Pointer Dereference\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=17
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	429
Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
429.              if (libusb_set_interface_alt_setting(dfu_root-
>dev_handle, dfu_root->interface, 0) < 0) {
```

NULL Pointer Dereference\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=18
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	429
Object	dfu_root	interface

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
429.              if (libusb_set_interface_alt_setting(dfu_root-
>dev_handle, dfu_root->interface, 0) < 0) {
```

NULL Pointer Dereference\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=19
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	455
Object	dfu_root	dev_handle

Code Snippet

File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
455.                    if (dfu_detach(dfu_root->dev_handle,
```

NULL Pointer Dereference\Path 19:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=20
Status New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	456
Object	dfu_root	interface

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
456.                                dfu_root->interface, 1000) < 0) {
```

NULL Pointer Dereference\Path 20:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=21

Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by func_dfu at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	459
Object	dfu_root	func_dfu

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
459.                    if (dfu_root->func_dfu.bmAttributes &
USB_DFU_WILL_DETACH) {
```

NULL Pointer Dereference\Path 21:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=22
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	463
Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
463.                            ret = libusb_reset_device(dfu_root-
>dev_handle);
```

NULL Pointer Dereference\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=23
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	471
Object	dfu_root	dev_handle

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
471.                    if (dfu_clear_status(dfu_root->dev_handle,
```

NULL Pointer Dereference\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=24
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	472
Object	dfu_root	interface

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
472.                                    dfu_root->interface) < 0) {
```

NULL Pointer Dereference\Path 24:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=25
Status New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	478
Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
478.                    libusb_release_interface(dfu_root->dev_handle,
```

NULL Pointer Dereference\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=26
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	479
Object	dfu_root	interface

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
479.                        dfu_root->interface);
```

NULL Pointer Dereference\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=27
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	482

Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
482.              libusb_release_interface(dfu_root->dev_handle,
```

NULL Pointer Dereference\Path 27:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=28
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	483
Object	dfu_root	interface

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
483.                                  dfu_root->interface);
```

NULL Pointer Dereference\Path 28:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=29
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	484
Object	dfu_root	dev_handle

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
484.              libusb_close(dfu_root->dev_handle);
```

NULL Pointer Dereference\Path 29:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=30
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	485
Object	dfu_root	dev_handle

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
485.              dfu_root->dev_handle = NULL;
```

NULL Pointer Dereference\Path 30:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=31
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by next at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	505
Object	dfu_root	next

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
505.              } else if (dfu_root->next != NULL) {
```

NULL Pointer Dereference\Path 31:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=32
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by flags at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	512
Object	dfu_root	flags

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
512.              if (!(dfu_root->flags | DFU_IFF_DFU))
```

NULL Pointer Dereference\Path 32:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=33
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	516
Object	dfu_root	dev

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method	int main(int argc, char **argv)

```
....
516.              ret = libusb_open(dfu_root->dev, &dfu_root-
>dev_handle);
```

NULL Pointer Dereference\Path 33:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=34
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	516
Object	dfu_root	dev_handle

Code Snippet

File Name	Arduino_STM32/main.c
Method	struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name	Arduino_STM32/main.c
Method	int main(int argc, char **argv)

```
....
516.              ret = libusb_open(dfu_root->dev, &dfu_root-
>dev_handle);
```

NULL Pointer Dereference\Path 34:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=35
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination

File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	517
Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
517.              if (ret || !dfu_root->dev_handle) {
```

NULL Pointer Dereference\Path 35:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=36
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by vendor at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	525
Object	dfu_root	vendor

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
525.              runtime_vendor = match_vendor < 0 ? dfu_root->vendor :
match_vendor;
```

NULL Pointer Dereference\Path 36:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=37
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by product at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	526
Object	dfu_root	product

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
526.              runtime_product = match_product < 0 ? dfu_root-
>product : match_product;
```

NULL Pointer Dereference\Path 37:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=38
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c

Line	49	537
Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
537.        if (libusb_claim_interface(dfu_root->dev_handle, dfu_root->interface) < 0) {
```

NULL Pointer Dereference\Path 38:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=39
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	537
Object	dfu_root	interface

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
537.        if (libusb_claim_interface(dfu_root->dev_handle, dfu_root->interface) < 0) {
```

NULL Pointer Dereference\Path 39:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=40
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by altsetting at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	541
Object	dfu_root	altsetting

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
541.        printf("Setting Alternate Setting #%d ...\n", dfu_root->altsetting);
```

NULL Pointer Dereference\Path 40:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=41
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	542
Object	dfu_root	dev_handle

Code Snippet

File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
542.        if (libusb_set_interface_alt_setting(dfu_root->dev_handle,
dfu_root->interface, dfu_root->altsetting) < 0) {
```

NULL Pointer Dereference\Path 41:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=42
Status New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	542
Object	dfu_root	interface

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
542.        if (libusb_set_interface_alt_setting(dfu_root->dev_handle,
dfu_root->interface, dfu_root->altsetting) < 0) {
```

NULL Pointer Dereference\Path 42:

Severity Low
Result State To Verify
Online Results http://WIN-

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=43
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by altsetting at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	542
Object	dfu_root	altsetting

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
542.        if (libusb_set_interface_alt_setting(dfu_root->dev_handle,
dfu_root->interface, dfu_root->altsetting) < 0) {
```

NULL Pointer Dereference\Path 43:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=44
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	563
Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
563.              if (dfu_clear_status(dfu_root->dev_handle, dfu_root->interface) < 0) {
```

NULL Pointer Dereference\Path 44:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=45
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	563
Object	dfu_root	interface

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
563.              if (dfu_clear_status(dfu_root->dev_handle, dfu_root->interface) < 0) {
```

NULL Pointer Dereference\Path 45:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=46

Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	571
Object	dfu_root	dev_handle

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
571.              if (dfu_abort(dfu_root->dev_handle, dfu_root->interface) < 0) {
```

NULL Pointer Dereference\Path 46:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=47
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	571
Object	dfu_root	interface

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
571.              if (dfu_abort(dfu_root->dev_handle, dfu_root->interface) < 0) {
```

NULL Pointer Dereference\Path 47:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=48
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by dev_handle at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	587
Object	dfu_root	dev_handle

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
587.              if (dfu_clear_status(dfu_root->dev_handle, dfu_root->interface) < 0)
```

NULL Pointer Dereference\Path 48:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=49
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by interface at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	587
Object	dfu_root	interface

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
587.              if (dfu_clear_status(dfu_root->dev_handle, dfu_root->interface) < 0)
```

NULL Pointer Dereference\Path 49:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=50
Status New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by func_dfu at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	598
Object	dfu_root	func_dfu

Code Snippet
File Name Arduino_STM32/main.c
Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
598.                libusb_le16_to_cpu(dfu_root-
>func_dfu.bcdDFUVersion));
```

NULL Pointer Dereference\Path 50:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=51
Status	New

The variable declared in dfu_root at Arduino_STM32/main.c in line 49 is not initialized when it is used by func_dfu at Arduino_STM32/main.c in line 237.

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	49	600
Object	dfu_root	func_dfu

Code Snippet

File Name Arduino_STM32/main.c

Method struct dfu_if *dfu_root = NULL;

```
....
49.  struct dfu_if *dfu_root = NULL;
```

File Name Arduino_STM32/main.c

Method int main(int argc, char **argv)

```
....
600.        if (dfu_root->func_dfu.bcdDFUVersion ==
libusb_cpu_to_le16(0x11a))
```

Improper Resource Access Authorization

Query Path:

CPP\Cx\CPP Low Visibility\Improper Resource Access Authorization Version:1

Categories

FISMA 2014: Identification And Authentication
NIST SP 800-53: AC-3 Access Enforcement (P1)
OWASP Top 10 2017: A2-Broken Authentication

Description

Improper Resource Access Authorization\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=79
Status	New

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	350	350
Object	fprintf	fprintf

Code Snippet
File Name Arduino_STM32/main.c
Method int main(int argc, char **argv)

```
....
350.                fprintf(stderr, "You need to specify one of -D or -
U\n");
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=80
Status	New

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	175	175
Object	fprintf	fprintf

Code Snippet
File Name Arduino_STM32/main.c
Method static void help(void)

```
....
175.        fprintf(stderr, "Usage: dfu-util [options] ...\n"
```

Improper Resource Access Authorization\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=81
Status	New

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c

Line	180	180
Object	fprintf	fprintf

Code Snippet
File Name Arduino_STM32/main.c
Method static void help(void)

```
....
180.        fprintf(stderr, "  -e --detach\t\t\tDetach currently
attached DFU capable devices\n"
```

Improper Resource Access Authorization\Path 4:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=82
Status New

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	191	191
Object	fprintf	fprintf

Code Snippet
File Name Arduino_STM32/main.c
Method static void help(void)

```
....
191.        fprintf(stderr, "  -t --transfer-size <size>\tSpecify the
number of bytes per USB Transfer\n"
```

Inconsistent Implementations

Query Path:
CPP\Cx\CPP Low Visibility\Inconsistent Implementations Version:0

Description

Inconsistent Implementations\Path 1:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050082&projectid=50072&pathid=1
Status New

	Source	Destination
File	Arduino_STM32/main.c	Arduino_STM32/main.c
Line	262	262
Object	getopt_long	getopt_long

Code Snippet
File Name Arduino_STM32/main.c
Method int main(int argc, char **argv)

```
....
262.              c = getopt_long(argc, argv,
"hVvleE:d:p:c:i:a:S:t:U:D:Rs:Z:", opts,
```

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;
}
```

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

Use After Free

Risk

What might happen

A use after free error will cause code to use an area of memory previously assigned with a specific value, which has since been freed and may have been overwritten by another value. This error will likely cause unexpected behavior, memory corruption and crash errors. In some cases where the freed and used section of memory is used to determine execution flow, and the error can be induced by an attacker, this may result in execution of malicious code.

Cause

How does it happen

Pointers to variables allow code to have an address with a set size to a dynamically allocated variable. Eventually, the pointer's destination may become free - either explicitly in code, such as when programmatically freeing this variable, or implicitly, such as when a local variable is returned - once it is returned, the variable's scope is released. Once freed, this memory will be re-used by the application, overwritten with new data. At this point, dereferencing this pointer will potentially resolve newly written and unexpected data.

General Recommendations

How to avoid it

- Do not return local variables or pointers
- Review code to ensure no flow allows use of a pointer after it has been explicitly freed

---

Source Code Examples

CPP

Use of Variable after It was Freed

```
free(input);
printf("%s", input);
```

Use of Pointer to Local Variable That Was Freed On Return

```
int* func1()
{
	int i;
	i = 1;
	return &i;
}

void func2()

{

	int j;
	j = 5;
```

```
}

//..
      int * i = func1();
      printf("%d\r\n", *i); // Output could be 1 or Segmentation Fault
      func2();
      printf("%d\r\n", *i); // Output is 5, which is j's value, as func2() overwrote data in
the stack
//..
```

Use of Zero Initialized Pointer

Risk

What might happen

A null pointer dereference is likely to cause a run-time exception, a crash, or other unexpected behavior.

Cause

How does it happen

Variables which are declared without being assigned will implicitly retain a null value until they are assigned. The null value can also be explicitly set to a variable, to ensure clear out its contents. Since null is not really a value, it may not have object variables and methods, and any attempt to access contents of a null object, instead of verifying it is set beforehand, will result in a null pointer dereference exception.

General Recommendations

How to avoid it

- For any variable that is created, ensure all logic flows between declaration and use assign a non-null value to the variable first.
- Enforce null checks on any received variable or object before it is dereferenced, to ensure it does not contain a null assigned to it elsewhere.
- Consider the need to assign null values in order to overwrite initialized variables. Consider reassigning or releasing these variables instead.

Source Code Examples

CPP

Explicit NULL Dereference

```
char * input = NULL;
printf("%s", input);
```

Implicit NULL Dereference

```
char * input;
printf("%s", input);
```

Java

Explicit Null Dereference

```
Object o = null;
out.println(o.getClass());
```

Use of Function with Inconsistent Implementations

Weakness ID: 474 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The code uses a function that has inconsistent implementations across operating systems and versions, which might cause security-relevant portability problems.

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C: *(Often)*

PHP: *(Often)*

All

Potential Mitigations

Do not accept inconsistent behavior from the API specifications when the deviant behavior increase the risk level.

Other Notes

The behavior of functions in this category varies by operating system, and at times, even by operating system version. Implementation differences can include:

- Slight differences in the way parameters are interpreted leading to inconsistent results.
- Some implementations of the function carry significant security risks.
- The function might not be defined on all platforms.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	398	Indicator of Poor Code Quality	**Development Concepts (primary)699 Seven Pernicious Kingdoms (primary)700 Research Concepts (primary)1000**
ParentOf	Weakness Variant	589	Call to Non-ubiquitous API	**Research Concepts (primary)1000**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
7 Pernicious Kingdoms			Inconsistent Implementations

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	7 Pernicious Kingdoms		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Eric Dalci	Cigital	External
	updated Potential Mitigations, Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Relationships, Other Notes, Taxonomy Mappings		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2008-04-11	Inconsistent Implementations		

BACK TO TOP

NULL Pointer Dereference

Risk

What might happen

A null pointer dereference is likely to cause a run-time exception, a crash, or other unexpected behavior.

Cause

How does it happen

Variables which are declared without being assigned will implicitly retain a null value until they are assigned. The null value can also be explicitly set to a variable, to ensure clear out its contents. Since null is not really a value, it may not have object variables and methods, and any attempt to access contents of a null object, instead of verifying it is set beforehand, will result in a null pointer dereference exception.

General Recommendations

How to avoid it

- For any variable that is created, ensure all logic flows between declaration and use assign a non-null value to the variable first.
- Enforce null checks on any received variable or object before it is dereferenced, to ensure it does not contain a null assigned to it elsewhere.
- Consider the need to assign null values in order to overwrite initialized variables. Consider reassigning or releasing these variables instead.

Source Code Examples

Improper Access Control (Authorization)

Weakness ID: 285 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software does not perform or incorrectly performs access control checks across all potential execution paths.

Extended Description

When access control checks are not applied consistently - or not at all - users are able to access data or perform actions that they should not be allowed to perform. This can lead to a wide range of problems, including information leaks, denial of service, and arbitrary code execution.

Alternate Terms

AuthZ:	"AuthZ" is typically used as an abbreviation of "authorization" within the web application security community. It is also distinct from "AuthC," which is an abbreviation of "authentication." The use of "Auth" as an abbreviation is discouraged, since it could be used for either authentication or authorization.

Time of Introduction

- Architecture and Design
- Implementation
- Operation

Applicable Platforms

Languages

Language-independent

Technology Classes

Web-Server: *(Often)*

Database-Server: *(Often)*

Modes of Introduction

A developer may introduce authorization weaknesses because of a lack of understanding about the underlying technologies. For example, a developer may assume that attackers cannot modify certain inputs such as headers or cookies.

Authorization weaknesses may arise when a single-user application is ported to a multi-user environment.

Common Consequences

Scope	Effect
Confidentiality	An attacker could read sensitive data, either by reading the data directly from a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to read the data.
Integrity	An attacker could modify sensitive data, either by writing the data directly to a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to write the data.
Integrity	An attacker could gain privileges by modifying or reading critical data directly, or by accessing insufficiently-protected, privileged functionality.

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

Automated static analysis is useful for detecting commonly-used idioms for authorization. A tool may be able to analyze related configuration files, such as .htaccess in Apache web servers, or detect the usage of commonly-used authorization libraries.

Generally, automated static analysis tools have difficulty detecting custom authorization schemes. In addition, the software's design may include some functionality that is accessible to any user and does not require an authorization check; an automated technique that detects the absence of authorization may report false positives.

Effectiveness: Limited

Automated Dynamic Analysis

Automated dynamic analysis may find many or all possible interfaces that do not require authorization, but manual analysis is required to determine if the lack of authorization violates business logic

Manual Analysis

This weakness can be detected using tools and techniques that require manual (human) analysis, such as penetration testing, threat modeling, and interactive tools that allow the tester to record and modify an active session.

Specifically, manual static analysis is useful for evaluating the correctness of custom authorization mechanisms.

Effectiveness: Moderate

These may be more effective than strictly automated techniques. This is especially the case with weaknesses that are related to design and business rules. However, manual efforts might not achieve desired code coverage within limited time constraints.

Demonstrative Examples

Example 1

The following program could be part of a bulletin board system that allows users to send private messages to each other. This program intends to authenticate the user before deciding whether a private message should be displayed. Assume that LookupMessageObject() ensures that the $id argument is numeric, constructs a filename based on that id, and reads the message details from that file. Also assume that the program stores all private messages for all users in the same directory.

(Bad Code)

Example Language: **Perl**

```
sub DisplayPrivateMessage {
my($id) = @_;
my $Message = LookupMessageObject($id);
print "From: " . encodeHTML($Message->{from}) . "<br>\n";
print "Subject: " . encodeHTML($Message->{subject}) . "\n";
print "<hr>\n";
print "Body: " . encodeHTML($Message->{body}) . "\n";
}

my $q = new CGI;
# For purposes of this example, assume that CWE-309 and
# CWE-523 do not apply.
if (! AuthenticateUser($q->param('username'), $q->param('password'))) {
ExitError("invalid username or password");
}

my $id = $q->param('id');
DisplayPrivateMessage($id);
```

While the program properly exits if authentication fails, it does not ensure that the message is addressed to the user. As a result, an authenticated attacker could provide any arbitrary identifier and read private messages that were intended for other users.

One way to avoid this problem would be to ensure that the "to" field in the message object matches the username of the authenticated user.

Observed Examples

Reference	Description
CVE-2009-3168	Web application does not restrict access to admin scripts, allowing authenticated users to reset administrative passwords.

CVE-2009-2960	Web application does not restrict access to admin scripts, allowing authenticated users to modify passwords of other users.
CVE-2009-3597	Web application stores database file under the web root with insufficient access control (CWE-219), allowing direct request.
CVE-2009-2282	Terminal server does not check authorization for guest access.
CVE-2009-3230	Database server does not use appropriate privileges for certain sensitive operations.
CVE-2009-2213	Gateway uses default "Allow" configuration for its authorization settings.
CVE-2009-0034	Chain: product does not properly interpret a configuration option for a system group, allowing users to gain privileges.
CVE-2008-6123	Chain: SNMP product does not properly parse a configuration option for which hosts are allowed to connect, allowing unauthorized IP addresses to connect.
CVE-2008-5027	System monitoring software allows users to bypass authorization by creating custom forms.
CVE-2008-7109	Chain: reliance on client-side security (CWE-602) allows attackers to bypass authorization using a custom client.
CVE-2008-3424	Chain: product does not properly handle wildcards in an authorization policy list, allowing unintended access.
CVE-2009-3781	Content management system does not check access permissions for private files, allowing others to view those files.
CVE-2008-4577	ACL-based protection mechanism treats negative access rights as if they are positive, allowing bypass of intended restrictions.
CVE-2008-6548	Product does not check the ACL of a page accessed using an "include" directive, allowing attackers to read unauthorized files.
CVE-2007-2925	Default ACL list for a DNS server does not set certain ACLs, allowing unauthorized DNS queries.
CVE-2006-6679	Product relies on the X-Forwarded-For HTTP header for authorization, allowing unintended access by spoofing the header.
CVE-2005-3623	OS kernel does not check for a certain privilege before setting ACLs for files.
CVE-2005-2801	Chain: file-system code performs an incorrect comparison (CWE-697), preventing defauls ACLs from being properly applied.
CVE-2001-1155	Chain: product does not properly check the result of a reverse DNS lookup because of operator precedence (CWE-783), allowing bypass of DNS-based access restrictions.

Potential Mitigations

Phase: Architecture and Design

Divide your application into anonymous, normal, privileged, and administrative areas. Reduce the attack surface by carefully mapping roles with data and functionality. Use role-based access control (RBAC) to enforce the roles at the appropriate boundaries.

Note that this approach may not protect against horizontal authorization, i.e., it will not protect a user from attacking others with the same role.

Phase: Architecture and Design

Ensure that you perform access control checks related to your business logic. These checks may be different than the access control checks that you apply to more generic resources such as files, connections, processes, memory, and database records. For example, a database may restrict access for medical records to a specific database user, but each record might only be intended to be accessible to the patient and the patient's doctor.

Phase: Architecture and Design

Strategy: Libraries or Frameworks

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness

easier to avoid.

For example, consider using authorization frameworks such as the JAAS Authorization Framework and the OWASP ESAPI Access Control feature.

Phase: Architecture and Design

For web applications, make sure that the access control mechanism is enforced correctly at the server side on every page. Users should not be able to access any unauthorized functionality or information by simply requesting direct access to that page.

One way to do this is to ensure that all pages containing sensitive information are not cached, and that all such pages restrict access to requests that are accompanied by an active and authenticated session token associated with a user who has the required permissions to access that page.

Phases: System Configuration; Installation

Use the access control capabilities of your operating system and server environment and define your access control lists accordingly. Use a "default deny" policy when defining these ACLs.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	254	Security Features	**Seven Pernicious Kingdoms (primary)700**
ChildOf	Weakness Class	284	Access Control (Authorization) Issues	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	721	OWASP Top Ten 2007 Category A10 - Failure to Restrict URL Access	**Weaknesses in OWASP Top Ten (2007) (primary)629**
ChildOf	Category	723	OWASP Top Ten 2004 Category A2 - Broken Access Control	**Weaknesses in OWASP Top Ten (2004) (primary)711**
ChildOf	Category	753	2009 Top 25 - Porous Defenses	**Weaknesses in the 2009 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)750**
ChildOf	Category	803	2010 Top 25 - Porous Defenses	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
ParentOf	Weakness Variant	219	Sensitive Data Under Web Root	**Research Concepts (primary)1000**
ParentOf	Weakness Base	551	Incorrect Behavior Order: Authorization Before Parsing and Canonicalization	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Class	638	Failure to Use Complete Mediation	Research Concepts1000
ParentOf	Weakness Base	804	Guessable CAPTCHA	**Development Concepts (primary)699 Research Concepts (primary)1000**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
7 Pernicious Kingdoms			Missing Access Control
OWASP Top Ten 2007	A10	CWE More Specific	Failure to Restrict URL Access
OWASP Top Ten 2004	A2	CWE More Specific	Broken Access Control

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
1	Accessing Functionality Not Properly Constrained by ACLs	
13	Subverting Environment Variable Values	

17	Accessing, Modifying or Executing Executable Files
87	Forceful Browsing
39	Manipulating Opaque Client-based Data Tokens
45	Buffer Overflow via Symbolic Links
51	Poison Web Service Registry
59	Session Credential Falsification through Prediction
60	Reusing Session IDs (aka Session Replay)
77	Manipulating User-Controlled Variables
76	Manipulating Input to File System Calls
104	Cross Zone Scripting

References

NIST. "Role Based Access Control and Role Based Security". <http://csrc.nist.gov/groups/SNS/rbac/>.

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 4, "Authorization" Page 114; Chapter 6, "Determining Appropriate Access Control" Page 171. 2nd Edition. Microsoft. 2002.

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	7 Pernicious Kingdoms		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-15		Veracode	External
	Suggested OWASP Top Ten 2004 mapping		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Relationships, Other Notes, Taxonomy Mappings		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Common Consequences, Description, Likelihood of Exploit, Name, Other Notes, Potential Mitigations, References, Relationships		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Potential Mitigations		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Description, Related Attack Patterns		
2009-07-27	CWE Content Team	MITRE	Internal
	updated Relationships		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Type		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Demonstrative Examples, Detection Factors, Modes of Introduction, Observed Examples, Relationships		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Detection Factors, Potential Mitigations, References, Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Potential Mitigations		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2009-01-12	Missing or Inconsistent Access Control		

BACK TO TOP

Scanned Languages

Language	Hash Number	Change Date
CPP	4541647240435660	6/19/2024
Common	0105849645654507	6/19/2024