sdlpal Scan Report

Project Name	sdlpal
Scan Start	Friday, June 21, 2024 10:40:30 PM
Preset	Checkmarx Default
Scan Time	00h:03m:02s
Lines Of Code Scanned	14303
Files Scanned	9
Report Creation Time	Friday, June 21, 2024 10:46:12 PM
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044
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

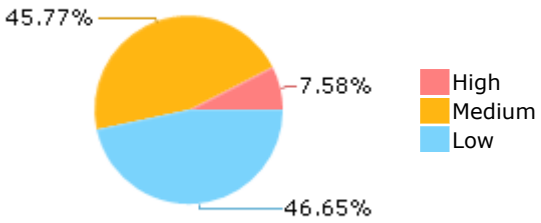
Results Limit

Results limit per query was set to 50

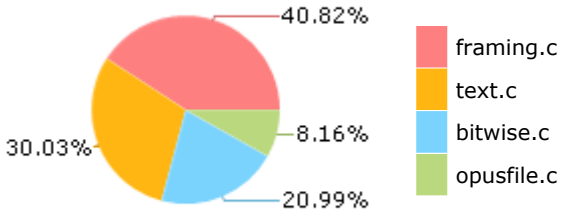
Selected Queries

Selected queries are listed in Result Summary

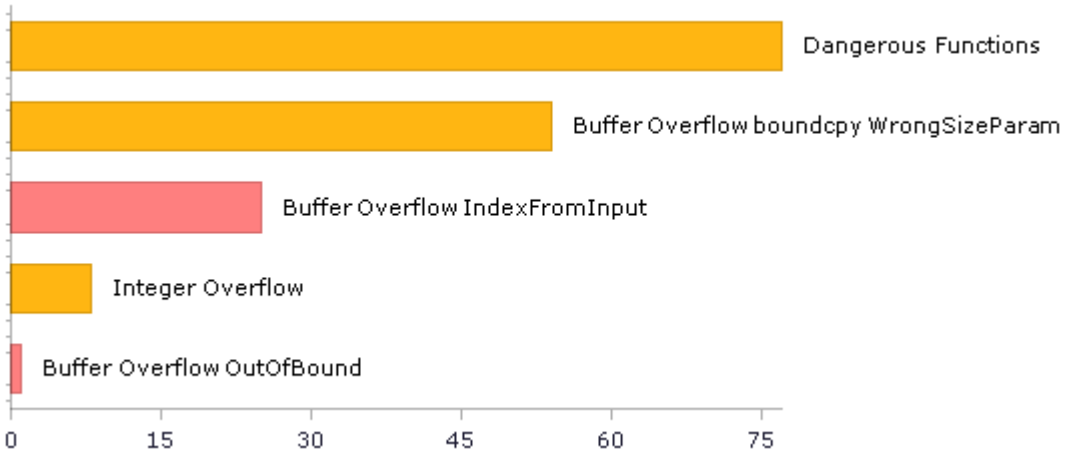
Result Summary



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	103	64
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	135	135
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	77	77
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	77	77
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	64	64
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.	1	1
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.	135	135
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.	8	8

***** 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)	135	135
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)*	19	9
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*	22	15
SI-11 Error Handling (P2)*	1	1
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

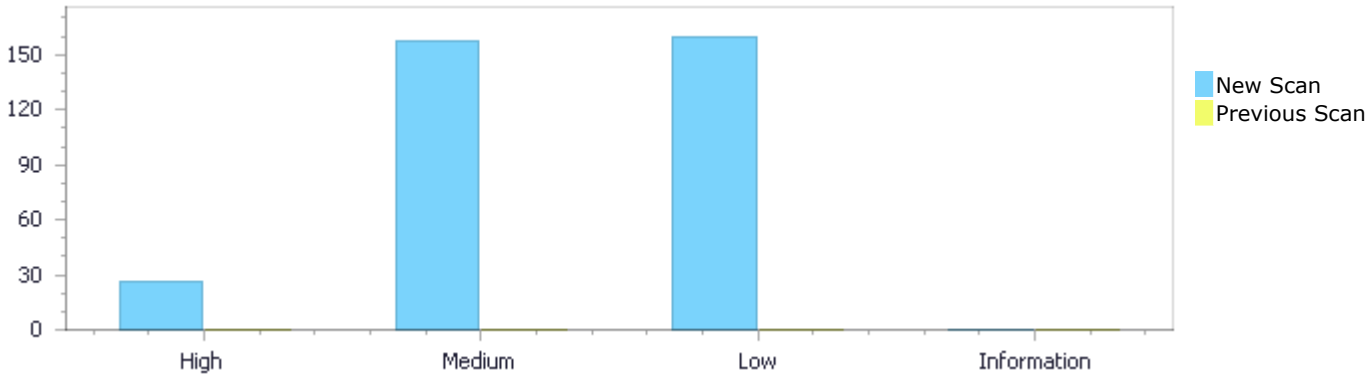
Scan Summary - Custom

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	26	157	160	0	343
Recurrent Issues	0	0	0	0	0
Total	26	157	160	0	343
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	26	157	160	0	343
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	26	157	160	0	343

Result Summary

Vulnerability Type	Occurrences	Severity
Buffer Overflow IndexFromInput	25	High
Buffer Overflow OutOfBound	1	High
Dangerous Functions	77	Medium
Buffer Overflow boundcpy WrongSizeParam	54	Medium
Integer Overflow	8	Medium

Stored Buffer Overflow boundcpy	8	Medium
Wrong Size t Allocation	4	Medium
Use of Zero Initialized Pointer	3	Medium
Memory Leak	2	Medium
Divide By Zero	1	Medium
Improper Resource Access Authorization	135	Low
NULL Pointer Dereference	13	Low
Use of Sizeof On a Pointer Type	5	Low
Unchecked Array Index	3	Low
Arithmenic Operation On Boolean	1	Low
Heuristic 2nd Order Buffer Overflow read	1	Low
Potential Precision Problem	1	Low
Unchecked Return Value	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
sdlpal/text.c	87
sdlpal/framing.c	73
sdlpal/opusfile.c	23

Scan Results Details

Buffer Overflow IndexFromInput

Query Path:
CPP\Cx\CPP Buffer Overflow\Buffer Overflow IndexFromInput Version:1

Categories

OWASP Top 10 2017: A1-Injection

Description

Buffer Overflow IndexFromInput\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=1
Status	New

The size of the buffer used by PAL_ReadMessageFile in oldCount, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	583
Object	temp	oldCount

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item->index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item->index] - oldCount));
```

Buffer Overflow IndexFromInput\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=2
Status	New

The size of the buffer used by PAL_ReadMessageFile in oldCount, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	583
Object	BinaryExpr	oldCount

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item-
>index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item-
>index] - oldCount));
```

Buffer Overflow IndexFromInput\Path 3:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=3
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	517
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
517.                                                    while (w < limit && j <
len - 1) w += PAL_CharWidth(g_rcCredits[i][j++]);
```

Buffer Overflow IndexFromInput\Path 4:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=4
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	517
Object	BinaryExpr	i

Code Snippet

File Name sdlpal/text.c

Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
517.                                                    while (w < limit && j <
len - 1) w += PAL_CharWidth(g_rcCredits[i][j++]);
```

Buffer Overflow IndexFromInput\Path 5:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=5
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	514
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
514.                                    if (g_rcCredits[i])
```

Buffer Overflow IndexFromInput\Path 6:

Severity High
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=6
Status New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	514
Object	BinaryExpr	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c

Method	PAL_ReadMessageFile(

```
....
514.                                        if (g_rcCredits[i])
```

Buffer Overflow IndexFromInput\Path 7:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=7
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	505
Object	temp	i

Code Snippet

File Name	sdlpal/text.c
Method	PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name	sdlpal/text.c
Method	PAL_ReadMessageFile(

```
....
505.
      PAL_MultiByteToWideCharCP(CP_UTF_8, tmp, -1, g_rcCredits[i], len);
```

Buffer Overflow IndexFromInput\Path 8:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=8
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination

File	sdlpal/text.c	sdlpal/text.c
Line	131	505
Object	BinaryExpr	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
505.
      PAL_MultiByteToWideCharCP(CP_UTF_8, tmp, -1, g_rcCredits[i], len);
```

Buffer Overflow IndexFromInput\Path 9:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=9
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	504
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
504.                                                    g_rcCredits[i] =
(wchar_t *)UTIL_malloc(len * sizeof(wchar_t));
```

Buffer Overflow IndexFromInput\Path 10:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=10
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	504
Object	BinaryExpr	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
504.                                                    g_rcCredits[i] =
(wchar_t *)UTIL_malloc(len * sizeof(wchar_t));
```

Buffer Overflow IndexFromInput\Path 11:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=11
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c

Line	117	512
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
512.
      PAL_MultiByteToWideCharCP(CP_UTF_8, v, -1, g_rcCredits[i], len);
```

Buffer Overflow IndexFromInput\Path 12:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=12
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	512
Object	BinaryExpr	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                          if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
512.
      PAL_MultiByteToWideCharCP(CP_UTF_8, v, -1, g_rcCredits[i], len);
```

Buffer Overflow IndexFromInput\Path 13:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=13
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	511
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
511.                                                g_rcCredits[i] =
(wchar_t *)UTIL_malloc(len * sizeof(wchar_t));
```

Buffer Overflow IndexFromInput\Path 14:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=14
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c

Line	131	511
Object	BinaryExpr	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
511.                                                    g_rcCredits[i] =
(wchar_t *)UTIL_malloc(len * sizeof(wchar_t));
```

Buffer Overflow IndexFromInput\Path 15:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=15
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	473
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
473.                                    if ((i == 1 || (i >= 6 && i <= 11))
&& !g_rcCredits[i])
```

Buffer Overflow IndexFromInput\Path 16:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=16
Status	New

The size of the buffer used by PAL_ReadMessageFile in i, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	473
Object	BinaryExpr	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
473.                                    if ((i == 1 || (i >= 6 && i <= 11))
&& !g_rcCredits[i])
```

Buffer Overflow IndexFromInput\Path 17:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=17
Status	New

The size of the buffer used by PAL_MultiByteToWideCharCP in i, at line 1956 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c

Line	117	2070
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c

Method PAL_MultiByteToWideCharCP(

```
....
2070.             if (i < mbslength && !mbs[i]) null = 1;
```

Buffer Overflow IndexFromInput\Path 18:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=18
Status	New

The size of the buffer used by PAL_MultiByteToWideCharCP in i, at line 1956 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	2070
Object	BinaryExpr	i

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c

Method PAL_MultiByteToWideCharCP(

```
....
2070.             if (i < mbslength && !mbs[i]) null = 1;
```

Buffer Overflow IndexFromInput\Path 19:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=19
Status	New

The size of the buffer used by PAL_MultiByteToWideCharCP in i, at line 1956 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_InitText passes to temp, at line 649 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	744	2070
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_InitText(

```
....
744.            if (fread(temp, 1, i, fpWord) < (size_t)i)
```

File Name sdlpal/text.c
Method PAL_MultiByteToWideCharCP(

```
....
2070.               if (i < mbslength && !mbs[i]) null = 1;
```

Buffer Overflow IndexFromInput\Path 20:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=20
Status	New

The size of the buffer used by PAL_MultiByteToWideCharCP in i, at line 1956 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_InitText passes to temp, at line 649 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	826	2070
Object	temp	i

Code Snippet
File Name sdlpal/text.c

Method PAL_InitText(

```
....
826.            if (fread(temp, 1, i, fpMsg) < (size_t)i)
```

File Name sdlpal/text.c

Method PAL_MultiByteToWideCharCP(

```
....
2070.             if (i < mbslength && !mbs[i]) null = 1;
```

Buffer Overflow IndexFromInput\Path 21:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=21
Status	New

The size of the buffer used by PAL_InitText in l, at line 649 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_InitText passes to temp, at line 649 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	744	787
Object	temp	l

Code Snippet

File Name sdlpal/text.c

Method PAL_InitText(

```
....
744.            if (fread(temp, 1, i, fpWord) < (size_t)i)
....
787.                  g_TextLib.lpWordBuf[i][l] = 0;
```

Buffer Overflow IndexFromInput\Path 22:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=22
Status	New

The size of the buffer used by PAL_InitText in i, at line 649 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_InitText passes to temp, at line 649 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination

File	sdlpal/text.c	sdlpal/text.c
Line	744	784
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_InitText(

```
....
744.           if (fread(temp, 1, i, fpWord) < (size_t)i)
....
784.                   l = PAL_MultiByteToWideChar((LPCSTR)temp + i *
gConfig.dwWordLength, gConfig.dwWordLength, g_TextLib.lpWordBuf[i], wlen
- wpos);
```

Buffer Overflow IndexFromInput\Path 23:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=23
Status	New

The size of the buffer used by PAL_InitText in i, at line 649 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_InitText passes to temp, at line 649 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	744	783
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_InitText(

```
....
744.           if (fread(temp, 1, i, fpWord) < (size_t)i)
....
783.                 g_TextLib.lpWordBuf[i] = tmp + wpos;
```

Buffer Overflow IndexFromInput\Path 24:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=24
Status	New

The size of the buffer used by PAL_InitText in i, at line 649 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_InitText passes to temp, at line 649 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	744	785
Object	temp	i

Code Snippet
File Name sdlpal/text.c
Method PAL_InitText(

```
....
744.            if (fread(temp, 1, i, fpWord) < (size_t)i)
....
785.                  if (l > 0 && g_TextLib.lpWordBuf[i][l - 1] == '1')
```

Buffer Overflow IndexFromInput\Path 25:

Severity High
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=25
Status New

The size of the buffer used by PAL_InitText in l, at line 649 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_InitText passes to temp, at line 649 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	826	864
Object	temp	l

Code Snippet
File Name sdlpal/text.c
Method PAL_InitText(

```
....
826.            if (fread(temp, 1, i, fpMsg) < (size_t)i)
....
864.                  g_TextLib.lpMsgBuf[i][l] = 0;
```

Buffer Overflow OutOfBound

Query Path:
CPP\Cx\CPP Buffer Overflow\Buffer Overflow OutOfBound Version:1

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
NIST SP 800-53: SI-10 Information Input Validation (P1)

OWASP Top 10 2017: A1-Injection

Description

Buffer Overflow OutOfBound\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=26
Status	New

The size of the buffer used by print_header in j, at line 1117 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1814	1138
Object	og	j

Code Snippet

File Name sdlpal/framing.c

Method int main(void){

```
....
1814.       ogg_page og[5];
```

File Name sdlpal/framing.c

Method void print_header(ogg_page *og){

```
....
1138.       fprintf(stderr,"%d ",(int)og->header[j]);
```

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=1050054&projectid=50044&pathid=119
Status	New

The dangerous function, alloca, was found in use at line 154 in sdlpal/text.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	484	484
Object	alloca	alloca

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
484.                                                    char *tmp = (char
*)alloca(valuelen[0] + valuelen[1] + valuelen[2] + 1 + 1);
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=120
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1852	1852
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1852.
memcpy(ogg_sync_buffer(&oy,og[i].header_len),og[i].header,
```

Dangerous Functions\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=121
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1855	1855
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1855.
memcpy(ogg_sync_buffer(&oy,og[i].body_len),og[i].body,og[i].body_len);
```

Dangerous Functions\Path 4:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=122
Status New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1901	1901
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1901.
memcpy(ogg_sync_buffer(&oy,og[i].header_len),og[i].header,
```

Dangerous Functions\Path 5:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=123
Status New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1904	1904
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1904.
memcpy(ogg_sync_buffer(&oy,og[i].body_len),og[i].body,og[i].body_len);
```

Dangerous Functions\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=124
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1954	1954
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1954.        memcpy(ogg_sync_buffer(&oy,og[1].header_len),og[1].header,
```

Dangerous Functions\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=125
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1960	1960
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1960.
memcpy(ogg_sync_buffer(&oy,og[1].header_len),og[1].header+3,
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=126
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1966	1966
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1966.
memcpy(ogg_sync_buffer(&oy,og[1].header_len),og[1].header+23,
```

Dangerous Functions\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=127
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination

File	sdlpal/framing.c	sdlpal/framing.c
Line	1973	1973
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1973.
memcpy(ogg_sync_buffer(&oy,og[1].header_len),og[1].header+28,
```

Dangerous Functions\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=128
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1978	1978
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1978.
memcpy(ogg_sync_buffer(&oy,og[1].body_len),og[1].body,1000);
```

Dangerous Functions\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=129
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c

Line	1982	1982
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1982.        memcpy(ogg_sync_buffer(&oy,og[1].body_len),og[1].body+1000,
```

Dangerous Functions\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=130
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1996	1996
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1996.        memcpy(ogg_sync_buffer(&oy,og[1].header_len),og[1].header,
```

Dangerous Functions\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=131
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2000	2000
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2000.        memcpy(ogg_sync_buffer(&oy,og[1].body_len),og[1].body,
```

Dangerous Functions\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=132
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2004	2004
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2004.        memcpy(ogg_sync_buffer(&oy,og[1].header_len),og[1].header,
```

Dangerous Functions\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=133
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2010	2010
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c

Method	int main(void){

```
....
2010.
memcpy(ogg_sync_buffer(&oy,og[1].header_len),og[1].header+20,
```

Dangerous Functions\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=134
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2013	2013
Object	memcpy	memcpy

Code Snippet	
File Name	sdlpal/framing.c
Method	int main(void){

```
....
2013.        memcpy(ogg_sync_buffer(&oy,og[1].body_len),og[1].body,
```

Dangerous Functions\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=135
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2028	2028
Object	memcpy	memcpy

Code Snippet	
File Name	sdlpal/framing.c
Method	int main(void){

```
....
2028.        memcpy(ogg_sync_buffer(&oy,og[1].body_len),og[1].body,
```

Dangerous Functions\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=136
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2032	2032
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2032.        memcpy(ogg_sync_buffer(&oy,og[1].header_len),og[1].header,
```

Dangerous Functions\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=137
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2036	2036
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2036.        memcpy(ogg_sync_buffer(&oy,og[1].body_len),og[1].body,
```

Dangerous Functions\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=138
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2040	2040
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2040.        memcpy(ogg_sync_buffer(&oy,og[2].header_len),og[2].header,
```

Dangerous Functions\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=139
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2047	2047
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2047.
memcpy(ogg_sync_buffer(&oy,og[2].header_len),og[2].header+20,
```

Dangerous Functions\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=140
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2050	2050
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2050.        memcpy(ogg_sync_buffer(&oy,og[2].body_len),og[2].body,
```

Dangerous Functions\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=141
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2064	2064
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2064.        memcpy(ogg_sync_buffer(&oy,og[1].header_len),og[1].header,
```

Dangerous Functions\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=142
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2068	2068
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2068.        memcpy(ogg_sync_buffer(&oy,og[1].body_len),og[1].body,
```

Dangerous Functions\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=143
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2072	2072
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2072.        memcpy(ogg_sync_buffer(&oy,og[2].header_len),og[2].header,
```

Dangerous Functions\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=144
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2076	2076
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2076.        memcpy(ogg_sync_buffer(&oy,og[2].header_len),og[2].header,
```

Dangerous Functions\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=145
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2082	2082
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2082.        memcpy(ogg_sync_buffer(&oy,og[2].body_len),og[2].body,
```

Dangerous Functions\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=146
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2086	2086
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2086.        memcpy(ogg_sync_buffer(&oy,og[3].header_len),og[3].header,
```

Dangerous Functions\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=147
Status	New

The dangerous function, memcpy, was found in use at line 1674 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2090	2090
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2090.         memcpy(ogg_sync_buffer(&oy,og[3].body_len),og[3].body,
```

Dangerous Functions\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=148
Status	New

The dangerous function, memcpy, was found in use at line 317 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	355	355
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int ogg_stream_iovecin(ogg_stream_state *os, ogg_iovec_t *iov, int count,

```
....
355.       memcpy(os->body_data+os->body_fill, iov[i].iov_base,
iov[i].iov_len);
```

Dangerous Functions\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=149
Status	New

The dangerous function, memcpy, was found in use at line 390 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	445	445
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method static int ogg_stream_flush_i(ogg_stream_state *os,ogg_page *og, int force, int nfill){

```
....
445.    memcpy(os->header,"OggS",4);
```

Dangerous Functions\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=150
Status	New

The dangerous function, memcpy, was found in use at line 677 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	709	709
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method long ogg_sync_pageseek(ogg_sync_state *oy,ogg_page *og){

```
....
709.      memcpy(chksum,page+22,4);
```

Dangerous Functions\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=151
Status	New

The dangerous function, memcpy, was found in use at line 677 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	724	724
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method long ogg_sync_pageseek(ogg_sync_state *oy,ogg_page *og){

```
....
724.        memcpy(page+22,chksum,4);
```

Dangerous Functions\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=152
Status	New

The dangerous function, memcpy, was found in use at line 808 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	894	894
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method int ogg_stream_pagein(ogg_stream_state *os, ogg_page *og){

```
....
894.      memcpy(os->body_data+os->body_fill,body,bodysize);
```

Dangerous Functions\Path 35:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=153
Status	New

The dangerous function, memcpy, was found in use at line 1142 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1144	1144
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method void copy_page(ogg_page *og){

```
....
1144.    memcpy(temp,og->header,og->header_len);
```

Dangerous Functions\Path 36:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=154
Status	New

The dangerous function, memcpy, was found in use at line 1142 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1148	1148
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method void copy_page(ogg_page *og){

```
....
1148.    memcpy(temp,og->body,og->body_len);
```

Dangerous Functions\Path 37:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=155
Status	New

The dangerous function, memcpy, was found in use at line 1495 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1568	1568
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method void test_pack(const int *pl, const int **headers, int byteskip,

```
....
1568.               memcpy(next,og.header,byteskipcount-byteskip);
```

Dangerous Functions\Path 38:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=156
Status	New

The dangerous function, memcpy, was found in use at line 1495 in sdlpal/framing.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1575	1575
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/framing.c
Method void test_pack(const int *pl, const int **headers, int byteskip,

```
....
1575.               memcpy(next,og.body,byteskipcount-byteskip);
```

Dangerous Functions\Path 39:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=157
Status	New

The dangerous function, memcpy, was found in use at line 73 in sdlpal/opusfile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	96	96
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/opusfile.c
Method int op_test(OpusHead *_head,

```
....
96.      memcpy(data,_initial_data,_initial_bytes);
```

Dangerous Functions\Path 40:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=158
Status	New

The dangerous function, memcpy, was found in use at line 1346 in sdlpal/opusfile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1375	1375
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_make_decode_ready(OggOpusFile *_of){

```
....
1375.      memcpy(_of->od_mapping,head->mapping,sizeof(*head-
>mapping)*channel_count);
```

Dangerous Functions\Path 41:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=159
Status	New

The dangerous function, memcpy, was found in use at line 1417 in sdlpal/opusfile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1443	1443
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_open_seekable2(OggOpusFile *_of){

```
....
1443.    memcpy(op_start,_of->op,sizeof(*op_start)*start_op_count);
```

Dangerous Functions\Path 42:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=160
Status	New

The dangerous function, memcpy, was found in use at line 1417 in sdlpal/opusfile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1455	1455
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_open_seekable2(OggOpusFile *_of){

```
....
1455.    memcpy(_of->op,op_start,sizeof(*_of->op)*start_op_count);
```

Dangerous Functions\Path 43:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=161
Status	New

The dangerous function, memcpy, was found in use at line 1504 in sdlpal/opusfile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1530	1530
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_open1(OggOpusFile *_of,

```
....
1530.       memcpy(buffer,_initial_data,_initial_bytes*sizeof(*buffer));
```

Dangerous Functions\Path 44:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=162
Status	New

The dangerous function, memcpy, was found in use at line 2803 in sdlpal/opusfile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	2819	2819
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_read_native(OggOpusFile *_of,

```
....
2819.           memcpy(_pcm,_of->od_buffer+nchannels*od_buffer_pos,
```

Dangerous Functions\Path 45:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=163
Status	New

The dangerous function, memcpy, was found in use at line 3028 in sdlpal/opusfile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	3032	3032
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_stereo_filter(OggOpusFile *_of,void *_dst,int _dst_sz,

```
....
3032.    if(_nchannels==2)memcpy(_dst,_src,_nsamples*2*sizeof(*_src));
```

Dangerous Functions\Path 46:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=164
Status	New

The dangerous function, memcpy, was found in use at line 1348 in sdlpal/text.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	1375	1375
Object	memcpy	memcpy

Code Snippet
File Name sdlpal/text.c
Method PAL_DialogWaitForKeyWithMaximumSeconds(

```
....
1375.     memcpy(palette, pCurrentPalette, sizeof(palette));
```

Dangerous Functions\Path 47:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=165
Status	New

The dangerous function, sscanf, was found in use at line 66 in sdlpal/text.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	99	99
Object	sscanf	sscanf

Code Snippet
File Name sdlpal/text.c
Method PAL_ParseLine(

```
....
99.                 if (sscanf(line, "%d", &index) == 1)
```

Dangerous Functions\Path 48:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=166
Status	New

The dangerous function, sscanf, was found in use at line 154 in sdlpal/text.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	204	204
Object	sscanf	sscanf

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
204.                                    sscanf(buffer + 15, "%d",
&sid) == 1)
```

Dangerous Functions\Path 49:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=167
Status	New

The dangerous function, sscanf, was found in use at line 154 in sdlpal/text.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	261	261
Object	sscanf	sscanf

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
261.                                    sscanf(buffer + 13, "%d", &eid) == 1
&& eid >= sid)
```

Dangerous Functions\Path 50:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=168
Status	New

The dangerous function, sscanf, was found in use at line 154 in sdlpal/text.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	360	360
Object	sscanf	sscanf

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
360.                                        sscanf(line, "%x",
&index);
```

Buffer Overflow boundcpy WrongSizeParam

Query Path:
CPP\Cx\CPP Buffer Overflow\Buffer Overflow boundcpy WrongSizeParam Version:1

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
OWASP Top 10 2017: A1-Injection

Description

Buffer Overflow boundcpy WrongSizeParam\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=34
Status	New

The size of the buffer used by PAL_ReadMessageFile in OBJECTDESC, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadMessageFile passes to OBJECTDESC, at line 154 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination

File	sdlpal/text.c	sdlpal/text.c
Line	377	377
Object	OBJECTDESC	OBJECTDESC

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
377.                        memset(lpObjectDesc,0,sizeof(OBJECTDESC));
```

Buffer Overflow boundcpy WrongSizeParam\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=35
Status	New

The size of the buffer used by op_make_decode_ready in channel_count, at line 1346 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_make_decode_ready passes to channel_count, at line 1346 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1375	1375
Object	channel_count	channel_count

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_make_decode_ready(OggOpusFile *_of){

```
....
1375.      memcpy(_of->od_mapping,head->mapping,sizeof(*head-
>mapping)*channel_count);
```

Buffer Overflow boundcpy WrongSizeParam\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=36
Status	New

The size of the buffer used by op_make_decode_ready in head, at line 1346 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_make_decode_ready passes to head, at line 1346 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination

File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1375	1375
Object	head	head

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_make_decode_ready(OggOpusFile *_of){

```
....
1375.      memcpy(_of->od_mapping,head->mapping,sizeof(*head-
>mapping)*channel_count);
```

Buffer Overflow boundcpy WrongSizeParam\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=37
Status	New

The size of the buffer used by op_open_seekable2 in start_op_count, at line 1417 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_open_seekable2 passes to start_op_count, at line 1417 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1443	1443
Object	start_op_count	start_op_count

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_open_seekable2(OggOpusFile *_of){

```
....
1443.    memcpy(op_start,_of->op,sizeof(*op_start)*start_op_count);
```

Buffer Overflow boundcpy WrongSizeParam\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=38
Status	New

The size of the buffer used by op_open_seekable2 in op_start, at line 1417 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_open_seekable2 passes to op_start, at line 1417 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination

File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1443	1443
Object	op_start	op_start

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_open_seekable2(OggOpusFile *_of){

```
....
1443.    memcpy(op_start,_of->op,sizeof(*op_start)*start_op_count);
```

Buffer Overflow boundcpy WrongSizeParam\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=39
Status	New

The size of the buffer used by op_open_seekable2 in start_op_count, at line 1417 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_open_seekable2 passes to start_op_count, at line 1417 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1455	1455
Object	start_op_count	start_op_count

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_open_seekable2(OggOpusFile *_of){

```
....
1455.    memcpy(_of->op,op_start,sizeof(*_of->op)*start_op_count);
```

Buffer Overflow boundcpy WrongSizeParam\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=40
Status	New

The size of the buffer used by op_open_seekable2 in _of, at line 1417 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_open_seekable2 passes to _of, at line 1417 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c

Line	1455	1455
Object	_of	_of

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_open_seekable2(OggOpusFile *_of){

```
....
1455.    memcpy(_of->op,op_start,sizeof(*_of->op)*start_op_count);
```

Buffer Overflow boundcpy WrongSizeParam\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=41
Status	New

The size of the buffer used by op_open1 in _initial_bytes, at line 1504 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_open1 passes to _initial_bytes, at line 1504 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1530	1530
Object	_initial_bytes	_initial_bytes

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_open1(OggOpusFile *_of,

```
....
1530.      memcpy(buffer,_initial_data,_initial_bytes*sizeof(*buffer));
```

Buffer Overflow boundcpy WrongSizeParam\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=42
Status	New

The size of the buffer used by op_open1 in buffer, at line 1504 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_open1 passes to buffer, at line 1504 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	1530	1530

Object	buffer	buffer

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_open1(OggOpusFile *_of,

```
....
1530.      memcpy(buffer,_initial_data,_initial_bytes*sizeof(*buffer));
```

Buffer Overflow boundcpy WrongSizeParam\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=43
Status	New

The size of the buffer used by op_stereo_filter in _nsamples, at line 3028 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_stereo_filter passes to _nsamples, at line 3028 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	3032	3032
Object	_nsamples	_nsamples

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_stereo_filter(OggOpusFile *_of,void *_dst,int _dst_sz,

```
....
3032.    if(_nchannels==2)memcpy(_dst,_src,_nsamples*2*sizeof(*_src));
```

Buffer Overflow boundcpy WrongSizeParam\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=44
Status	New

The size of the buffer used by op_stereo_filter in _src, at line 3028 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_stereo_filter passes to _src, at line 3028 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	3032	3032
Object	_src	_src

Code Snippet	
File Name	sdlpal/opusfile.c
Method	static int op_stereo_filter(OggOpusFile *_of,void *_dst,int _dst_sz,

```
....
3032.    if(_nchannels==2)memcpy(_dst,_src,_nsamples*2*sizeof(*_src));
```

Buffer Overflow boundcpy WrongSizeParam\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=45
Status	New

The size of the buffer used by ogg_stream_flush_i in os, at line 390 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ogg_stream_flush_i passes to os, at line 390 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	509	509
Object	os	os

Code Snippet	
File Name	sdlpal/framing.c
Method	static int ogg_stream_flush_i(ogg_stream_state *os,ogg_page *og, int force, int nfill){

```
....
509.    memmove(os->lacing_vals,os->lacing_vals+vals,os-
>lacing_fill*sizeof(*os->lacing_vals));
```

Buffer Overflow boundcpy WrongSizeParam\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=46
Status	New

The size of the buffer used by ogg_stream_flush_i in os, at line 390 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ogg_stream_flush_i passes to os, at line 390 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	509	509
Object	os	os

Code Snippet

File Name	sdlpal/framing.c
Method	static int ogg_stream_flush_i(ogg_stream_state *os,ogg_page *og, int force, int nfill){

```
....
509.    memmove(os->lacing_vals,os->lacing_vals+vals,os-
>lacing_fill*sizeof(*os->lacing_vals));
```

Buffer Overflow boundcpy WrongSizeParam\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=47
Status	New

The size of the buffer used by ogg_stream_flush_i in os, at line 390 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ogg_stream_flush_i passes to os, at line 390 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	510	510
Object	os	os

Code Snippet	
File Name	sdlpal/framing.c
Method	static int ogg_stream_flush_i(ogg_stream_state *os,ogg_page *og, int force, int nfill){

```
....
510.    memmove(os->granule_vals,os->granule_vals+vals,os-
>lacing_fill*sizeof(*os->granule_vals));
```

Buffer Overflow boundcpy WrongSizeParam\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=48
Status	New

The size of the buffer used by ogg_stream_flush_i in os, at line 390 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ogg_stream_flush_i passes to os, at line 390 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	510	510
Object	os	os

Code Snippet
File Name sdlpal/framing.c
Method static int ogg_stream_flush_i(ogg_stream_state *os,ogg_page *og, int force, int nfill){

```
....
510.    memmove(os->granule_vals,os->granule_vals+vals,os-
>lacing_fill*sizeof(*os->granule_vals));
```

Buffer Overflow boundcpy WrongSizeParam\Path 16:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=49
Status New

The size of the buffer used by ogg_stream_pagein in os, at line 808 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ogg_stream_pagein passes to os, at line 808 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	842	842
Object	os	os

Code Snippet
File Name sdlpal/framing.c
Method int ogg_stream_pagein(ogg_stream_state *os, ogg_page *og){

```
....
842.                    (os->lacing_fill-lr)*sizeof(*os->lacing_vals));
```

Buffer Overflow boundcpy WrongSizeParam\Path 17:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=50
Status New

The size of the buffer used by ogg_stream_pagein in os, at line 808 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ogg_stream_pagein passes to os, at line 808 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	844	844
Object	os	os

Code Snippet

File Name	sdlpal/framing.c
Method	int ogg_stream_pagein(ogg_stream_state *os, ogg_page *og){

```
....
844.                    (os->lacing_fill-lr)*sizeof(*os->granule_vals));
```

Buffer Overflow boundcpy WrongSizeParam\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=51
Status	New

The size of the buffer used by PAL_ReadMessageFile in int, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadMessageFile passes to int, at line 154 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	583	583
Object	int	int

Code Snippet	
File Name	sdlpal/text.c
Method	PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item-
>index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item-
>index] - oldCount));
```

Buffer Overflow boundcpy WrongSizeParam\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=52
Status	New

The size of the buffer used by op_read_native in nsamples, at line 2803 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_read_native passes to nsamples, at line 2803 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	2820	2820
Object	nsamples	nsamples

Code Snippet	
File Name	sdlpal/opusfile.c

Method	static int op_read_native(OggOpusFile *_of,

```
....
2820.           sizeof(*_pcm)*nchannels*nsamples);
```

Buffer Overflow boundcpy WrongSizeParam\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=53
Status	New

The size of the buffer used by op_read_native in nchannels, at line 2803 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_read_native passes to nchannels, at line 2803 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	2820	2820
Object	nchannels	nchannels

Code Snippet	
File Name	sdlpal/opusfile.c
Method	static int op_read_native(OggOpusFile *_of,

```
....
2820.           sizeof(*_pcm)*nchannels*nsamples);
```

Buffer Overflow boundcpy WrongSizeParam\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=54
Status	New

The size of the buffer used by op_read_native in _pcm, at line 2803 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_read_native passes to _pcm, at line 2803 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	2820	2820
Object	_pcm	_pcm

Code Snippet	
File Name	sdlpal/opusfile.c
Method	static int op_read_native(OggOpusFile *_of,

```
....
2820.           sizeof(*_pcm)*nchannels*nsamples);
```

Buffer Overflow boundcpy WrongSizeParam\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=55
Status	New

The size of the buffer used by op_read_native in nchannels, at line 2803 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_read_native passes to nchannels, at line 2803 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	2888	2888
Object	nchannels	nchannels

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_read_native(OggOpusFile *_of,

```
....
2888.                 sizeof(*_pcm)*trimmed_duration*nchannels);
```

Buffer Overflow boundcpy WrongSizeParam\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=56
Status	New

The size of the buffer used by op_read_native in trimmed_duration, at line 2803 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_read_native passes to trimmed_duration, at line 2803 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	2888	2888
Object	trimmed_duration	trimmed_duration

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_read_native(OggOpusFile *_of,

```
....
2888.                   sizeof(*_pcm)*trimmed_duration*nchannels);
```

Buffer Overflow boundcpy WrongSizeParam\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=57
Status	New

The size of the buffer used by op_read_native in _pcm, at line 2803 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_read_native passes to _pcm, at line 2803 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	2888	2888
Object	_pcm	_pcm

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_read_native(OggOpusFile *_of,

```
....
2888.                   sizeof(*_pcm)*trimmed_duration*nchannels);
```

Buffer Overflow boundcpy WrongSizeParam\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=58
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1853	1853
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1853.                 og[i].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=59
Status	New

The size of the buffer used by main in i, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to i, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1853	1853
Object	i	i

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1853.                 og[i].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=60
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1855	1855
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1855.
memcpy(ogg_sync_buffer(&oy,og[i].body_len),og[i].body,og[i].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=61
Status	New

The size of the buffer used by main in i, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to i, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1855	1855
Object	i	i

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1855.
memcpy(ogg_sync_buffer(&oy,og[i].body_len),og[i].body,og[i].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=62
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1902	1902
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1902.                   og[i].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=63
Status	New

The size of the buffer used by main in i, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to i, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1902	1902
Object	i	i

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1902.                   og[i].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=64
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1904	1904
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1904.
memcpy(ogg_sync_buffer(&oy,og[i].body_len),og[i].body,og[i].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=65
Status	New

The size of the buffer used by main in i, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to i, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1904	1904
Object	i	i

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1904.
memcpy(ogg_sync_buffer(&oy,og[i].body_len),og[i].body,og[i].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=66
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1997	1997
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
1997.                 og[1].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=67
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2001	2001
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2001.                 og[1].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 35:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=68
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2014	2014
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2014.                 og[1].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 36:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=69
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2029	2029
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2029.                 og[1].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 37:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=70
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2033	2033
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2033.                 og[1].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 38:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=71
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2037	2037
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2037.                 og[1].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 39:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=72
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2051	2051
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2051.                 og[2].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 40:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=73
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2065	2065
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2065.                 og[1].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 41:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=74
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2069	2069
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2069.               og[1].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 42:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=75
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2073	2073
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2073.               og[2].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 43:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=76
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2077	2077
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2077.                og[2].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 44:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=77
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2087	2087
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2087.                og[3].header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 45:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=78
Status	New

The size of the buffer used by main in og, at line 1674 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to og, at line 1674 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	2091	2091
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method int main(void){

```
....
2091.                og[3].body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 46:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=79
Status	New

The size of the buffer used by ogg_stream_iovecin in iov, at line 317 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ogg_stream_iovecin passes to iov, at line 317 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	355	355
Object	iov	iov

Code Snippet
File Name sdlpal/framing.c
Method int ogg_stream_iovecin(ogg_stream_state *os, ogg_iovec_t *iov, int count,

```
....
355.      memcpy(os->body_data+os->body_fill, iov[i].iov_base,
iov[i].iov_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 47:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=80
Status	New

The size of the buffer used by ogg_stream_iovecin in i, at line 317 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ogg_stream_iovecin passes to i, at line 317 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	355	355
Object	i	i

Code Snippet
File Name sdlpal/framing.c
Method int ogg_stream_iovecin(ogg_stream_state *os, ogg_iovec_t *iov, int count,

```
....
355.      memcpy(os->body_data+os->body_fill, iov[i].iov_base,
iov[i].iov_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 48:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=81
Status	New

The size of the buffer used by copy_page in og, at line 1142 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that copy_page passes to og, at line 1142 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1144	1144
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method void copy_page(ogg_page *og){

```
....
1144.    memcpy(temp,og->header,og->header_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 49:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=82
Status	New

The size of the buffer used by copy_page in og, at line 1142 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that copy_page passes to og, at line 1142 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	1148	1148
Object	og	og

Code Snippet
File Name sdlpal/framing.c
Method void copy_page(ogg_page *og){

```
....
1148.    memcpy(temp,og->body,og->body_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 50:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=83
Status	New

The size of the buffer used by ogg_stream_iovecin in os, at line 317 of sdlpal/framing.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ogg_stream_iovecin passes to os, at line 317 of sdlpal/framing.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	341	341
Object	os	os

Code Snippet
File Name sdlpal/framing.c
Method int ogg_stream_iovecin(ogg_stream_state *os, ogg_iovec_t *iov, int count,

```
....
341.                os->body_fill);
```

Integer Overflow

Query Path:
CPP\Cx\CPP Integer Overflow\Integer Overflow Version:0

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
FISMA 2014: System And Information Integrity
NIST SP 800-53: SI-10 Information Input Validation (P1)

Description

Integer Overflow\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=107
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 2255 of sdlpal/text.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c

Line	2467	2467
Object	AssignExpr	AssignExpr

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2467.                                   precision = buffer_end - buffer;
```

Integer Overflow\Path 2:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=108
Status New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 2255 of sdlpal/text.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2359	2359
Object	AssignExpr	AssignExpr

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2359.                                   width = width * 10 + (*format -
L'0');
```

Integer Overflow\Path 3:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=109
Status New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 2255 of sdlpal/text.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2391	2391
Object	AssignExpr	AssignExpr

Code Snippet	
File Name	sdlpal/text.c
Method	PAL_swprintf(

```
....
2391.                                   precision = precision * 10 +
(*format - L'0');
```

Integer Overflow\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=110
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 2255 of sdlpal/text.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2459	2459
Object	AssignExpr	AssignExpr

Code Snippet	
File Name	sdlpal/text.c
Method	PAL_swprintf(

```
....
2459.                                   precision = len;
```

Integer Overflow\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=111
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 1956 of sdlpal/text.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2064	2064
Object	AssignExpr	AssignExpr

Code Snippet	
File Name	sdlpal/text.c
Method	PAL_MultiByteToWideCharCP(

```
....
2064.               i = mbslength;
```

Integer Overflow\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=112
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 1956 of sdlpal/text.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2065	2065
Object	AssignExpr	AssignExpr

Code Snippet
File Name sdlpal/text.c
Method PAL_MultiByteToWideCharCP(

```
....
2065.               wlen = mbslength/2;
```

Integer Overflow\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=113
Status	New

A variable of a larger data type, buf_len, is being assigned to a smaller data type, in 2255 of sdlpal/text.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2543	2543
Object	buf_len	buf_len

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2543.               int buf_len = buffer_end - buffer;
```

Integer Overflow\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=114
Status	New

A variable of a larger data type, fmt_len, is being assigned to a smaller data type, in 2255 of sdlpal/text.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2542	2542
Object	fmt_len	fmt_len

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2542.             int fmt_len = format - fmt_start;
```

Stored Buffer Overflow boundcpy

Query Path:
CPP\Cx\CPP Stored Vulnerabilities\Stored Buffer Overflow boundcpy Version:1

Categories

NIST SP 800-53: SI-10 Information Input Validation (P1)
OWASP Top 10 2017: A1-Injection

Description

Stored Buffer Overflow boundcpy\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=201
Status	New

The size of the buffer used by PAL_ReadMessageFile in oldCount, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	583
Object	temp	oldCount

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item-
>index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item-
>index] - oldCount));
```

Stored Buffer Overflow boundcpy\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=202
Status	New

The size of the buffer used by PAL_ReadMessageFile in oldCount, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	583
Object	BinaryExpr	oldCount

Code Snippet

File Name sdlpal/text.c

Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item-
>index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item-
>index] - oldCount));
```

Stored Buffer Overflow boundcpy\Path 3:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=203
Status	New

The size of the buffer used by PAL_ReadMessageFile in BinaryExpr, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	583
Object	temp	BinaryExpr

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item->index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item->index] - oldCount));
```

Stored Buffer Overflow boundcpy\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=204
Status	New

The size of the buffer used by PAL_ReadMessageFile in BinaryExpr, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	583
Object	BinaryExpr	BinaryExpr

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item-
>index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item-
>index] - oldCount));
```

Stored Buffer Overflow boundcpy\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=205
Status	New

The size of the buffer used by PAL_ReadMessageFile in BinaryExpr, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	583
Object	temp	BinaryExpr

Code Snippet

File Name sdlpal/text.c

Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item-
>index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item-
>index] - oldCount));
```

Stored Buffer Overflow boundcpy\Path 6:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=206
Status	New

The size of the buffer used by PAL_ReadMessageFile in BinaryExpr, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	583
Object	BinaryExpr	BinaryExpr

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item->index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item->index] - oldCount));
```

Stored Buffer Overflow boundcpy\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=207
Status	New

The size of the buffer used by PAL_ReadMessageFile in sizeof, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to temp, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	583
Object	temp	sizeof

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item-
>index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item-
>index] - oldCount));
```

Stored Buffer Overflow boundcpy\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=208
Status	New

The size of the buffer used by PAL_ReadMessageFile in sizeof, at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadOneLine passes to BinaryExpr, at line 111 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	583
Object	BinaryExpr	sizeof

Code Snippet

File Name sdlpal/text.c

Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

File Name sdlpal/text.c

Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item-
>index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item-
>index] - oldCount));
```

Wrong Size t Allocation

Query Path:

CPP\Cx\CPP Integer Overflow\Wrong Size t Allocation Version:0

Description

Wrong Size t Allocation\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=88
Status	New

The function cur_fmt in sdlpal/text.c at line 2255 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2488	2488
Object	cur_fmt	cur_fmt

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2488.                                cur_fmt = realloc(cur_fmt, ((fmt_len
= format - fmt_start + 1) + 1) * sizeof(WCHAR));
```

Wrong Size t Allocation\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=89
Status	New

The function fmt_len in sdlpal/text.c at line 2255 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2488	2488
Object	fmt_len	fmt_len

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2488.                                cur_fmt = realloc(cur_fmt, ((fmt_len
= format - fmt_start + 1) + 1) * sizeof(WCHAR));
```

Wrong Size t Allocation\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=90
Status	New

The function format in sdlpal/text.c at line 2255 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2488	2488
Object	format	format

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2488.                                cur_fmt = realloc(cur_fmt, ((fmt_len
= format - fmt_start + 1) + 1) * sizeof(WCHAR));
```

Wrong Size t Allocation\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=91
Status	New

The function fmt_start in sdlpal/text.c at line 2255 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2488	2488
Object	fmt_start	fmt_start

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2488.                                cur_fmt = realloc(cur_fmt, ((fmt_len
= format - fmt_start + 1) + 1) * sizeof(WCHAR));
```

Use of Zero Initialized Pointer

Query Path:

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=1050054&projectid=50044&pathid=198
Status	New

The variable declared in fmt_start at sdlpal/text.c in line 2255 is not initialized when it is used by fmt_start at sdlpal/text.c in line 2255.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2291	2323
Object	fmt_start	fmt_start

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2291.       LPCWSTR fmt_start = NULL;
....
2323.                           fmt_start = format++;
```

Use of Zero Initialized Pointer\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=199
Status	New

The variable declared in fmt_start at sdlpal/text.c in line 2255 is not initialized when it is used by fmt_start at sdlpal/text.c in line 2255.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2291	2542
Object	fmt_start	fmt_start

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2291.       LPCWSTR fmt_start = NULL;
....
2542.             int fmt_len = format - fmt_start;
```

Use of Zero Initialized Pointer\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=200
Status	New

The variable declared in cur_fmt at sdlpal/text.c in line 2255 is not initialized when it is used by cur_fmt at sdlpal/text.c in line 2255.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	2292	2488
Object	cur_fmt	cur_fmt

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2292.       LPWSTR cur_fmt = NULL;
....
2488.                                   cur_fmt = realloc(cur_fmt, ((fmt_len
= format - fmt_start + 1) + 1) * sizeof(WCHAR));
```

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=1050054&projectid=50044&pathid=196
Status	New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c

Line	772	772
Object	tmp	tmp

Code Snippet
File Name sdlpal/text.c
Method PAL_InitText(

```
....
772.            tmp = (LPWSTR)malloc(wlen * sizeof(WCHAR));
```

Memory Leak\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=197
Status	New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	850	850
Object	tmp	tmp

Code Snippet
File Name sdlpal/text.c
Method PAL_InitText(

```
....
850.            tmp = (LPWSTR)malloc(wlen * sizeof(WCHAR));
```

Divide By Zero

Query Path:
CPP\Cx\CPP Medium Threat\Divide By Zero Version:1

Description

Divide By Zero\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=33
Status	New

The application performs an illegal operation in PAL_DetectCodePageForString, in sdlpal/text.c. In line 1890, the program attempts to divide by text_len, which might be evaluate to 0 (zero) at time of division. This value could be a hard-coded zero value, or received from external, untrusted input text_len in PAL_DetectCodePageForString of sdlpal/text.c, at line 1890.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c

Line	1947	1947
Object	text_len	text_len

Code Snippet
File Name sdlpal/text.c
Method PAL_DetectCodePageForString(

```
....
1947.                    *probability = (text_len / 2 - min_invalids) *
200 / text_len;
```

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=1050054&projectid=50044&pathid=209
Status	New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	117
Object	fgets	fgets

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=210
Status	New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	131
Object	fgets	fgets

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

Improper Resource Access Authorization\Path 3:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=211
Status New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	117	117
Object	temp	temp

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadOneLine(

```
....
117.        if (fgets(temp, limit, fp))
```

Improper Resource Access Authorization\Path 4:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=212
Status New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	131	131
Object	BinaryExpr	BinaryExpr

Code Snippet

File Name	sdlpal/text.c
Method	PAL_ReadOneLine(

```
....
131.                            if (fgets(tmp + n, limit + 1, fp))
```

Improper Resource Access Authorization\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=213
Status	New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	744	744
Object	temp	temp

Code Snippet

File Name	sdlpal/text.c
Method	PAL_InitText(

```
....
744.            if (fread(temp, 1, i, fpWord) < (size_t)i)
```

Improper Resource Access Authorization\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=214
Status	New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	826	826
Object	temp	temp

Code Snippet

File Name	sdlpal/text.c
Method	PAL_InitText(

```
....
826.            if (fread(temp, 1, i, fpMsg) < (size_t)i)
```

Improper Resource Access Authorization\Path 7:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=215
Status	New

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	151	151
Object	buffer	buffer

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_get_data(OggOpusFile *_of,int _nbytes){

```
....
151.    nbytes=(int)(*_of->callbacks.read)(_of->stream,buffer,_nbytes);
```

Improper Resource Access Authorization\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=216
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	871	871
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
871.    fprintf(stderr,"\nSmall preclipped packing (LSb): ");
```

Improper Resource Access Authorization\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=217
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c

Line	873	873
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
873.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 10:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=218
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	875	875
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
875.    fprintf(stderr,"\nNull bit call (LSb): ");
```

Improper Resource Access Authorization\Path 11:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=219
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	877	877
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
877.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=220
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	879	879
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
879.    fprintf(stderr,"\nLarge preclipped packing (LSb): ");
```

Improper Resource Access Authorization\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=221
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	881	881
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
881.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 14:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=222
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	883	883
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
883.    fprintf(stderr,"\n32 bit preclipped packing (LSb): ");
```

Improper Resource Access Authorization\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=223
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	893	893
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
893.        fprintf(stderr,"%ld != %ld
(%lx!=%lx):",oggpack_look(&r,32),large[i],
```

Improper Resource Access Authorization\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=224
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c

Line	900	900
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
900.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 17:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=225
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	902	902
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
902.    fprintf(stderr,"\nSmall unclipped packing (LSb): ");
```

Improper Resource Access Authorization\Path 18:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=226
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	904	904
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
904.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 19:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=227
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	906	906
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
906.    fprintf(stderr,"\nLarge unclipped packing (LSb): ");
```

Improper Resource Access Authorization\Path 20:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=228
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	908	908
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
908.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 21:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=229
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	910	910
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
910.    fprintf(stderr,"\nSingle bit unclipped packing (LSb): ");
```

Improper Resource Access Authorization\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=230
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	912	912
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
912.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=231
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	914	914

Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
914.    fprintf(stderr,"\nTesting read past end (LSb): ");
```

Improper Resource Access Authorization\Path 24:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=232
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	918	918
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
918.        fprintf(stderr,"failed; got -1 prematurely.\n");
```

Improper Resource Access Authorization\Path 25:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=233
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	924	924
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
924.        fprintf(stderr,"failed; read past end without -1.\n");
```

Improper Resource Access Authorization\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=234
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	929	929
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
929.        fprintf(stderr,"failed 2; got -1 prematurely.\n");
```

Improper Resource Access Authorization\Path 27:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=235
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	935	935
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
935.      fprintf(stderr,"failed 3; got -1 prematurely.\n");
```

Improper Resource Access Authorization\Path 28:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=236
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	940	940
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
940.      fprintf(stderr,"failed; read past end without -1.\n");
```

Improper Resource Access Authorization\Path 29:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=237
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	945	945
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
945.      fprintf(stderr,"failed; read past end without -1.\n");
```

Improper Resource Access Authorization\Path 30:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=238
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	949	949

Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
949.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 31:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=239
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	953	953
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
953.    fprintf(stderr,"\nTesting aligned writecopies (LSb): ");
```

Improper Resource Access Authorization\Path 32:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=240
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	960	960
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
960.    fprintf(stderr,"ok.      ");
```

Improper Resource Access Authorization\Path 33:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=241
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	962	962
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
962.    fprintf(stderr,"\nTesting unaligned writecopies (LSb): ");
```

Improper Resource Access Authorization\Path 34:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=242
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	972	972
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
972.    fprintf(stderr,"ok.      \n");
```

Improper Resource Access Authorization\Path 35:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=243
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	981	981
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
981.    fprintf(stderr,"\nSmall preclipped packing (MSb): ");
```

Improper Resource Access Authorization\Path 36:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=244
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	983	983
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
983.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 37:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=245
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	985	985

Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
985.    fprintf(stderr,"\nNull bit call (MSb): ");
```

Improper Resource Access Authorization\Path 38:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=246
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	987	987
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
987.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 39:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=247
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	989	989
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
989.    fprintf(stderr,"\nLarge preclipped packing (MSb): ");
```

Improper Resource Access Authorization\Path 40:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=248
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	991	991
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
991.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 41:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=249
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	993	993
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
993.    fprintf(stderr,"\n32 bit preclipped packing (MSb): ");
```

Improper Resource Access Authorization\Path 42:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=250
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	1003	1003
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
1003.         fprintf(stderr,"%ld != %ld
(%lx!=%lx):",oggpackB_look(&r,32),large[i],
```

Improper Resource Access Authorization\Path 43:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=251
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	1010	1010
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
1010.     fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 44:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=252
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c

Line	1012	1012
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
1012.    fprintf(stderr,"\nSmall unclipped packing (MSb): ");
```

Improper Resource Access Authorization\Path 45:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=253
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	1014	1014
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
1014.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 46:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=254
Status New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	1016	1016
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
1016.    fprintf(stderr,"\nLarge unclipped packing (MSb): ");
```

Improper Resource Access Authorization\Path 47:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=255
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	1018	1018
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
1018.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 48:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=256
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	1020	1020
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
1020.    fprintf(stderr,"\nSingle bit unclipped packing (MSb): ");
```

Improper Resource Access Authorization\Path 49:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=257
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	1022	1022
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
1022.    fprintf(stderr,"ok.");
```

Improper Resource Access Authorization\Path 50:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=258
Status	New

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	1024	1024
Object	fprintf	fprintf

Code Snippet
File Name sdlpal/bitwise.c
Method int main(void){

```
....
1024.    fprintf(stderr,"\nTesting read past end (MSb): ");
```

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=1050054&projectid=50044&pathid=92
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 628.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	629	657
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
629.    if(ogg_sync_check(oy)) return NULL;
....
657.    return((char *)oy->data+oy->fill);
```

NULL Pointer Dereference\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=93
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 628.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	650	657
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
650.        return NULL;
....
657.    return((char *)oy->data+oy->fill);
```

NULL Pointer Dereference\Path 3:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=94
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 628.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	629	632
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
629.    if(ogg_sync_check(oy)) return NULL;
....
632.    if(oy->returned){
```

NULL Pointer Dereference\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=95
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 628.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	650	632
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
650.        return NULL;
....
632.    if(oy->returned){
```

NULL Pointer Dereference\Path 5:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=96
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 628.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	629	657
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
629.    if(ogg_sync_check(oy)) return NULL;
....
657.    return((char *)oy->data+oy->fill);
```

NULL Pointer Dereference\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=97
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 628.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	650	657
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
650.        return NULL;
....
657.    return((char *)oy->data+oy->fill);
```

NULL Pointer Dereference\Path 7:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=98
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 628.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	629	633
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
629.    if(ogg_sync_check(oy)) return NULL;
....
633.      oy->fill-=oy->returned;
```

NULL Pointer Dereference\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=99
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 628.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	650	633
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
650.        return NULL;
....
633.      oy->fill-=oy->returned;
```

NULL Pointer Dereference\Path 9:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=100
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 660.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	629	663
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
629.    if(ogg_sync_check(oy)) return NULL;
```

File Name sdlpal/framing.c
Method int ogg_sync_wrote(ogg_sync_state *oy, long bytes){

```
....
663.    oy->fill+=bytes;
```

NULL Pointer Dereference\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=101
Status	New

The variable declared in null at sdlpal/framing.c in line 628 is not initialized when it is used by oy at sdlpal/framing.c in line 660.

	Source	Destination
File	sdlpal/framing.c	sdlpal/framing.c
Line	650	663
Object	null	oy

Code Snippet
File Name sdlpal/framing.c
Method char *ogg_sync_buffer(ogg_sync_state *oy, long size){

```
....
650.        return NULL;
```

File Name	sdlpal/framing.c
Method	int ogg_sync_wrote(ogg_sync_state *oy, long bytes){

```
....
663.    oy->fill+=bytes;
```

NULL Pointer Dereference\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=102
Status	New

The variable declared in 0 at sdlpal/bitwise.c in line 180 is not initialized when it is used by b at sdlpal/bitwise.c in line 180.

	Source	Destination
File	sdlpal/bitwise.c	sdlpal/bitwise.c
Line	216	216
Object	0	b

Code Snippet

File Name	sdlpal/bitwise.c
Method	static void oggpack_writecopy_helper(oggpack_buffer *b,

```
....
216.      *b->ptr=0;
```

NULL Pointer Dereference\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=103
Status	New

The variable declared in 0 at sdlpal/opusfile.c in line 630 is not initialized when it is used by _of at sdlpal/opusfile.c in line 829.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	662	952
Object	0	_of

Code Snippet

File Name	sdlpal/opusfile.c

Method static int op_granpos_add(ogg_int64_t *_dst_gp,ogg_int64_t _src_gp,

```
....
662.    return 0;
```

File Name sdlpal/opusfile.c

Method static int op_find_initial_pcm_offset(OggOpusFile *_of,

```
....
952.      prev_packet_gp=_of->op[pi].granulepos;
```

NULL Pointer Dereference\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=104
Status	New

The variable declared in 0 at sdlpal/opusfile.c in line 630 is not initialized when it is used by _of at sdlpal/opusfile.c in line 829.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	662	950
Object	0	_of

Code Snippet

File Name sdlpal/opusfile.c

Method static int op_granpos_add(ogg_int64_t *_dst_gp,ogg_int64_t _src_gp,

```
....
662.    return 0;
```

File Name sdlpal/opusfile.c

Method static int op_find_initial_pcm_offset(OggOpusFile *_of,

```
....
950.      OP_ALWAYS_TRUE(!op_granpos_add(&_of->op[pi].granulepos,
```

Use of Sizeof On a Pointer Type

Query Path:

CPP\Cx\CPP Low Visibility\Use of Sizeof On a Pointer Type Version:1

Description

Use of Sizeof On a Pointer Type\Path 1:

Severity	Low
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=28
Status	New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	560	560
Object	sizeof	sizeof

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
560.              g_TextLib.lpIndexBuf = (int ***)UTIL_calloc(idx_cnt,
sizeof(int **));
```

Use of Sizeof On a Pointer Type\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=29
Status	New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	562	562
Object	sizeof	sizeof

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
562.              g_TextLib.indexMaxCounter = (int
*)UTIL_calloc(idx_cnt, sizeof(int *));
```

Use of Sizeof On a Pointer Type\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=30
Status	New

	Source	Destination

File	sdlpal/text.c	sdlpal/text.c
Line	579	579
Object	sizeof	sizeof

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
579.                                    g_TextLib.lpIndexBuf[item->index] =
(int **)realloc(g_TextLib.lpIndexBuf[item->index], sizeof(int *) *
(item->indexEnd - item->index + 1));
```

Use of Sizeof On a Pointer Type\Path 4:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=31
Status New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	583	583
Object	sizeof	sizeof

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
583.                                    memset(&g_TextLib.lpIndexBuf[item-
>index][oldCount], 0, sizeof(int**)*(g_TextLib.indexMaxCounter[item-
>index] - oldCount));
```

Use of Sizeof On a Pointer Type\Path 5:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=32
Status New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	587	587
Object	sizeof	sizeof

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
587.                            g_TextLib.lpIndexBuf[item->index] = (int
**)UTIL_calloc((item->indexEnd - item->index + 1), sizeof(int *));
```

Unchecked Array Index

Query Path:
CPP\Cx\CPP Low Visibility\Unchecked Array Index Version:1

Categories

NIST SP 800-53: SI-10 Information Input Validation (P1)

Description

Unchecked Array Index\Path 1:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=116
Status New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	614	614
Object	count	count

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
614.                    g_TextLib.lpIndexBuf[item->index][item->indexEnd
- item->index][item->count] = -1;
```

Unchecked Array Index\Path 2:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=117
Status New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	633	633
Object	index	index

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
633.                    g_TextLib.lpWordBuf[witem->index] = witem-
>value;
```

Unchecked Array Index\Path 3:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=118
Status New

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	864	864
Object	l	l

Code Snippet
File Name sdlpal/text.c
Method PAL_InitText(

```
....
864.                g_TextLib.lpMsgBuf[i][l] = 0;
```

Unchecked Return Value

Query Path:
CPP\Cx\CPP Low Visibility\Unchecked Return Value Version:1

Categories

NIST SP 800-53: SI-11 Error Handling (P2)

Description

Unchecked Return Value\Path 1:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=27
Status New

The PAL_swprintf method calls the cur_fmt function, at line 2255 of sdlpal/text.c. However, the code does not check the return value from this function, and thus would not detect runtime errors or other unexpected states.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c

Line	2488	2488
Object	cur_fmt	cur_fmt

Code Snippet
File Name sdlpal/text.c
Method PAL_swprintf(

```
....
2488.                                cur_fmt = realloc(cur_fmt, ((fmt_len
= format - fmt_start + 1) + 1) * sizeof(WCHAR));
```

Heuristic 2nd Order Buffer Overflow read

Query Path:
CPP\Cx\CPP Heuristic\Heuristic 2nd Order Buffer Overflow read Version:0

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
NIST SP 800-53: SI-10 Information Input Validation (P1)
OWASP Top 10 2017: A1-Injection

Description
Heuristic 2nd Order Buffer Overflow read\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=105
Status	New

The size of the buffer used by op_get_data in _nbytes, at line 146 of sdlpal/opusfile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that op_get_data passes to buffer, at line 146 of sdlpal/opusfile.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	151	151
Object	buffer	_nbytes

Code Snippet
File Name sdlpal/opusfile.c
Method static int op_get_data(OggOpusFile *_of,int _nbytes){

```
....
151.    nbytes=(int)(*_of->callbacks.read)(_of->stream,buffer,_nbytes);
```

Potential Precision Problem

Query Path:
CPP\Cx\CPP Buffer Overflow\Potential Precision Problem Version:0

Categories

NIST SP 800-53: SI-10 Information Input Validation (P1)

OWASP Top 10 2017: A1-Injection

Description

Potential Precision Problem\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=106
Status	New

The size of the buffer used by PAL_ReadMessageFile in "%s", at line 154 of sdlpal/text.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PAL_ReadMessageFile passes to "%s", at line 154 of sdlpal/text.c, to overwrite the target buffer.

	Source	Destination
File	sdlpal/text.c	sdlpal/text.c
Line	428	428
Object	"%s"	"%s"

Code Snippet
File Name sdlpal/text.c
Method PAL_ReadMessageFile(

```
....
428.                                                    if (sscanf(line, "%s",
index) == 1)
```

Arithmenic Operation On Boolean

Query Path:
CPP\Cx\CPP Low Visibility\Arithmenic Operation On Boolean Version:1

Categories

FISMA 2014: Audit And Accountability
NIST SP 800-53: SC-5 Denial of Service Protection (P1)

Description

Arithmenic Operation On Boolean\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050054&projectid=50044&pathid=115
Status	New

	Source	Destination
File	sdlpal/opusfile.c	sdlpal/opusfile.c
Line	734	734
Object	BinaryExpr	BinaryExpr

Code Snippet

File Name	sdlpal/opusfile.c
Method	static int op_granpos_cmp(ogg_int64_t _gp_a,ogg_int64_t _gp_b){

```
....
734.    return (_gp_a>_gp_b)-(_gp_b>_gp_a);
```

Buffer Overflow IndexFromInput

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Buffer Overflow OutOfBound

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Divide By Zero

Risk

What might happen

When a program divides a number by zero, an exception will be raised. If this exception is not handled by the application, unexpected results may occur, including crashing the application. This can be considered a DoS (Denial of Service) attack, if an external user has control of the value of the denominator or can cause this error to occur.

Cause

How does it happen

The program receives an unexpected value, and uses it for division without filtering, validation, or verifying that the value is not zero. The application does not explicitly handle this error or prevent division by zero from occuring.

General Recommendations

How to avoid it

- Before dividing by an unknown value, validate the number and explicitly ensure it does not evaluate to zero.
- Validate all untrusted input from all sources, in particular verifying that it is not zero before dividing with it.
- Verify output of methods, calculations, dictionary lookups, and so on, and ensure it is not zero before dividing with the result.
- Ensure divide-by-zero errors are caught and handled appropriately.

Source Code Examples

Java

Divide by Zero

```
public float getAverage(HttpServletRequest req) {
      int total = Integer.parseInt(req.getParameter("total"));
      int count = Integer.parseInt(req.getParameter("count"));

      return total / count;
}
```

Checked Division

```
public float getAverage(HttpServletRequest req) {
      int total = Integer.parseInt(req.getParameter("total"));
      int count = Integer.parseInt(req.getParameter("count"));
```

```
	if (count > 0)
		return total / count;
	else
		return 0;
}
```

Buffer Overflow boundcpy WrongSizeParam

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Wrong Size t Allocation

Risk

What might happen

Incorrect allocation of memory may result in unexpected behavior by either overwriting sections of memory with unexpected values. Under certain conditions where both an incorrect allocation of memory and the values being written can be controlled by an attacker, such an issue may result in execution of malicious code.

Cause

How does it happen

Some memory allocation functions require a size value to be provided as a parameter. The allocated size should be derived from the provided value, by providing the length value of the intended source, multiplied by the size of that length. Failure to perform the correct arithmetic to obtain the exact size of the value will likely result in the source overflowing its destination.

General Recommendations

How to avoid it

- Always perform the correct arithmetic to determine size.
- Specifically for memory allocation, calculate the allocation size from the allocation source:
    - Derive the size value from the length of intended source to determine the amount of units to be processed.
    - Always programmatically consider the size of the each unit and their conversion to memory units - for example, by using sizeof() on the unit's type.
    - Memory allocation should be a multiplication of the amount of units being written, times the size of each unit.

Source Code Examples

CPP

Allocating and Assigning Memory without Sizeof Arithmetic

```
int *ptr;
ptr = (int*)malloc(5);
for (int i = 0; i < 5; i++)
{
      ptr[i] = i * 2 + 1;
}
```

Allocating and Assigning Memory with Sizeof Arithmetic

```
int *ptr;
ptr = (int*)malloc(5 * sizeof(int));
```

```
for (int i = 0; i < 5; i++)
{
      ptr[i] = i * 2 + 1;
}
```

Incorrect Arithmetic of Multi-Byte String Allocation

```
wchar_t * dest;
dest = (wchar_t *)malloc(wcslen(source) + 1); // Would not crash for a short "source"
wcscpy((wchar_t *)dest, source);
wprintf(L"Dest: %s\r\n", dest);
```

Correct Arithmetic of Multi-Byte String Allocation

```
wchar_t * dest;
dest = (wchar_t *)malloc((wcslen(source) + 1) * sizeof(wchar_t));
wcscpy((wchar_t *)dest, source);
wprintf(L"Dest: %s\r\n", dest);
```

Integer Overflow

Risk

What might happen

Assigning large data types into smaller data types, without proper checks and explicit casting, will lead to undefined behavior and unintentional effects, such as data corruption (e.g. value wraparound, wherein maximum values become minimum values); system crashes; infinite loops; logic errors, such as bypassing of security mechanisms; or even buffer overflows leading to arbitrary code execution.

Cause

How does it happen

This flaw can occur when implicitly casting numerical data types of a larger size, into a variable with a data type of a smaller size. This forces the program to discard some bits of information from the number. Depending on how the numerical data types are stored in memory, this is often the bits with the highest value, causing substantial corruption of the stored number. Alternatively, the sign bit of a signed integer could be lost, completely reversing the intention of the number.

General Recommendations

How to avoid it

- Avoid casting larger data types to smaller types.
- Prefer promoting the target variable to a large enough data type.
- If downcasting is necessary, always check that values are valid and in range of the target type, before casting

---

Source Code Examples

CPP

Unsafe Downsize Casting

```
int unsafe_addition(short op1, int op2) {

    // op2 gets forced from int into a short
    short total = op1 + op2;

    return total;
}
```

Safer Use of Proper Data Types

```
int safe_addition(short op1, int op2) {

    // total variable is of type int, the largest type that is needed
    int total = 0;

    // check if total will overflow available integer size
    if (INT_MAX - abs(op2) > op1)
```

```
    {
        total = op1 + op2;
    }
    else
    {
        // instead of overflow, saturate (but this is not always a good thing)
        total = INT_MAX
    }

    return total;
}
```

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 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());
```

Stored Buffer Overflow boundcpy

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

CPP

Overflowing Buffers

```
const int BUFFER_SIZE = 10;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
{
    strcpy(buffer, inputString);
}
```

Checked Buffers

```
const int BUFFER_SIZE = 10;
const int MAX_INPUT_SIZE = 256;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
```

```
{
    if (strnlen(inputString, MAX_INPUT_SIZE) < sizeof(buffer))
    {
        strncpy(buffer, inputString, sizeof(buffer));
    }
}
```

Unchecked Return Value

Risk

What might happen

A program that does not check function return values could cause the application to enter an undefined state. This could lead to unexpected behavior and unintended consequences, including inconsistent data, system crashes or other error-based exploits.

Cause

How does it happen

The application calls a system function, but does not receive or check the result of this funciton. These functions often return error codes in the result, or share other status codes with it's caller. The application simply ignores this result value, losing this vital information.

General Recommendations

How to avoid it

- Always check the result of any called function that returns a value, and verify the result is an expected value.

- Ensure the calling function responds to all possible return values.

- Expect runtime errors and handle them gracefully. Explicitly define a mechanism for handling unexpected errors.

Source Code Examples

CPP

Unchecked Memory Allocation

```
buff = (char*) malloc(size);
strncpy(buff, source, size);
```

Safer Memory Allocation

```
buff = (char*) malloc(size+1);
if (buff==NULL) exit(1);

strncpy(buff, source, size);
buff[size] = '\0';
```

Use of sizeof() on a Pointer Type

Weakness ID: 467 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The code calls sizeof() on a malloced pointer type, which always returns the wordsize/8. This can produce an unexpected result if the programmer intended to determine how much memory has been allocated.

Time of Introduction

- Implementation

Applicable Platforms

Languages

C

C++

Common Consequences

Scope	Effect
Integrity	This error can often cause one to allocate a buffer that is much smaller than what is needed, leading to resultant weaknesses such as buffer overflows.

Likelihood of Exploit

High

Demonstrative Examples

Example 1

Care should be taken to ensure sizeof returns the size of the data structure itself, and not the size of the pointer to the data structure.

In this example, sizeof(foo) returns the size of the pointer.

(Bad Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(foo));
```

In this example, sizeof(*foo) returns the size of the data structure and not the size of the pointer.

(Good Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(*foo));
```

Example 2

This example defines a fixed username and password. The AuthenticateUser() function is intended to accept a username and a password from an untrusted user, and check to ensure that it matches the username and password. If the username and password match, AuthenticateUser() is intended to indicate that authentication succeeded.

(Bad Code)

```
/* Ignore CWE-259 (hard-coded password) and CWE-309 (use of password system for authentication) for this example. */

char *username = "admin";
char *pass = "password";

int AuthenticateUser(char *inUser, char *inPass) {
```

```
printf("Sizeof username = %d\n", sizeof(username));
printf("Sizeof pass = %d\n", sizeof(pass));

if (strncmp(username, inUser, sizeof(username))) {
printf("Auth failure of username using sizeof\n");
return(AUTH_FAIL);
}
/* Because of CWE-467, the sizeof returns 4 on many platforms and architectures. */
if (! strncmp(pass, inPass, sizeof(pass))) {
printf("Auth success of password using sizeof\n");
return(AUTH_SUCCESS);
}
else {
printf("Auth fail of password using sizeof\n");
return(AUTH_FAIL);
}
}

int main (int argc, char **argv)
{
int authResult;

if (argc < 3) {
ExitError("Usage: Provide a username and password");
}
authResult = AuthenticateUser(argv[1], argv[2]);
if (authResult != AUTH_SUCCESS) {
ExitError("Authentication failed");
}
else {
DoAuthenticatedTask(argv[1]);
}
}
```

In AuthenticateUser(), because sizeof() is applied to a parameter with an array type, the sizeof() call might return 4 on many modern architectures. As a result, the strncmp() call only checks the first four characters of the input password, resulting in a partial comparison (CWE-187), leading to improper authentication (CWE-287).

Because of the partial comparison, any of these passwords would still cause authentication to succeed for the "admin" user:

(Attack)

```
pass5
passABCDEFGH
passWORD
```

Because only 4 characters are checked, this significantly reduces the search space for an attacker, making brute force attacks more feasible.

The same problem also applies to the username, so values such as "adminXYZ" and "administrator" will succeed for the username.

Potential Mitigations

Phase: Implementation

Use expressions such as "sizeof(*pointer)" instead of "sizeof(pointer)", unless you intend to run sizeof() on a pointer type to gain some platform independence or if you are allocating a variable on the stack.

Other Notes

The use of sizeof() on a pointer can sometimes generate useful information. An obvious case is to find out the wordsize on a platform. More often than not, the appearance of sizeof(pointer) indicates a bug.

Weakness Ordinalities

Ordinality	Description
Primary	*(where the weakness exists independent of other weaknesses)*

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	465	Pointer Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	682	Incorrect Calculation	**Research Concepts (primary)1000**
ChildOf	Category	737	CERT C Secure Coding Section 03 - Expressions (EXP)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
CanPrecede	Weakness Base	131	Incorrect Calculation of Buffer Size	Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Use of sizeof() on a pointer type
CERT C Secure Coding	ARR01-C		Do not apply the sizeof operator to a pointer when taking the size of an array
CERT C Secure Coding	EXP01-C		Do not take the size of a pointer to determine the size of the pointed-to type

White Box Definitions

A weakness where code path has:

1. end statement that passes an identity of a dynamically allocated memory resource to a sizeof operator
2. start statement that allocates the dynamically allocated memory resource

References

Robert Seacord. "EXP01-A. Do not take the sizeof a pointer to determine the size of a type". <https://www.securecoding.cert.org/confluence/display/seccode/EXP01-A.+Do+not+take+the+sizeof+a+pointer+to+determine+the+size+of+a+type>.

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	CLASP		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		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

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

Heuristic 2nd Order Buffer Overflow read

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Potential Precision Problem

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Indicator of Poor Code Quality

Weakness ID: 398 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The code has features that do not directly introduce a weakness or vulnerability, but indicate that the product has not been carefully developed or maintained.

Extended Description

Programs are more likely to be secure when good development practices are followed. If a program is complex, difficult to maintain, not portable, or shows evidence of neglect, then there is a higher likelihood that weaknesses are buried in the code.

Time of Introduction

- Architecture and Design
- Implementation

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	18	Source Code	**Development Concepts (primary)699**
ChildOf	Weakness Class	710	Coding Standards Violation	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	107	Struts: Unused Validation Form	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	110	Struts: Validator Without Form Field	**Research Concepts (primary)1000**
ParentOf	Category	399	Resource Management Errors	**Development Concepts (primary)699**
ParentOf	Weakness Base	401	Failure to Release Memory Before Removing Last Reference ('Memory Leak')	**Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Base	404	Improper Resource Shutdown or Release	Development Concepts699 **Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Variant	415	Double Free	**Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Base	416	Use After Free	**Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Variant	457	Use of Uninitialized Variable	**Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Base	474	Use of Function with Inconsistent Implementations	**Development Concepts (primary)699 Seven Pernicious Kingdoms (primary)700 Research Concepts (primary)1000**
ParentOf	Weakness Base	475	Undefined Behavior for Input to API	**Development Concepts (primary)699 Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Base	476	NULL Pointer	**Development**

			Dereference	**Concepts (primary)699 Seven Pernicious Kingdoms (primary)700 Research Concepts (primary)1000**
ParentOf	Weakness Base	477	Use of Obsolete Functions	**Development Concepts (primary)699 Seven Pernicious Kingdoms (primary)700 Research Concepts (primary)1000**
ParentOf	Weakness Variant	478	Missing Default Case in Switch Statement	**Development Concepts (primary)699**
ParentOf	Weakness Variant	479	Unsafe Function Call from a Signal Handler	**Development Concepts (primary)699**
ParentOf	Weakness Variant	483	Incorrect Block Delimitation	**Development Concepts (primary)699**
ParentOf	Weakness Base	484	Omitted Break Statement in Switch	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Variant	546	Suspicious Comment	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	547	Use of Hard-coded, Security-relevant Constants	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	561	Dead Code	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	562	Return of Stack Variable Address	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Variant	563	Unused Variable	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Category	569	Expression Issues	**Development Concepts (primary)699**
ParentOf	Weakness Variant	585	Empty Synchronized Block	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	586	Explicit Call to Finalize()	**Development Concepts (primary)699**
ParentOf	Weakness Variant	617	Reachable Assertion	**Development Concepts (primary)699**
ParentOf	Weakness Base	676	Use of Potentially Dangerous Function	**Development Concepts (primary)699 Research Concepts (primary)1000**
MemberOf	View	700	Seven Pernicious Kingdoms	**Seven Pernicious Kingdoms (primary)700**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name

7 Pernicious Kingdoms			Code Quality

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		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Description, Relationships, Taxonomy Mappings		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Relationships		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2008-04-11	Code Quality		

BACK TO TOP

Improper Validation of Array Index

Weakness ID: 129 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The product uses untrusted input when calculating or using an array index, but the product does not validate or incorrectly validates the index to ensure the index references a valid position within the array.

Alternate Terms

out-of-bounds array index

index-out-of-range

array index underflow

Time of Introduction

- Implementation

Applicable Platforms

Languages

C: *(Often)*

C++: *(Often)*

Language-independent

Common Consequences

Scope	Effect
Integrity Availability	Unchecked array indexing will very likely result in the corruption of relevant memory and perhaps instructions, leading to a crash, if the values are outside of the valid memory area.
Integrity	If the memory corrupted is data, rather than instructions, the system will continue to function with improper values.
Confidentiality Integrity	Unchecked array indexing can also trigger out-of-bounds read or write operations, or operations on the wrong objects; i.e., "buffer overflows" are not always the result. This may result in the exposure or modification of sensitive data.
Integrity	If the memory accessible by the attacker can be effectively controlled, it may be possible to execute arbitrary code, as with a standard buffer overflow and possibly without the use of large inputs if a precise index can be controlled.
Integrity Availability Confidentiality	A single fault could allow either an overflow (CWE-788) or underflow (CWE-786) of the array index. What happens next will depend on the type of operation being performed out of bounds, but can expose sensitive information, cause a system crash, or possibly lead to arbitrary code execution.

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

This weakness can often be detected using automated static analysis tools. Many modern tools use data flow analysis or constraint-based techniques to minimize the number of false positives.

Automated static analysis generally does not account for environmental considerations when reporting out-of-bounds memory operations. This can make it difficult for users to determine which warnings should be investigated first. For example, an analysis tool might report array index errors that originate from command line arguments in a program that is not expected to run with setuid or other special privileges.

Effectiveness: High

This is not a perfect solution, since 100% accuracy and coverage are not feasible.

Automated Dynamic Analysis

This weakness can be detected using dynamic tools and techniques that interact with the software using large test suites with many diverse inputs, such as fuzz testing (fuzzing), robustness testing, and fault injection. The software's operation may slow down, but it should not become unstable, crash, or generate incorrect results.

Black Box

Black box methods might not get the needed code coverage within limited time constraints, and a dynamic test might not produce any noticeable side effects even if it is successful.

Demonstrative Examples

Example 1

The following C/C++ example retrieves the sizes of messages for a pop3 mail server. The message sizes are retrieved from a socket that returns in a buffer the message number and the message size, the message number (num) and size (size) are extracted from the buffer and the message size is placed into an array using the message number for the array index.

(Bad Code)

Example Language: **C**

```
/* capture the sizes of all messages */
int getsizes(int sock, int count, int *sizes) {
...
char buf[BUFFER_SIZE];
int ok;
int num, size;

// read values from socket and added to sizes array
while ((ok = gen_recv(sock, buf, sizeof(buf))) == 0)
{

// continue read from socket until buf only contains '.'
if (DOTLINE(buf))
break;
else if (sscanf(buf, "%d %d", &num, &size) == 2)
sizes[num - 1] = size;
}
...
}
```

In this example the message number retrieved from the buffer could be a value that is outside the allowable range of indices for the array and could possibly be a negative number. Without proper validation of the value to be used for the array index an array overflow could occur and could potentially lead to unauthorized access to memory addresses and system crashes. The value of the array index should be validated to ensure that it is within the allowable range of indices for the array as in the following code.

(Good Code)

Example Language: **C**

```
/* capture the sizes of all messages */
int getsizes(int sock, int count, int *sizes) {
...
char buf[BUFFER_SIZE];
int ok;
int num, size;

// read values from socket and added to sizes array
while ((ok = gen_recv(sock, buf, sizeof(buf))) == 0)
{

// continue read from socket until buf only contains '.'
if (DOTLINE(buf))
```

```
break;
else if (sscanf(buf, "%d %d", &num, &size) == 2) {
if (num > 0 && num <= (unsigned)count)
sizes[num - 1] = size;
else
/* warn about possible attempt to induce buffer overflow */
report(stderr, "Warning: ignoring bogus data for message sizes returned by server.\n");
}
}
...
}
```

Example 2

In the code snippet below, an unchecked integer value is used to reference an object in an array.

(Bad Code)

Example Language: **Java**

```
public String getValue(int index) {
return array[index];
}
```

If index is outside of the range of the array, this may result in an ArrayIndexOutOfBounds Exception being raised.

Example 3

In the following Java example the method displayProductSummary is called from a Web service servlet to retrieve product summary information for display to the user. The servlet obtains the integer value of the product number from the user and passes it to the displayProductSummary method. The displayProductSummary method passes the integer value of the product number to the getProductSummary method which obtains the product summary from the array object containing the project summaries using the integer value of the product number as the array index.

(Bad Code)

Example Language: **Java**

```
// Method called from servlet to obtain product information
public String displayProductSummary(int index) {

String productSummary = new String("");

try {
String productSummary = getProductSummary(index);

} catch (Exception ex) {...}

return productSummary;
}

public String getProductSummary(int index) {
return products[index];
}
```

In this example the integer value used as the array index that is provided by the user may be outside the allowable range of indices for the array which may provide unexpected results or may comes the application to fail. The integer value used for the array index should be validated to ensure that it is within the allowable range of indices for the array as in the following code.

(Good Code)

Example Language: **Java**

```
// Method called from servlet to obtain product information
public String displayProductSummary(int index) {

String productSummary = new String("");
```

```
try {
String productSummary = getProductSummary(index);

} catch (Exception ex) {...}

return productSummary;
}

public String getProductSummary(int index) {
String productSummary = "";

if ((index >= 0) && (index < MAX_PRODUCTS)) {
productSummary = products[index];
}
else {
System.err.println("index is out of bounds");
throw new IndexOutOfBoundsException();
}

return productSummary;
}
```

An alternative in Java would be to use one of the collection objects such as ArrayList that will automatically generate an exception if an attempt is made to access an array index that is out of bounds.

(Good Code)

Example Language: **Java**

```
ArrayList productArray = new ArrayList(MAX_PRODUCTS);
...
try {
productSummary = (String) productArray.get(index);
} catch (IndexOutOfBoundsException ex) {...}
```

Observed Examples

Reference	Description
CVE-2005-0369	large ID in packet used as array index
CVE-2001-1009	negative array index as argument to POP LIST command
CVE-2003-0721	Integer signedness error leads to negative array index
CVE-2004-1189	product does not properly track a count and a maximum number, which can lead to resultant array index overflow.
CVE-2007-5756	chain: device driver for packet-capturing software allows access to an unintended IOCTL with resultant array index error.

Potential Mitigations

Phase: Architecture and Design

Strategies: Input Validation; Libraries or Frameworks

Use an input validation framework such as Struts or the OWASP ESAPI Validation API. If you use Struts, be mindful of weaknesses covered by the CWE-101 category.

Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.

Even though client-side checks provide minimal benefits with respect to server-side security, they are still useful. First, they can support intrusion detection. If the server receives input that should have been rejected by the client, then it may be an indication of an attack. Second, client-side error-checking can provide helpful feedback to the user about the expectations for valid input. Third, there may be a reduction in server-side processing time for accidental input errors, although this is typically a small savings.

Phase: Requirements

Strategy: Language Selection

Use a language with features that can automatically mitigate or eliminate out-of-bounds indexing errors.

For example, Ada allows the programmer to constrain the values of a variable and languages such as Java and Ruby will allow the programmer to handle exceptions when an out-of-bounds index is accessed.

Phase: Implementation

Strategy: Input Validation

Assume all input is malicious. Use an "accept known good" input validation strategy (i.e., use a whitelist). Reject any input that does not strictly conform to specifications, or transform it into something that does. Use a blacklist to reject any unexpected inputs and detect potential attacks.

When accessing a user-controlled array index, use a stringent range of values that are within the target array. Make sure that you do not allow negative values to be used. That is, verify the minimum as well as the maximum of the range of acceptable values.

Phase: Implementation

Be especially careful to validate your input when you invoke code that crosses language boundaries, such as from an interpreted language to native code. This could create an unexpected interaction between the language boundaries. Ensure that you are not violating any of the expectations of the language with which you are interfacing. For example, even though Java may not be susceptible to buffer overflows, providing a large argument in a call to native code might trigger an overflow.

Weakness Ordinalities

Ordinality	Description
Resultant	The most common condition situation leading to unchecked array indexing is the use of loop index variables as buffer indexes. If the end condition for the loop is subject to a flaw, the index can grow or shrink unbounded, therefore causing a buffer overflow or underflow. Another common situation leading to this condition is the use of a function's return value, or the resulting value of a calculation directly as an index in to a buffer.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	20	Improper Input Validation	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	189	Numeric Errors	Development Concepts699
ChildOf	Category	633	Weaknesses that Affect Memory	**Resource-specific Weaknesses (primary)631**
ChildOf	Category	738	CERT C Secure Coding Section 04 - Integers (INT)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
ChildOf	Category	802	2010 Top 25 - Risky Resource Management	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
CanPrecede	Weakness Class	119	Failure to Constrain Operations within the Bounds of a Memory Buffer	Research Concepts1000
CanPrecede	Weakness Variant	789	Uncontrolled Memory Allocation	Research Concepts1000
PeerOf	Weakness Base	124	Buffer Underwrite ('Buffer Underflow')	Research Concepts1000

Theoretical Notes

An improperly validated array index might lead directly to the always-incorrect behavior of "access of array using out-of-bounds index."

Affected Resources

- Memory

f Causal Nature

Explicit

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Unchecked array indexing
PLOVER			INDEX - Array index overflow
CERT C Secure Coding	ARR00-C		Understand how arrays work
CERT C Secure Coding	ARR30-C		Guarantee that array indices are within the valid range
CERT C Secure Coding	ARR38-C		Do not add or subtract an integer to a pointer if the resulting value does not refer to a valid array element
CERT C Secure Coding	INT32-C		Ensure that operations on signed integers do not result in overflow

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
100	Overflow Buffers	

References

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 5, "Array Indexing Errors" Page 144. 2nd Edition. Microsoft. 2002.

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	CLASP		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Sean Eidemiller	Cigital	External
	added/updated demonstrative examples		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Common Consequences		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Description, Name, Relationships		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Observed Examples, Other Notes, Potential Mitigations, Theoretical Notes, Weakness Ordinalities		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Demonstrative Examples, Detection Factors, Likelihood of Exploit, Potential Mitigations, References, Related Attack Patterns, Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Related Attack Patterns		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2009-10-29	Unchecked Array Indexing		

BACK TO TOP

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