vifm Scan Report

Project Name	vifm
Scan Start	Friday, June 21, 2024 11:52:53 AM
Preset	Checkmarx Default
Scan Time	00h:02m:43s
Lines Of Code Scanned	12738
Files Scanned	15
Report Creation Time	Friday, June 21, 2024 12:05:08 PM
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011
Team	CxServer
Checkmarx Version	8.7.0
Scan Type	Full
Source Origin	LocalPath
Density	3/100 (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

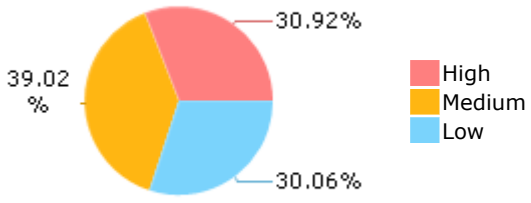
Results Limit

Results limit per query was set to 50

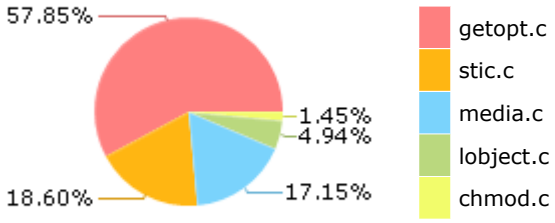
Selected Queries

Selected queries are listed in Result Summary

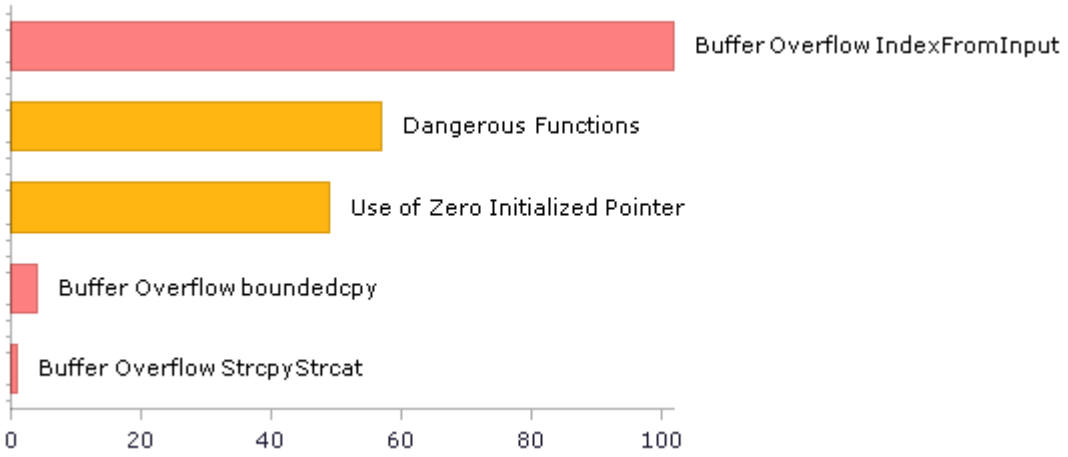
Result Summary



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	144	34
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	34	34
A3-Sensitive Data Exposure	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	1	1
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	57	57
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	3	1
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	57	57
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	19	12
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.	16	16
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.	18	18
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.	1	1
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.	3	1

***** 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)	34	34
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)	1	1
SC-4 Information in Shared Resources (P1)	0	0
SC-5 Denial of Service Protection (P1)*	64	17
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*	24	15
SI-11 Error Handling (P2)*	32	32
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

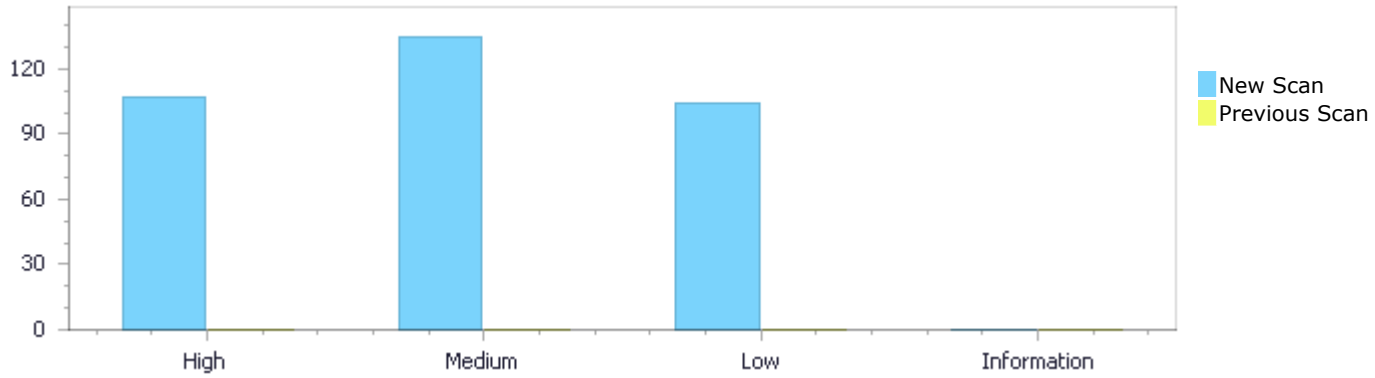
Scan Summary - Custom

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	107	135	104	0	346
Recurrent Issues	0	0	0	0	0
Total	107	135	104	0	346
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	107	135	104	0	346
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	107	135	104	0	346

Result Summary

Vulnerability Type	Occurrences	Severity
Buffer Overflow IndexFromInput	102	High
Buffer Overflow boundedcpy	4	High
Buffer Overflow StrcpyStrcat	1	High
Dangerous Functions	57	Medium
Use of Zero Initialized Pointer	49	Medium

Use After Free	10	Medium
Buffer Overflow boundcpy WrongSizeParam	7	Medium
MemoryFree on StackVariable	6	Medium
Environment Injection	3	Medium
Memory Leak	3	Medium
Unchecked Return Value	32	Low
Improper Resource Access Authorization	18	Low
Incorrect Permission Assignment For Critical Resources	16	Low
TOCTOU	16	Low
Potential Precision Problem	9	Low
Heuristic Buffer Overflow malloc	7	Low
Inconsistent Implementations	2	Low
Arithmenic Operation On Boolean	1	Low
NULL Pointer Dereference	1	Low
Use of Insufficiently Random Values	1	Low
Use of Sizeof On a Pointer Type	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
vifm/getopt.c	188
vifm/stic.c	31
vifm/lobject.c	16
vifm/chmod.c	4
vifm/media.c	3

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=1020015&projectid=20011&pathid=5
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	264
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=6
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	264
Object	argv	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 3:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=7
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	264
Object	getenv	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 4:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=8
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	264
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 5:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=9
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c

Line	1223	264
Object	argv	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 6:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=10
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	264
Object	getenv	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 7:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=11
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	263
Object	argc	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method main (int argc, char **argv)

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
263.              argv[middle + i] = tem;
```

Buffer Overflow IndexFromInput\Path 8:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=12
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	263
Object	argv	BinaryExpr

Code Snippet

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

```
....
1223.  main (int argc, char **argv)
```

File Name	vifm/getopt.c
Method	exchange (char **argv, struct _getopt_data *d)

```
....
263.              argv[middle + i] = tem;
```

Buffer Overflow IndexFromInput\Path 9:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=13
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	263
Object	getenv	BinaryExpr

Code Snippet

File Name	vifm/getopt.c
Method	_getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name	vifm/getopt.c
Method	exchange (char **argv, struct _getopt_data *d)

```
....
263.              argv[middle + i] = tem;
```

Buffer Overflow IndexFromInput\Path 10:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=200

	11&pathid=14
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	262
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
262.              argv[bottom + i] = argv[middle + i];
```

Buffer Overflow IndexFromInput\Path 11:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=15
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	262
Object	argv	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
262.              argv[bottom + i] = argv[middle + i];
```

Buffer Overflow IndexFromInput\Path 12:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=16
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	262
Object	getenv	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
262.              argv[bottom + i] = argv[middle + i];
```

Buffer Overflow IndexFromInput\Path 13:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=17
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	247
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
247.              SWAP_FLAGS (bottom + i, top - (middle - bottom) + i);
```

Buffer Overflow IndexFromInput\Path 14:

Severity High
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=18
Status New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	247
Object	argv	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
247.              SWAP_FLAGS (bottom + i, top - (middle - bottom) + i);
```

Buffer Overflow IndexFromInput\Path 15:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=19
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	247
Object	getenv	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
247.              SWAP_FLAGS (bottom + i, top - (middle - bottom) + i);
```

Buffer Overflow IndexFromInput\Path 16:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=20
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c

Line	1223	247
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
247.              SWAP_FLAGS (bottom + i, top - (middle - bottom) + i);
```

Buffer Overflow IndexFromInput\Path 17:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=21
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	247
Object	argv	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
247.              SWAP_FLAGS (bottom + i, top - (middle - bottom) + i);
```

Buffer Overflow IndexFromInput\Path 18:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=22
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	247
Object	getenv	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
247.              SWAP_FLAGS (bottom + i, top - (middle - bottom) + i);
```

Buffer Overflow IndexFromInput\Path 19:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=23
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	245
Object	argc	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method main (int argc, char **argv)

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
245.              argv[bottom + i] = argv[top - (middle - bottom) + i];
```

Buffer Overflow IndexFromInput\Path 20:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=24
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	245
Object	argv	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method main (int argc, char **argv)

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
245.              argv[bottom + i] = argv[top - (middle - bottom) + i];
```

Buffer Overflow IndexFromInput\Path 21:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=25
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	245
Object	getenv	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
245.              argv[bottom + i] = argv[top - (middle - bottom) + i];
```

Buffer Overflow IndexFromInput\Path 22:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=26
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	244
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name	vifm/getopt.c
Method	exchange (char **argv, struct _getopt_data *d)

```
....
244.              tem = argv[bottom + i];
```

Buffer Overflow IndexFromInput\Path 23:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=27
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	244
Object	argv	BinaryExpr

Code Snippet

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

```
....
1223.  main (int argc, char **argv)
```

File Name	vifm/getopt.c
Method	exchange (char **argv, struct _getopt_data *d)

```
....
244.              tem = argv[bottom + i];
```

Buffer Overflow IndexFromInput\Path 24:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=28
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination

File	vifm/getopt.c	vifm/getopt.c
Line	291	244
Object	getenv	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
244.              tem = argv[bottom + i];
```

Buffer Overflow IndexFromInput\Path 25:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=29
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	264
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 26:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=30
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	264
Object	argv	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 27:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=31
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	264

Object	getenv	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 28:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=32
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	264
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 29:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=33
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	264
Object	argv	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method main (int argc, char **argv)

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 30:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=34
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	264
Object	getenv	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
264.              SWAP_FLAGS (bottom + i, middle + i);
```

Buffer Overflow IndexFromInput\Path 31:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=35
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	262
Object	argc	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method main (int argc, char **argv)

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
262.              argv[bottom + i] = argv[middle + i];
```

Buffer Overflow IndexFromInput\Path 32:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=36
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	262
Object	argv	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
262.              argv[bottom + i] = argv[middle + i];
```

Buffer Overflow IndexFromInput\Path 33:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=37
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	262
Object	getenv	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name	vifm/getopt.c
Method	exchange (char **argv, struct _getopt_data *d)

```
....
262.              argv[bottom + i] = argv[middle + i];
```

Buffer Overflow IndexFromInput\Path 34:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=38
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	261
Object	argc	BinaryExpr

Code Snippet

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

```
....
1223.  main (int argc, char **argv)
```

File Name	vifm/getopt.c
Method	exchange (char **argv, struct _getopt_data *d)

```
....
261.              tem = argv[bottom + i];
```

Buffer Overflow IndexFromInput\Path 35:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=39
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination

File	vifm/getopt.c	vifm/getopt.c
Line	1223	261
Object	argv	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
261.              tem = argv[bottom + i];
```

Buffer Overflow IndexFromInput\Path 36:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=40
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	261
Object	getenv	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
261.              tem = argv[bottom + i];
```

Buffer Overflow IndexFromInput\Path 37:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=41
Status	New

The size of the buffer used by _getopt_internal_r in PostfixExpr, at line 400 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	504
Object	argc	PostfixExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
504.          d->optarg = argv[d->optind++];
```

Buffer Overflow IndexFromInput\Path 38:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=42
Status	New

The size of the buffer used by _getopt_internal_r in PostfixExpr, at line 400 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	504

Object	argv	PostfixExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
504.          d->optarg = argv[d->optind++];
```

Buffer Overflow IndexFromInput\Path 39:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=43
Status	New

The size of the buffer used by _getopt_internal_r in PostfixExpr, at line 400 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	504
Object	getenv	PostfixExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
504.          d->optarg = argv[d->optind++];
```

Buffer Overflow IndexFromInput\Path 40:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=44
Status	New

The size of the buffer used by _getopt_internal_r in optind, at line 400 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	602
Object	argc	optind

Code Snippet

File Name vifm/getopt.c

Method main (int argc, char **argv)

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
602.                        argv[0], argv[d->optind]);
```

Buffer Overflow IndexFromInput\Path 41:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=45
Status	New

The size of the buffer used by _getopt_internal_r in optind, at line 400 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	602
Object	argv	optind

Code Snippet

File Name vifm/getopt.c

Method main (int argc, char **argv)

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
602.                          argv[0], argv[d->optind]);
```

Buffer Overflow IndexFromInput\Path 42:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=46
Status	New

The size of the buffer used by _getopt_internal_r in optind, at line 400 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	602
Object	getenv	optind

Code Snippet

File Name vifm/getopt.c

Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
602.                          argv[0], argv[d->optind]);
```

Buffer Overflow IndexFromInput\Path 43:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=47
Status	New

The size of the buffer used by _getopt_internal_r in PostfixExpr, at line 400 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	724
Object	argc	PostfixExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
724.              d->optarg = argv[d->optind++];
```

Buffer Overflow IndexFromInput\Path 44:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=48
Status	New

The size of the buffer used by _getopt_internal_r in PostfixExpr, at line 400 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	724
Object	argv	PostfixExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name	vifm/getopt.c
Method	_getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
724.              d->optarg = argv[d->optind++];
```

Buffer Overflow IndexFromInput\Path 45:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=49
Status	New

The size of the buffer used by _getopt_internal_r in PostfixExpr, at line 400 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	724
Object	getenv	PostfixExpr

Code Snippet

File Name	vifm/getopt.c
Method	_getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name	vifm/getopt.c
Method	_getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
724.              d->optarg = argv[d->optind++];
```

Buffer Overflow IndexFromInput\Path 46:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=50
Status	New

The size of the buffer used by _getopt_internal_r in optind, at line 400 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination

File	vifm/getopt.c	vifm/getopt.c
Line	1223	801
Object	argc	optind

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
801.                              argv[0], argv[d->optind][0], d-
>__nextchar);
```

Buffer Overflow IndexFromInput\Path 47:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=51
Status	New

The size of the buffer used by _getopt_internal_r in optind, at line 400 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	801
Object	argv	optind

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
801.                              argv[0], argv[d->optind][0], d-
>__nextchar);
```

Buffer Overflow IndexFromInput\Path 48:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=52
Status	New

The size of the buffer used by _getopt_internal_r in optind, at line 400 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	801
Object	getenv	optind

Code Snippet

File Name vifm/getopt.c

Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
801.                              argv[0], argv[d->optind][0], d-
>__nextchar);
```

Buffer Overflow IndexFromInput\Path 49:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=53
Status	New

The size of the buffer used by _getopt_internal_r in optind, at line 400 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination

File	vifm/getopt.c	vifm/getopt.c
Line	1223	785
Object	argc	optind

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
785.              if (argv[d->optind][1] == '-')
```

Buffer Overflow IndexFromInput\Path 50:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=54
Status	New

The size of the buffer used by _getopt_internal_r in optind, at line 400 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	785
Object	argv	optind

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
785.              if (argv[d->optind][1] == '-')
```

Buffer Overflow boundedcpy

Query Path:
CPP\Cx\CPP Buffer Overflow\Buffer Overflow boundedcpy 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 boundedcpy\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=1
Status	New

The size parameter BinaryExpr in line 199 in file vifm/getopt.c is influenced by the user input argc in line 1223 in file vifm/getopt.c. This may lead to a buffer overflow vulnerability, which may in turn result in malicious code execution.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	226
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
226.                '\0', top + 1 - d->__nonoption_flags_max_len);
```

Buffer Overflow boundedcpy\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=2
Status	New

The size parameter BinaryExpr in line 199 in file vifm/getopt.c is influenced by the user input argv in line 1223 in file vifm/getopt.c. This may lead to a buffer overflow vulnerability, which may in turn result in malicious code execution.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	226
Object	argv	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
226.                '\0', top + 1 - d->__nonoption_flags_max_len);
```

Buffer Overflow boundedcpy\Path 3:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=3
Status	New

The size parameter BinaryExpr in line 199 in file vifm/getopt.c is influenced by the user input getenv in line 280 in file vifm/getopt.c. This may lead to a buffer overflow vulnerability, which may in turn result in malicious code execution.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	226
Object	getenv	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
226.                '\0', top + 1 - d->__nonoption_flags_max_len);
```

Buffer Overflow boundedcpy\Path 4:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=4
Status	New

The size parameter BinaryExpr in line 280 in file vifm/getopt.c is influenced by the user input argc in line 1223 in file vifm/getopt.c. This may lead to a buffer overflow vulnerability, which may in turn result in malicious code execution.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	331
Object	argc	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method main (int argc, char **argv)

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
331.                    '\0', d->__nonoption_flags_max_len - len);
```

Buffer Overflow StrcpyStrcat

Query Path:
CPP\Cx\CPP Buffer Overflow\Buffer Overflow StrcpyStrcat 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 StrcpyStrcat\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=156
Status	New

The size of the buffer used by set_magic_marker in marker, at line 493 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that set_magic_marker passes to marker, at line 493 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	493	496
Object	marker	marker

Code Snippet
File Name vifm/stic.c
Method void set_magic_marker(char* marker)

```
....
493.  void set_magic_marker(char* marker)
....
496.        strcpy(stic_magic_marker, marker);
```

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=1020015&projectid=20011&pathid=175
Status	New

The dangerous function, alloca, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	577	577
Object	alloca	alloca

Code Snippet

File Name	vifm/getopt.c
Method	_getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
577.              struct option_list *newp = alloca (sizeof (*newp));
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=176
Status	New

The dangerous function, memcpy, was found in use at line 557 in vifm/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	561	561
Object	memcpy	memcpy

Code Snippet	
File Name	vifm/lobject.c
Method	void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
561.        memcpy(out, source + 1, srclen * sizeof(char));
```

Dangerous Functions\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=177
Status	New

The dangerous function, memcpy, was found in use at line 557 in vifm/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	569	569
Object	memcpy	memcpy

Code Snippet	
File Name	vifm/lobject.c
Method	void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
569.        memcpy(out, source + 1, srclen * sizeof(char));
```

Dangerous Functions\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=178
Status	New

The dangerous function, memcpy, was found in use at line 557 in vifm/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	573	573
Object	memcpy	memcpy

Code Snippet
File Name vifm/lobject.c
Method void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
573.        memcpy(out, source + 1 + srclen - bufflen, bufflen *
sizeof(char));
```

Dangerous Functions\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=179
Status	New

The dangerous function, memcpy, was found in use at line 557 in vifm/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	589	589
Object	memcpy	memcpy

Code Snippet
File Name vifm/lobject.c
Method void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
589.      memcpy(out, POS, (LL(POS) + 1) * sizeof(char));
```

Dangerous Functions\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=180
Status	New

The dangerous function, memcpy, was found in use at line 443 in vifm/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	446	446
Object	memcpy	memcpy

Code Snippet
File Name vifm/lobject.c
Method static void addstr2buff (BuffFS *buff, const char *str, size_t slen) {

```
....
446.      memcpy(bf, str, slen);  /* add string to buffer */
```

Dangerous Functions\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=181
Status	New

The dangerous function, sprintf, was found in use at line 296 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	299	299
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_int_equal(int expected, int actual, const char* function, const char file[], unsigned int line)

```
....
299.        sprintf(s, "Expected %d but was %d", expected, actual);
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=182
Status	New

The dangerous function, sprintf, was found in use at line 303 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	306	306
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_ulong_equal(unsigned long expected, unsigned long actual, const char* function, const char file[], unsigned int line)

```
....
306.        sprintf(s, "Expected %lu but was %lu", expected, actual);
```

Dangerous Functions\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=183
Status	New

The dangerous function, sprintf, was found in use at line 310 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	314	314
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_float_equal(float expected, float actual, float delta, const char* function, const char file[], unsigned int line)

```
....
314.        sprintf(s, "Expected %f but was %f", expected, actual);
```

Dangerous Functions\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=184
Status	New

The dangerous function, sprintf, was found in use at line 319 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	323	323
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_double_equal(double expected, double actual, double delta, const char* function, const char file[], unsigned int line)

```
....
323.        sprintf(s, "Expected %f but was %f", expected, actual);
```

Dangerous Functions\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=185
Status	New

The dangerous function, sprintf, was found in use at line 328 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	335	335
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
335.            sprintf(s, "Expected <NULL> but was <NULL>");
```

Dangerous Functions\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=186
Status	New

The dangerous function, sprintf, was found in use at line 328 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	340	340
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
340.          sprintf(s, "Expected <NULL> but was \"%s\"", actual);
```

Dangerous Functions\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=187
Status	New

The dangerous function, sprintf, was found in use at line 328 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	345	345
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
345.          sprintf(s, "Expected \"%s\" but was <NULL>", expected);
```

Dangerous Functions\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=188
Status	New

The dangerous function, sprintf, was found in use at line 328 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	351	351
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
351.          sprintf(s, "Expected \"%s\" but was \"%s\"", expected,
actual);
```

Dangerous Functions\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=189
Status	New

The dangerous function, sprintf, was found in use at line 357 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	364	364
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_wstring_equal(const wchar_t expected[], const wchar_t actual[], const char function[], const char file[], unsigned int line)

```
....
364.              sprintf(s, "Expected <NULL> but was <NULL>");
```

Dangerous Functions\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=190
Status	New

The dangerous function, sprintf, was found in use at line 357 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	370	370
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_wstring_equal(const wchar_t expected[], const wchar_t actual[], const char function[], const char file[], unsigned int line)

```
....
370.              sprintf(s, "Expected <NULL> but was \"%ls\"", actual);
```

Dangerous Functions\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=191
Status	New

The dangerous function, sprintf, was found in use at line 357 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	379	379
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_wstring_equal(const wchar_t expected[], const wchar_t actual[], const char function[], const char file[], unsigned int line)

```
....
379.                sprintf(s, "Expected \"%ls\" but was <NULL>",
expected);
```

Dangerous Functions\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=192
Status	New

The dangerous function, sprintf, was found in use at line 357 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	389	389
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_wstring_equal(const wchar_t expected[], const wchar_t actual[], const char function[], const char file[], unsigned int line)

```
....
389.                sprintf(s, "Expected \"%ls\" but was \"%ls\"",
expected, actual);
```

Dangerous Functions\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=193
Status	New

The dangerous function, sprintf, was found in use at line 398 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	401	401
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method	void stic_assert_string_ends_with(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
401.        sprintf(s, "Expected \"%s\" to end with \"%s\"", actual,
expected);
```

Dangerous Functions\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=194
Status	New

The dangerous function, sprintf, was found in use at line 405 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	408	408
Object	sprintf	sprintf

Code Snippet	
File Name	vifm/stic.c
Method	void stic_assert_string_starts_with(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
408.        sprintf(s, "Expected \"%s\" to start with \"%s\"", actual,
expected);
```

Dangerous Functions\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=195
Status	New

The dangerous function, sprintf, was found in use at line 412 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	415	415
Object	sprintf	sprintf

Code Snippet

File Name	vifm/stic.c
Method	void stic_assert_string_contains(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
415.        sprintf(s, "Expected \"%s\" to be in \"%s\"", expected,
actual);
```

Dangerous Functions\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=196
Status	New

The dangerous function, sprintf, was found in use at line 419 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	422	422
Object	sprintf	sprintf

Code Snippet	
File Name	vifm/stic.c
Method	void stic_assert_string_doesnt_contain(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
422.        sprintf(s, "Expected \"%s\" not to have \"%s\" in it",
actual, expected);
```

Dangerous Functions\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=197
Status	New

The dangerous function, sprintf, was found in use at line 447 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	473	473
Object	sprintf	sprintf

Code Snippet	
File Name	vifm/stic.c
Method	void stic_test_fixture_end()

```
....
473.        sprintf(s, "%d test%s run  %d check%s failed", nrun, nrun ==
1 ? "" : "s",
```

Dangerous Functions\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=198
Status	New

The dangerous function, sprintf, was found in use at line 525 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	563	563
Object	sprintf	sprintf

Code Snippet	
File Name	vifm/stic.c
Method	int run_tests(stic_void_void tests)

```
....
563.              sprintf(time,"< 1 ms");
```

Dangerous Functions\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=199
Status	New

The dangerous function, sprintf, was found in use at line 525 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	567	567
Object	sprintf	sprintf

Code Snippet	
File Name	vifm/stic.c

Method	int run_tests(stic_void_void tests)

```
....
567.              sprintf(time,"%lu ms",end - start);
```

Dangerous Functions\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=200
Status	New

The dangerous function, sprintf, was found in use at line 525 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	570	570
Object	sprintf	sprintf

Code Snippet	
File Name	vifm/stic.c
Method	int run_tests(stic_void_void tests)

```
....
570.          sprintf(s,"%d check%s :: %d run test%s :: %d skipped test%s
:: %s",
```

Dangerous Functions\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=201
Status	New

The dangerous function, strcpy, was found in use at line 68 in vifm/chmod.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/chmod.c	vifm/chmod.c
Line	85	85
Object	strcpy	strcpy

Code Snippet	
File Name	vifm/chmod.c
Method	set_file_perms(const int perms[13])

```
....
85.    strcpy(lwin.curr_dir, SANDBOX_PATH);
```

Dangerous Functions\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=202
Status	New

The dangerous function, strcpy, was found in use at line 95 in vifm/chmod.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/chmod.c	vifm/chmod.c
Line	117	117
Object	strcpy	strcpy

Code Snippet
File Name vifm/chmod.c
Method TEST(reset_executable_bits_from_files_only, IF(can_reset_x_on_files))

```
....
117.         strcpy(lwin.curr_dir, SANDBOX_PATH);
```

Dangerous Functions\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=203
Status	New

The dangerous function, strcpy, was found in use at line 129 in vifm/chmod.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/chmod.c	vifm/chmod.c
Line	151	151
Object	strcpy	strcpy

Code Snippet
File Name vifm/chmod.c
Method TEST(set_executable_bit_via_X_flag)

```
....
151.        strcpy(lwin.curr_dir, SANDBOX_PATH);
```

Dangerous Functions\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=204
Status	New

The dangerous function, strcpy, was found in use at line 251 in vifm/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	263	263
Object	strcpy	strcpy

Code Snippet
File Name vifm/lobject.c
Method static const char *l_str2d (const char *s, lua_Number *result) {

```
....
263.      strcpy(buff, s);  /* copy string to buffer */
```

Dangerous Functions\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=205
Status	New

The dangerous function, strcpy, was found in use at line 467 in vifm/media.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	483	483
Object	strcpy	strcpy

Code Snippet
File Name vifm/media.c
Method TEST(barckets_navigates_between_devices)

```
....
483.        strcpy(lwin.curr_dir, sandbox);
```

Dangerous Functions\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=206
Status	New

The dangerous function, strcpy, was found in use at line 410 in vifm/media.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	424	424
Object	strcpy	strcpy

Code Snippet
File Name vifm/media.c
Method TEST(mount_directory_is_left_before_unmounting)

```
....
424.        strcpy(lwin.curr_dir, sandbox);
```

Dangerous Functions\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=207
Status	New

The dangerous function, strcpy, was found in use at line 449 in vifm/media.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	459	459
Object	strcpy	strcpy

Code Snippet
File Name vifm/media.c
Method TEST(mount_matching_current_path_is_picked_by_default)

```
....
459.        strcpy(lwin.curr_dir, sandbox);
```

Dangerous Functions\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=208
Status	New

The dangerous function, strcpy, was found in use at line 493 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	496	496
Object	strcpy	strcpy

Code Snippet
File Name vifm/stic.c
Method void set_magic_marker(char* marker)

```
....
496.        strcpy(stic_magic_marker, marker);
```

Dangerous Functions\Path 35:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=209
Status	New

The dangerous function, strlen, was found in use at line 280 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	322	322
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
322.              int len = d->__nonoption_flags_max_len = strlen
(orig_str);
```

Dangerous Functions\Path 36:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=210
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	557	557
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
557.            if (namelen == (unsigned int) strlen (p->name))
```

Dangerous Functions\Path 37:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=211
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	642	642
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
642.          d->__nextchar += strlen (d->__nextchar);
```

Dangerous Functions\Path 38:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=212
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	715	715
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
715.                d->__nextchar += strlen (d->__nextchar);
```

Dangerous Functions\Path 39:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=213
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	755	755
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
755.                d->__nextchar += strlen (d->__nextchar);
```

Dangerous Functions\Path 40:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=214
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	760	760
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
760.          d->__nextchar += strlen (d->__nextchar);
```

Dangerous Functions\Path 41:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=215
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	953	953
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
953.              if ((unsigned int) (nameend - d->__nextchar) == strlen
(p->name))
```

Dangerous Functions\Path 42:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=216
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1001	1001
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1001.           d->__nextchar += strlen (d->__nextchar);
```

Dangerous Functions\Path 43:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=217
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1045	1045
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1045.                   d->__nextchar += strlen (d->__nextchar);
```

Dangerous Functions\Path 44:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=218
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1083	1083
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1083.                   d->__nextchar += strlen (d->__nextchar);
```

Dangerous Functions\Path 45:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=219
Status	New

The dangerous function, strlen, was found in use at line 400 in vifm/getopt.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1089	1089
Object	strlen	strlen

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1089.           d->__nextchar += strlen (d->__nextchar);
```

Dangerous Functions\Path 46:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=220
Status	New

The dangerous function, strlen, was found in use at line 251 in vifm/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	261	261
Object	strlen	strlen

Code Snippet
File Name vifm/lobject.c
Method static const char *l_str2d (const char *s, lua_Number *result) {

```
....
261.      if (pdot == NULL || strlen(s) > L_MAXLENNUM)
```

Dangerous Functions\Path 47:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=221
Status	New

The dangerous function, strlen, was found in use at line 470 in vifm/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	481	481
Object	strlen	strlen

Code Snippet
File Name vifm/lobject.c
Method const char *luaO_pushvfstring (lua_State *L, const char *fmt, va_list argp) {

```
....
481.          addstr2buff(&buff, s, strlen(s));
```

Dangerous Functions\Path 48:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=222
Status	New

The dangerous function, strlen, was found in use at line 470 in vifm/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	532	532
Object	strlen	strlen

Code Snippet
File Name vifm/lobject.c
Method const char *luaO_pushvfstring (lua_State *L, const char *fmt, va_list argp) {

```
....
532.    addstr2buff(&buff, fmt, strlen(fmt));  /* rest of 'fmt' */
```

Dangerous Functions\Path 49:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=223
Status	New

The dangerous function, strlen, was found in use at line 111 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	113	113
Object	strlen	strlen

Code Snippet
File Name vifm/stic.c
Method static void stic_header_printer(const char s[], int length, char f)

```
....
113.        int l = strlen(s);
```

Dangerous Functions\Path 50:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=224
Status	New

The dangerous function, strlen, was found in use at line 173 in vifm/stic.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	175	175
Object	strlen	strlen

Code Snippet
File Name vifm/stic.c
Method static const char * test_file_name(const char path[])

```
....
175.        const char * file = path + strlen(path);
```

Use of Zero Initialized Pointer

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

Categories

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

Description

Use of Zero Initialized Pointer\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=248
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 280 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	289	642

Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
289.    d->__nextchar = NULL;
```

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
642.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=249
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	409	642
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
409.    d->optarg = NULL;
....
642.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=250
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1088	642
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1088.             d->optarg = NULL;
....
642.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 4:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=251
Status New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1101	642
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1101.       d->__nextchar = NULL;
....
642.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 5:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=252
Status New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1115	642
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1115.             d->optarg = NULL;
....
642.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 6:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=253
Status New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1116	642
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1116.           d->__nextchar = NULL;
....
642.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 7:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=254
Status New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1167	642
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1167.           d->__nextchar = NULL;
....
642.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=255
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 280 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	289	760
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
289.    d->__nextchar = NULL;
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
760.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=256
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	409	760
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
409.    d->optarg = NULL;
....
760.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=257
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1088	760
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1088.             d->optarg = NULL;
....
760.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=258
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1101	760
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1101.       d->__nextchar = NULL;
....
760.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=259
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1115	760
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1115.             d->optarg = NULL;
....
760.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=260
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1116	760
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1116.           d->__nextchar = NULL;
....
760.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=261
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1167	760
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1167.           d->__nextchar = NULL;
....
760.          d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=262
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 280 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	289	715
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
289.    d->__nextchar = NULL;
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
715.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=263
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	409	715
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
409.    d->optarg = NULL;
....
715.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=264
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1088	715
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1088.             d->optarg = NULL;
....
715.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=265
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1101	715
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1101.       d->__nextchar = NULL;
....
715.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=266
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1115	715
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1115.             d->optarg = NULL;
....
715.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=267
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1116	715
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1116.           d->__nextchar = NULL;
....
715.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=268
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1167	715
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1167.           d->__nextchar = NULL;
....
715.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=269
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 280 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	289	755
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
289.    d->__nextchar = NULL;
```

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
755.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=270
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	409	755
Object	optarg	__nextchar

Code Snippet

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
409.    d->optarg = NULL;
....
755.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=271
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c

Line	1088	755
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1088.             d->optarg = NULL;
....
755.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=272
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1101	755
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1101.       d->__nextchar = NULL;
....
755.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=273
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination

File	vifm/getopt.c	vifm/getopt.c
Line	1115	755
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1115.             d->optarg = NULL;
....
755.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=274
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1116	755
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1116.           d->__nextchar = NULL;
....
755.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=275
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1167	755
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1167.           d->__nextchar = NULL;
....
755.                d->__nextchar += strlen (d->__nextchar);
```

Use of Zero Initialized Pointer\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=276
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 280 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	289	835
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
289.    d->__nextchar = NULL;
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
835.      char c = *d->__nextchar++;
```

Use of Zero Initialized Pointer\Path 30:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=277
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	409	835
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
409.    d->optarg = NULL;
....
835.      char c = *d->__nextchar++;
```

Use of Zero Initialized Pointer\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=278
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1088	835
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1088.             d->optarg = NULL;
....
835.      char c = *d->__nextchar++;
```

Use of Zero Initialized Pointer\Path 32:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=279
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1101	835
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1101.       d->__nextchar = NULL;
....
835.      char c = *d->__nextchar++;
```

Use of Zero Initialized Pointer\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=280
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1115	835
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1115.             d->optarg = NULL;
....
835.      char c = *d->__nextchar++;
```

Use of Zero Initialized Pointer\Path 34:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=281
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1116	835
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1116.           d->__nextchar = NULL;
....
835.      char c = *d->__nextchar++;
```

Use of Zero Initialized Pointer\Path 35:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=282
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1167	835
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1167.           d->__nextchar = NULL;
....
835.      char c = *d->__nextchar++;
```

Use of Zero Initialized Pointer\Path 36:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=283
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 280 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	289	550
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
289.    d->__nextchar = NULL;
```

File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
550.        namelen = nameend - d->__nextchar;
```

Use of Zero Initialized Pointer\Path 37:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=284
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	409	550
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
409.    d->optarg = NULL;
....
550.        namelen = nameend - d->__nextchar;
```

Use of Zero Initialized Pointer\Path 38:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=285
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1088	550
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1088.             d->optarg = NULL;
....
550.        namelen = nameend - d->__nextchar;
```

Use of Zero Initialized Pointer\Path 39:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=286
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1101	550
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1101.       d->__nextchar = NULL;
....
550.        namelen = nameend - d->__nextchar;
```

Use of Zero Initialized Pointer\Path 40:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=287
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1115	550
Object	optarg	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1115.             d->optarg = NULL;
....
550.        namelen = nameend - d->__nextchar;
```

Use of Zero Initialized Pointer\Path 41:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=288
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1116	550
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1116.           d->__nextchar = NULL;
....
550.        namelen = nameend - d->__nextchar;
```

Use of Zero Initialized Pointer\Path 42:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=289
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by __nextchar at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1167	550
Object	__nextchar	__nextchar

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1167.           d->__nextchar = NULL;
....
550.        namelen = nameend - d->__nextchar;
```

Use of Zero Initialized Pointer\Path 43:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=290
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 280 is not initialized when it is used by optarg at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	289	657
Object	__nextchar	optarg

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
289.    d->__nextchar = NULL;
```

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
657.              d->optarg = nameend + 1;
```

Use of Zero Initialized Pointer\Path 44:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=291
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by optarg at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	409	657
Object	optarg	optarg

Code Snippet

File Name vifm/getopt.c

Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
409.    d->optarg = NULL;
....
657.              d->optarg = nameend + 1;
```

Use of Zero Initialized Pointer\Path 45:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=292
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by optarg at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c

Line	1088	657
Object	optarg	optarg

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1088.             d->optarg = NULL;
....
657.              d->optarg = nameend + 1;
```

Use of Zero Initialized Pointer\Path 46:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=293
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by optarg at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1101	657
Object	__nextchar	optarg

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1101.       d->__nextchar = NULL;
....
657.              d->optarg = nameend + 1;
```

Use of Zero Initialized Pointer\Path 47:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=294
Status	New

The variable declared in optarg at vifm/getopt.c in line 400 is not initialized when it is used by optarg at vifm/getopt.c in line 400.

	Source	Destination

File	vifm/getopt.c	vifm/getopt.c
Line	1115	657
Object	optarg	optarg

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1115.             d->optarg = NULL;
....
657.              d->optarg = nameend + 1;
```

Use of Zero Initialized Pointer\Path 48:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=295
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by optarg at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1116	657
Object	__nextchar	optarg

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1116.           d->__nextchar = NULL;
....
657.              d->optarg = nameend + 1;
```

Use of Zero Initialized Pointer\Path 49:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=296
Status	New

The variable declared in __nextchar at vifm/getopt.c in line 400 is not initialized when it is used by optarg at vifm/getopt.c in line 400.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1167	657
Object	__nextchar	optarg

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1167.           d->__nextchar = NULL;
....
657.              d->optarg = nameend + 1;
```

Use After Free

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

Categories

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

Description

Use After Free\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=238
Status	New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	625	620
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
625.                    free (buf);
....
620.                    __fxprintf (NULL, "%s", buf);
```

Use After Free\Path 2:

Severity	Medium

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=239
Status	New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	710	705
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
710.                    free (buf);
....
705.                    __fxprintf (NULL, "%s", buf);
```

Use After Free\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=240
Status	New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	747	742
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
747.                    free (buf);
....
742.                    __fxprintf (NULL, "%s", buf);
```

Use After Free\Path 4:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=241
Status	New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	821	816
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
821.                free (buf);
....
816.                __fxprintf (NULL, "%s", buf);
```

Use After Free\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=242
Status	New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	871	866
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
871.              free (buf);
....
866.              __fxprintf (NULL, "%s", buf);
```

Use After Free\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=243

Status	New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	921	916
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
921.                    free (buf);
....
916.                    __fxprintf (NULL, "%s", buf);
```

Use After Free\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=244
Status	New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	994	989
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
994.                    free (buf);
....
989.                    __fxprintf (NULL, "%s", buf);
```

Use After Free\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=245
Status	New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1036	1031
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1036.                         free (buf);
....
1031.                         __fxprintf (NULL, "%s", buf);
```

Use After Free\Path 9:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=246
Status New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1075	1070
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1075.                         free (buf);
....
1070.                         __fxprintf (NULL, "%s", buf);
```

Use After Free\Path 10:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=247
Status New

The pointer buf at vifm/getopt.c in line 400 is being used after it has been freed.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1149	1144
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
1149.                   free (buf);
....
1144.                   __fxprintf (NULL, "%s", buf);
```

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=1020015&projectid=20011&pathid=143
Status	New

The size of the buffer used by luaO_chunkid in srclen, at line 557 of vifm/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to srclen, at line 557 of vifm/lobject.c, to overwrite the target buffer.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	561	561
Object	srclen	srclen

Code Snippet
File Name vifm/lobject.c
Method void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
561.        memcpy(out, source + 1, srclen * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 2:

Severity	Medium
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=144
Status	New

The size of the buffer used by luaO_chunkid in char, at line 557 of vifm/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to char, at line 557 of vifm/lobject.c, to overwrite the target buffer.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	561	561
Object	char	char

Code Snippet
File Name vifm/lobject.c
Method void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
561.        memcpy(out, source + 1, srclen * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=145
Status	New

The size of the buffer used by luaO_chunkid in srclen, at line 557 of vifm/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to srclen, at line 557 of vifm/lobject.c, to overwrite the target buffer.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	569	569
Object	srclen	srclen

Code Snippet
File Name vifm/lobject.c
Method void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
569.        memcpy(out, source + 1, srclen * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=146

Status	New

The size of the buffer used by luaO_chunkid in char, at line 557 of vifm/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to char, at line 557 of vifm/lobject.c, to overwrite the target buffer.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	569	569
Object	char	char

Code Snippet
File Name vifm/lobject.c
Method void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
569.        memcpy(out, source + 1, srclen * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=147
Status	New

The size of the buffer used by luaO_chunkid in bufflen, at line 557 of vifm/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to bufflen, at line 557 of vifm/lobject.c, to overwrite the target buffer.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	573	573
Object	bufflen	bufflen

Code Snippet
File Name vifm/lobject.c
Method void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
573.        memcpy(out, source + 1 + srclen - bufflen, bufflen *
sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=148
Status	New

The size of the buffer used by luaO_chunkid in char, at line 557 of vifm/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to char, at line 557 of vifm/lobject.c, to overwrite the target buffer.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	573	573
Object	char	char

Code Snippet
File Name vifm/lobject.c
Method void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
573.        memcpy(out, source + 1 + srclen - bufflen, bufflen *
sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 7:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=149
Status New

The size of the buffer used by luaO_chunkid in char, at line 557 of vifm/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to char, at line 557 of vifm/lobject.c, to overwrite the target buffer.

	Source	Destination
File	vifm/lobject.c	vifm/lobject.c
Line	589	589
Object	char	char

Code Snippet
File Name vifm/lobject.c
Method void luaO_chunkid (char *out, const char *source, size_t srclen) {

```
....
589.      memcpy(out, POS, (LL(POS) + 1) * sizeof(char));
```

MemoryFree on StackVariable

Query Path:
CPP\Cx\CPP Medium Threat\MemoryFree on StackVariable Version:0

Description

MemoryFree on StackVariable\Path 1:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=150

Status	New

Calling free() (line 400) on a variable that was not dynamically allocated (line 400) in file vifm/getopt.c may result with a crash.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	625	625
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
625.                    free (buf);
```

MemoryFree on StackVariable\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=151
Status	New

Calling free() (line 400) on a variable that was not dynamically allocated (line 400) in file vifm/getopt.c may result with a crash.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	710	710
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
710.                    free (buf);
```

MemoryFree on StackVariable\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=152
Status	New

Calling free() (line 400) on a variable that was not dynamically allocated (line 400) in file vifm/getopt.c may result with a crash.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	821	821
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
821.                free (buf);
```

MemoryFree on StackVariable\Path 4:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=153
Status New

Calling free() (line 400) on a variable that was not dynamically allocated (line 400) in file vifm/getopt.c may result with a crash.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	871	871
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
871.                free (buf);
```

MemoryFree on StackVariable\Path 5:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=154
Status New

Calling free() (line 400) on a variable that was not dynamically allocated (line 400) in file vifm/getopt.c may result with a crash.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	921	921
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
921.                    free (buf);
```

MemoryFree on StackVariable\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=155
Status	New

Calling free() (line 400) on a variable that was not dynamically allocated (line 400) in file vifm/getopt.c may result with a crash.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	994	994
Object	buf	buf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
994.                    free (buf);
```

Environment Injection

Query Path:
CPP\Cx\CPP Medium Threat\Environment Injection Version:0

Categories

OWASP Top 10 2013: A1-Injection
FISMA 2014: System And Information Integrity
NIST SP 800-53: SI-10 Information Input Validation (P1)
OWASP Top 10 2017: A1-Injection

Description

Environment Injection\Path 1:

Severity	Medium
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=232
Status	New

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	291
Object	getenv	getenv

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

Environment Injection\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=233
Status	New

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	291
Object	argc	getenv

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

Environment Injection\Path 3:

Severity	Medium

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=234
Status	New

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	291
Object	argv	getenv

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

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=1020015&projectid=20011&pathid=235
Status	New

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	219	219
Object	new_str	new_str

Code Snippet
File Name vifm/getopt.c

Method	exchange (char **argv, struct _getopt_data *d)

```
....
219.        char *new_str = malloc (top + 1);
```

Memory Leak\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=236
Status	New

	Source	Destination
File	vifm/chmod.c	vifm/chmod.c
Line	170	170
Object	name	name

Code Snippet	
File Name	vifm/chmod.c
Method	alloc_file_list(view_t *view, const char filename[])

```
....
170.        view->dir_entry[0].name = strdup(filename);
```

Memory Leak\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=237
Status	New

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	325	325
Object	__getopt_nonoption_flags	__getopt_nonoption_flags

Code Snippet	
File Name	vifm/getopt.c
Method	_getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
325.              __getopt_nonoption_flags =
```

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=1020015&projectid=20011&pathid=110
Status	New

The TEARDOWN method calls the remove function, at line 67 of vifm/media.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	vifm/media.c	vifm/media.c
Line	69	69
Object	remove	remove

Code Snippet
File Name vifm/media.c
Method TEARDOWN()

```
....
69.    assert_success(remove("script"));
```

Unchecked Return Value\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=111
Status	New

The TEST method calls the remove function, at line 203 of vifm/media.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	vifm/media.c	vifm/media.c
Line	215	215
Object	remove	remove

Code Snippet
File Name vifm/media.c
Method TEST(enter_mounts_unmounted_device)

```
....
215.          (void)remove("out");
```

Unchecked Return Value\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=112
Status	New

The TEST method calls the remove function, at line 203 of vifm/media.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	vifm/media.c	vifm/media.c
Line	236	236
Object	remove	remove

Code Snippet
File Name vifm/media.c
Method TEST(enter_mounts_unmounted_device)

```
....
236.        assert_success(remove("out"));
```

Unchecked Return Value\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=113
Status	New

The TEST method calls the remove function, at line 271 of vifm/media.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	vifm/media.c	vifm/media.c
Line	296	296
Object	remove	remove

Code Snippet
File Name vifm/media.c
Method TEST(r_reloads_list)

```
....
296.        assert_success(remove("out"));
```

Unchecked Return Value\Path 5:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=114
Status	New

The TEST method calls the remove function, at line 299 of vifm/media.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	vifm/media.c	vifm/media.c
Line	320	320
Object	remove	remove

Code Snippet
File Name vifm/media.c
Method TEST(m_toggles_mounts)

```
....
320.         (void)remove("out");
```

Unchecked Return Value\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=115
Status	New

The TEST method calls the remove function, at line 299 of vifm/media.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	vifm/media.c	vifm/media.c
Line	377	377
Object	remove	remove

Code Snippet
File Name vifm/media.c
Method TEST(m_toggles_mounts)

```
....
377.         assert_success(remove("out"));
```

Unchecked Return Value\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=200

	11&pathid=116
Status	New

The TEST method calls the remove function, at line 380 of vifm/media.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	vifm/media.c	vifm/media.c
Line	407	407
Object	remove	remove

Code Snippet
File Name vifm/media.c
Method TEST(mounting_failure_is_handled)

```
....
407.        assert_success(remove("out"));
```

Unchecked Return Value\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=117
Status	New

The TEST method calls the remove function, at line 410 of vifm/media.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	vifm/media.c	vifm/media.c
Line	446	446
Object	remove	remove

Code Snippet
File Name vifm/media.c
Method TEST(mount_directory_is_left_before_unmounting)

```
....
446.        assert_success(remove("out"));
```

Unchecked Return Value\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=118
Status	New

The stic_assert_int_equal method calls the sprintf function, at line 296 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	299	299
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_int_equal(int expected, int actual, const char* function, const char file[], unsigned int line)

```
....
299.        sprintf(s, "Expected %d but was %d", expected, actual);
```

Unchecked Return Value\Path 10:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=119
Status New

The stic_assert_ulong_equal method calls the sprintf function, at line 303 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	306	306
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_ulong_equal(unsigned long expected, unsigned long actual, const char* function, const char file[], unsigned int line)

```
....
306.        sprintf(s, "Expected %lu but was %lu", expected, actual);
```

Unchecked Return Value\Path 11:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=120
Status New

The stic_assert_float_equal method calls the sprintf function, at line 310 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	314	314
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_float_equal(float expected, float actual, float delta, const char* function, const char file[], unsigned int line)

```
....
314.        sprintf(s, "Expected %f but was %f", expected, actual);
```

Unchecked Return Value\Path 12:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=121
Status New

The stic_assert_double_equal method calls the sprintf function, at line 319 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	323	323
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_double_equal(double expected, double actual, double delta, const char* function, const char file[], unsigned int line)

```
....
323.        sprintf(s, "Expected %f but was %f", expected, actual);
```

Unchecked Return Value\Path 13:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=122

Status	New

The stic_assert_string_equal method calls the sprintf function, at line 328 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	335	335
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
335.            sprintf(s, "Expected <NULL> but was <NULL>");
```

Unchecked Return Value\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=123
Status	New

The stic_assert_string_equal method calls the sprintf function, at line 328 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	340	340
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
340.          sprintf(s, "Expected <NULL> but was \"%s\"", actual);
```

Unchecked Return Value\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=200

	11&pathid=124
Status	New

The stic_assert_string_equal method calls the sprintf function, at line 328 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	345	345
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
345.          sprintf(s, "Expected \"%s\" but was <NULL>", expected);
```

Unchecked Return Value\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=125
Status	New

The stic_assert_string_equal method calls the sprintf function, at line 328 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	351	351
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
351.          sprintf(s, "Expected \"%s\" but was \"%s\"", expected,
actual);
```

Unchecked Return Value\Path 17:

Severity	Low
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=126
Status	New

The stic_assert_wstring_equal method calls the sprintf function, at line 357 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	364	364
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_wstring_equal(const wchar_t expected[], const wchar_t actual[], const char function[], const char file[], unsigned int line)

```
....
364.              sprintf(s, "Expected <NULL> but was <NULL>");
```

Unchecked Return Value\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=127
Status	New

The stic_assert_wstring_equal method calls the sprintf function, at line 357 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	370	370
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_wstring_equal(const wchar_t expected[], const wchar_t actual[], const char function[], const char file[], unsigned int line)

```
....
370.              sprintf(s, "Expected <NULL> but was \"%ls\"", actual);
```

Unchecked Return Value\Path 19:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=128
Status	New

The stic_assert_wstring_equal method calls the sprintf function, at line 357 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	379	379
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_wstring_equal(const wchar_t expected[], const wchar_t actual[], const char function[], const char file[], unsigned int line)

```
....
379.              sprintf(s, "Expected \"%ls\" but was <NULL>",
expected);
```

Unchecked Return Value\Path 20:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=129
Status	New

The stic_assert_wstring_equal method calls the sprintf function, at line 357 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	389	389
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_wstring_equal(const wchar_t expected[], const wchar_t actual[], const char function[], const char file[], unsigned int line)

```
....
389.              sprintf(s, "Expected \"%ls\" but was \"%ls\"",
expected, actual);
```

Unchecked Return Value\Path 21:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=130
Status	New

The stic_assert_string_ends_with method calls the sprintf function, at line 398 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	401	401
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_ends_with(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
401.         sprintf(s, "Expected \"%s\" to end with \"%s\"", actual,
expected);
```

Unchecked Return Value\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=131
Status	New

The stic_assert_string_starts_with method calls the sprintf function, at line 405 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	408	408
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_starts_with(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
408.        sprintf(s, "Expected \"%s\" to start with \"%s\"", actual,
expected);
```

Unchecked Return Value\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=132
Status	New

The stic_assert_string_contains method calls the sprintf function, at line 412 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	415	415
Object	sprintf	sprintf

Code Snippet

File Name vifm/stic.c

Method void stic_assert_string_contains(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
415.        sprintf(s, "Expected \"%s\" to be in \"%s\"", expected,
actual);
```

Unchecked Return Value\Path 24:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=133
Status	New

The stic_assert_string_doesnt_contain method calls the sprintf function, at line 419 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	422	422
Object	sprintf	sprintf

Code Snippet

File Name	vifm/stic.c
Method	void stic_assert_string_doesnt_contain(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
422.        sprintf(s, "Expected \"%s\" not to have \"%s\" in it",
actual, expected);
```

Unchecked Return Value\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=134
Status	New

The stic_test_fixture_end method calls the sprintf function, at line 447 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	473	473
Object	sprintf	sprintf

Code Snippet	
File Name	vifm/stic.c
Method	void stic_test_fixture_end()

```
....
473.        sprintf(s, "%d test%s run  %d check%s failed", nrun, nrun ==
1 ? "" : "s",
```

Unchecked Return Value\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=135
Status	New

The run_tests method calls the snprintf function, at line 525 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	541	541
Object	snprintf	snprintf

Code Snippet
File Name vifm/stic.c
Method int run_tests(stic_void_void tests)

```
....
541.        snprintf(version, sizeof(version), "stic v%s%s%s",
STIC_VERSION,
```

Unchecked Return Value\Path 27:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=136
Status New

The run_tests method calls the snprintf function, at line 525 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	547	547
Object	snprintf	snprintf

Code Snippet
File Name vifm/stic.c
Method int run_tests(stic_void_void tests)

```
....
547.              snprintf(s, sizeof(s), "%d CHECK%s IN %d TEST%s
FAILED",
```

Unchecked Return Value\Path 28:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=137
Status New

The run_tests method calls the snprintf function, at line 525 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	554	554
Object	snprintf	snprintf

Code Snippet

File Name	vifm/stic.c
Method	int run_tests(stic_void_void tests)

```
....
554.              snprintf(s, sizeof(s), "ALL TESTS PASSED");
```

Unchecked Return Value\Path 29:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=138
Status	New

The run_tests method calls the sprintf function, at line 525 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	563	563
Object	sprintf	sprintf

Code Snippet	
File Name	vifm/stic.c
Method	int run_tests(stic_void_void tests)

```
....
563.              sprintf(time,"< 1 ms");
```

Unchecked Return Value\Path 30:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=139
Status	New

The run_tests method calls the sprintf function, at line 525 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	567	567
Object	sprintf	sprintf

Code Snippet	
File Name	vifm/stic.c
Method	int run_tests(stic_void_void tests)

```
....
567.              sprintf(time,"%lu ms",end - start);
```

Unchecked Return Value\Path 31:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=140
Status	New

The run_tests method calls the sprintf function, at line 525 of vifm/stic.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	vifm/stic.c	vifm/stic.c
Line	570	570
Object	sprintf	sprintf

Code Snippet
File Name vifm/stic.c
Method int run_tests(stic_void_void tests)

```
....
570.          sprintf(s,"%d check%s :: %d run test%s :: %d skipped test%s
:: %s",
```

Unchecked Return Value\Path 32:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=141
Status	New

The alloc_file_list method calls the name function, at line 164 of vifm/chmod.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	vifm/chmod.c	vifm/chmod.c
Line	170	170
Object	name	name

Code Snippet
File Name vifm/chmod.c
Method alloc_file_list(view_t *view, const char filename[])

```
....
170.        view->dir_entry[0].name = strdup(filename);
```

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=1020015&projectid=20011&pathid=297
Status	New

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	600	600
Object	fprintf	fprintf

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
600.                fprintf (fp,
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=298
Status	New

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	606	606
Object	fprintf	fprintf

Code Snippet
File Name vifm/getopt.c

Method	_getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
606.                    fprintf (fp, " '--%s'", ambig_list->p->name);
```

Improper Resource Access Authorization\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=299
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	175	175
Object	fprintf	fprintf

Code Snippet
File Name vifm/media.c
Method TEST(enter_navigates_to_mount_point)

```
....
175.        fprintf(fp, SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=300
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	191	191
Object	fprintf	fprintf

Code Snippet
File Name vifm/media.c
Method TEST(enter_navigates_to_mount_point_on_device_line)

```
....
191.        fprintf(fp, SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 5:

Severity	Low
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=301
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	259	259
Object	fprintf	fprintf

Code Snippet
File Name vifm/media.c
Method TEST(unhandled_key_is_ignored)

```
....
259.        fprintf(fp, SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=302
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	274	274
Object	fprintf	fprintf

Code Snippet
File Name vifm/media.c
Method TEST(r_reloads_list)

```
....
274.        fprintf(fp, SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=303
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c

Line	413	413
Object	fprintf	fprintf

Code Snippet
File Name vifm/media.c
Method TEST(mount_directory_is_left_before_unmounting)

```
....
413.        fprintf(fp, "#!/bin/sh\n"
```

Improper Resource Access Authorization\Path 8:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=304
Status New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	452	452
Object	fprintf	fprintf

Code Snippet
File Name vifm/media.c
Method TEST(mount_matching_current_path_is_picked_by_default)

```
....
452.        fprintf(fp, SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 9:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=305
Status New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	470	470
Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(barckets_navigates_between_devices)

```
....
470.        fputs(SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=306
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	86	86
Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(menu_not_created_if_no_devices)

```
....
86.    fputs(SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=307
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	113	113
Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(menu_is_loaded)

```
....
113.        fputs(SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 12:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=308
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	126	126
Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(entries_are_formatted_correctly)

```
....
126.         fputs(SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=309
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	206	206
Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(enter_mounts_unmounted_device)

```
....
206.         fputs("#!/bin/sh\n"
```

Improper Resource Access Authorization\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=310
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	242	242

Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(enter_does_nothing_on_device_lines_with_multiple_mounts)

```
....
242.        fputs(SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 15:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=311
Status New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	283	283
Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(r_reloads_list)

```
....
283.        fputs("#!/bin/sh\n"
```

Improper Resource Access Authorization\Path 16:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=312
Status New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	302	302
Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(m_toggles_mounts)

```
....
302.        fputs("#!/bin/sh\n"
```

Improper Resource Access Authorization\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=313
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	383	383
Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(mounting_failure_is_handled)

```
....
383.        fputs(SHEBANG_ECHO
```

Improper Resource Access Authorization\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=314
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	391	391
Object	fputs	fputs

Code Snippet
File Name vifm/media.c
Method TEST(mounting_failure_is_handled)

```
....
391.        fputs("#!/bin/sh\n"
```

Incorrect Permission Assignment For Critical Resources

Query Path:
CPP\Cx\CPP Low Visibility\Incorrect Permission Assignment For Critical Resources Version:1

Categories

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

Description

Incorrect Permission Assignment For Critical Resources\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=315
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	282	282
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(r_reloads_list)

```
....
282.        fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=316
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	390	390
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(mounting_failure_is_handled)

```
....
390.        fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 3:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=317
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	469	469
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(barckets_navigates_between_devices)

```
....
469.         FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=318
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	85	85
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(menu_not_created_if_no_devices)

```
....
85.    FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=319
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c

Line	112	112
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(menu_is_loaded)

```
....
112.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 6:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=320
Status New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	125	125
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(entries_are_formatted_correctly)

```
....
125.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 7:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=321
Status New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	174	174
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(enter_navigates_to_mount_point)

```
....
174.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=322
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	190	190
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(enter_navigates_to_mount_point_on_device_line)

```
....
190.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=323
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	205	205
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(enter_mounts_unmounted_device)

```
....
205.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 10:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=324
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	241	241
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(enter_does_nothing_on_device_lines_with_multiple_mounts)

```
....
241.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=325
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	258	258
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(unhandled_key_is_ignored)

```
....
258.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=326
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	273	273

Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(r_reloads_list)

```
....
273.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 13:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=327
Status New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	301	301
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(m_toggles_mounts)

```
....
301.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 14:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=328
Status New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	382	382
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(mounting_failure_is_handled)

```
....
382.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=329
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	412	412
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(mount_directory_is_left_before_unmounting)

```
....
412.        FILE *fp = fopen("script", "w");
```

Incorrect Permission Assignment For Critical Resources\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=330
Status	New

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	451	451
Object	fp	fp

Code Snippet
File Name vifm/media.c
Method TEST(mount_matching_current_path_is_picked_by_default)

```
....
451.        FILE *fp = fopen("script", "w");
```

TOCTOU

Query Path:
CPP\Cx\CPP Low Visibility\TOCTOU Version:1
Description

TOCTOU\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=331
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	469	469
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(barckets_navigates_between_devices)

```
....
469.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=332
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	85	85
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(menu_not_created_if_no_devices)

```
....
85.    FILE *fp = fopen("script", "w");
```

TOCTOU\Path 3:

Severity	Low
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=333
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	112	112
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(menu_is_loaded)

```
....
112.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=334
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	125	125
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(entries_are_formatted_correctly)

```
....
125.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=335

Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	174	174
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(enter_navigates_to_mount_point)

```
....
174.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=336
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	190	190
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(enter_navigates_to_mount_point_on_device_line)

```
....
190.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=337
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	205	205
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(enter_mounts_unmounted_device)

```
....
205.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 8:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=338
Status New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	241	241
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(enter_does_nothing_on_device_lines_with_multiple_mounts)

```
....
241.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 9:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=339
Status New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	258	258
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(unhandled_key_is_ignored)

```
....
258.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 10:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=340
Status New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	273	273
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(r_reloads_list)

```
....
273.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 11:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=341
Status New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c

Line	282	282
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(r_reloads_list)

```
....
282.        fp = fopen("script", "w");
```

TOCTOU\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=342
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	301	301
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(m_toggles_mounts)

```
....
301.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=343
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	382	382
Object	fopen	fopen

Code Snippet	
File Name	vifm/media.c
Method	TEST(mounting_failure_is_handled)

```
....
382.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=344
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	390	390
Object	fopen	fopen

Code Snippet	
File Name	vifm/media.c
Method	TEST(mounting_failure_is_handled)

```
....
390.        fp = fopen("script", "w");
```

TOCTOU\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=345
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	412	412
Object	fopen	fopen

Code Snippet	
File Name	vifm/media.c

Method	TEST(mount_directory_is_left_before_unmounting)

```
....
412.        FILE *fp = fopen("script", "w");
```

TOCTOU\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=346
Status	New

The TEST method in vifm/media.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	vifm/media.c	vifm/media.c
Line	451	451
Object	fopen	fopen

Code Snippet
File Name vifm/media.c
Method TEST(mount_matching_current_path_is_picked_by_default)

```
....
451.        FILE *fp = fopen("script", "w");
```

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=1020015&projectid=20011&pathid=158
Status	New

The size of the buffer used by stic_assert_string_equal in "Expected but was \"%s\"", at line 328 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that stic_assert_string_equal passes to "Expected but was \"%s\"", at line 328 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination

File	vifm/stic.c	vifm/stic.c
Line	340	340
Object	"Expected but was \"%s\""	"Expected but was \"%s\""

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
340.          sprintf(s, "Expected <NULL> but was \"%s\"", actual);
```

Potential Precision Problem\Path 2:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=159
Status New

The size of the buffer used by stic_assert_string_equal in "Expected \"%s\" but was ", at line 328 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that stic_assert_string_equal passes to "Expected \"%s\" but was ", at line 328 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	345	345
Object	"Expected \"%s\" but was "	"Expected \"%s\" but was "

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
345.          sprintf(s, "Expected \"%s\" but was <NULL>", expected);
```

Potential Precision Problem\Path 3:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=160
Status New

The size of the buffer used by stic_assert_string_equal in "Expected \"%s\" but was \"%s\"", at line 328 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that stic_assert_string_equal passes to "Expected \"%s\" but was \"%s\"", at line 328 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	351	351
Object	"Expected \"%s\" but was \"%s\""	"Expected \"%s\" but was \"%s\""

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_equal(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
351.          sprintf(s, "Expected \"%s\" but was \"%s\"", expected,
actual);
```

Potential Precision Problem\Path 4:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=161
Status New

The size of the buffer used by stic_assert_string_ends_with in "Expected \"%s\" to end with \"%s\"", at line 398 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that stic_assert_string_ends_with passes to "Expected \"%s\" to end with \"%s\"", at line 398 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	401	401
Object	"Expected \"%s\" to end with \"%s\""	"Expected \"%s\" to end with \"%s\""

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_ends_with(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
401.          sprintf(s, "Expected \"%s\" to end with \"%s\"", actual,
expected);
```

Potential Precision Problem\Path 5:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=162
Status New

The size of the buffer used by stic_assert_string_starts_with in "Expected \"%s\" to start with \"%s\"", at line 405 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that stic_assert_string_starts_with passes to "Expected \"%s\" to start with \"%s\"", at line 405 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	408	408
Object	"Expected \"%s\" to start with \"%s\""	"Expected \"%s\" to start with \"%s\""

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_starts_with(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
408.        sprintf(s, "Expected \"%s\" to start with \"%s\"", actual,
expected);
```

Potential Precision Problem\Path 6:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=163
Status New

The size of the buffer used by stic_assert_string_contains in "Expected \"%s\" to be in \"%s\"", at line 412 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that stic_assert_string_contains passes to "Expected \"%s\" to be in \"%s\"", at line 412 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	415	415
Object	"Expected \"%s\" to be in \"%s\""	"Expected \"%s\" to be in \"%s\""

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_contains(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
415.        sprintf(s, "Expected \"%s\" to be in \"%s\"", expected,
actual);
```

Potential Precision Problem\Path 7:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=200

	11&pathid=164
Status	New

The size of the buffer used by stic_assert_string_doesnt_contain in "Expected \"%s\" not to have \"%s\" in it", at line 419 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that stic_assert_string_doesnt_contain passes to "Expected \"%s\" not to have \"%s\" in it", at line 419 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	422	422
Object	"Expected \"%s\" not to have \"%s\" in it"	"Expected \"%s\" not to have \"%s\" in it"

Code Snippet
File Name vifm/stic.c
Method void stic_assert_string_doesnt_contain(const char* expected, const char* actual, const char* function, const char file[], unsigned int line)

```
....
422.        sprintf(s, "Expected \"%s\" not to have \"%s\" in it",
actual, expected);
```

Potential Precision Problem\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=165
Status	New

The size of the buffer used by stic_test_fixture_end in "%d test%s run %d check%s failed", at line 447 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that stic_test_fixture_end passes to "%d test%s run %d check%s failed", at line 447 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	473	473
Object	"%d test%s run %d check%s failed"	"%d test%s run %d check%s failed"

Code Snippet
File Name vifm/stic.c
Method void stic_test_fixture_end()

```
....
473.        sprintf(s, "%d test%s run  %d check%s failed", nrun, nrun ==
1 ? "" : "s",
```

Potential Precision Problem\Path 9:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=166
Status	New

The size of the buffer used by run_tests in "%d check%s :: %d run test%s :: %d skipped test%s :: %s", at line 525 of vifm/stic.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that run_tests passes to "%d check%s :: %d run test%s :: %d skipped test%s :: %s", at line 525 of vifm/stic.c, to overwrite the target buffer.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	570	570
Object	"%d check%s :: %d run test%s :: %d skipped test%s :: %s"	"%d check%s :: %d run test%s :: %d skipped test%s :: %s"

Code Snippet
File Name vifm/stic.c
Method int run_tests(stic_void_void tests)

```
....
570.        sprintf(s,"%d check%s :: %d run test%s :: %d skipped test%s
:: %s",
```

Heuristic Buffer Overflow malloc

Query Path:
CPP\Cx\CPP Heuristic\Heuristic Buffer Overflow malloc 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 Buffer Overflow malloc\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=167
Status	New

The size of the buffer used by exchange in top, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	219
Object	argc	top

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
219.        char *new_str = malloc (top + 1);
```

Heuristic Buffer Overflow malloc\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=168
Status	New

The size of the buffer used by exchange in top, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	219
Object	argv	top

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
219.        char *new_str = malloc (top + 1);
```

Heuristic Buffer Overflow malloc\Path 3:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=169
Status	New

The size of the buffer used by exchange in top, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	219
Object	getenv	top

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
219.        char *new_str = malloc (top + 1);
```

Heuristic Buffer Overflow malloc\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=170
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	219
Object	argc	BinaryExpr

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
219.        char *new_str = malloc (top + 1);
```

Heuristic Buffer Overflow malloc\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=171
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.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 argv, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	219
Object	argv	BinaryExpr

Code Snippet

File Name vifm/getopt.c

Method main (int argc, char **argv)

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method exchange (char **argv, struct _getopt_data *d)

```
....
219.        char *new_str = malloc (top + 1);
```

Heuristic Buffer Overflow malloc\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=172
Status	New

The size of the buffer used by exchange in BinaryExpr, at line 199 of vifm/getopt.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _getopt_initialize passes to getenv, at line 280 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	291	219
Object	getenv	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
291.    d->__posixly_correct = posixly_correct | !!getenv
("POSIXLY_CORRECT");
```

File Name vifm/getopt.c
Method exchange (char **argv, struct _getopt_data *d)

```
....
219.        char *new_str = malloc (top + 1);
```

Heuristic Buffer Overflow malloc\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=173
Status	New

The size of the buffer used by _getopt_initialize in __nonoption_flags_max_len, at line 280 of vifm/getopt.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 argc, at line 1223 of vifm/getopt.c, to overwrite the target buffer.

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1223	326
Object	argc	__nonoption_flags_max_len

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

```
....
1223.  main (int argc, char **argv)
```

File Name vifm/getopt.c

Method	_getopt_initialize (int argc, char *const *argv, const char *optstring,

```
....
326.              (char *) malloc (d->__nonoption_flags_max_len);
```

Inconsistent Implementations

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

Description

Inconsistent Implementations\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=107
Status	New

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	1232	1232
Object	getopt	getopt

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

```
....
1232.         c = getopt (argc, argv, "abc:d:0123456789");
```

Inconsistent Implementations\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=108
Status	New

	Source	Destination
File	vifm/getopt1.c	vifm/getopt1.c
Line	125	125
Object	getopt_long	getopt_long

Code Snippet	
File Name	vifm/getopt1.c
Method	main (int argc, char **argv)

```
....
125.        c = getopt_long (argc, argv, "abc:d:0123456789",
```

Use of Insufficiently Random Values

Query Path:
CPP\Cx\CPP Low Visibility\Use of Insufficiently Random Values Version:0

Categories

FISMA 2014: Media Protection
NIST SP 800-53: SC-28 Protection of Information at Rest (P1)
OWASP Top 10 2017: A3-Sensitive Data Exposure

Description

Use of Insufficiently Random Values\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020015&projectid=20011&pathid=109
Status	New

Method stic_simple_test_result_log at line 192 of vifm/stic.c uses a weak method rand to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	vifm/stic.c	vifm/stic.c
Line	204	204
Object	rand	rand

Code Snippet
File Name vifm/stic.c
Method void stic_simple_test_result_log(int passed, char* reason, const char* function, const char file[], unsigned int line)

```
....
204.        if (stic_random_failures && rand() % 8 == 0)
```

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=1020015&projectid=20011&pathid=142
Status	New

Source	Destination

File	vifm/lobject.c	vifm/lobject.c
Line	508	508
Object	sizeof	sizeof

Code Snippet
File Name vifm/lobject.c
Method const char *luaO_pushvfstring (lua_State *L, const char *fmt, va_list argp) {

```
....
508.          const int sz = 3 * sizeof(void*) + 8; /* enough space for
'%p' */
```

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=1020015&projectid=20011&pathid=157
Status	New

The variable declared in 0 at vifm/lgc.c in line 125 is not initialized when it is used by g at vifm/lgc.c in line 652.

	Source	Destination
File	vifm/lgc.c	vifm/lgc.c
Line	137	655
Object	0	g

Code Snippet
File Name vifm/lgc.c
Method static GCObject **getgclist (GCObject *o) {

```
....
137.      default: lua_assert(0); return 0;
```

File Name vifm/lgc.c
Method static lu_mem propagatemark (global_State *g) {

```
....
655.    g->gray = *getgclist(o);  /* remove from 'gray' list */
```

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=1020015&projectid=20011&pathid=174
Status	New

	Source	Destination
File	vifm/getopt.c	vifm/getopt.c
Line	512	512
Object	BinaryExpr	BinaryExpr

Code Snippet
File Name vifm/getopt.c
Method _getopt_internal_r (int argc, char *const *argv, const char *optstring,

```
....
512.                + (longopts != NULL && argv[d->optind][1] == '-'));
```

Buffer Overflow boundedcpy

Risk

What might happen

Allowing tainted inputs to set the size of how many bytes to copy from source to destination may cause memory corruption, unexpected behavior, instability and data leakage. In some cases, such as when additional and specific areas of memory are also controlled by user input, it may result in code execution.

Cause

How does it happen

Should the size of the amount of bytes to copy from source to destination be greater than the size of the destination, an overflow will occur, and memory beyond the intended buffer will get overwritten. Since this size value is derived from user input, the user may provide an invalid and dangerous buffer size.

General Recommendations

How to avoid it

- Do not trust memory allocation sizes provided by the user; derive them from the copied values instead.
- If memory allocation by a provided value is absolutely required, restrict this size to safe values only. Specifically ensure that this value does not exceed the destination buffer's size.

Source Code Examples

CPP

Size Parameter is Influenced by User Input

```
char dest_buf[10];
memset(dest_buf, '\0', sizeof(dest_buf));
strncpy(dest_buf, src_buf, size); //Assuming size is provided by user input
```

Validating Destination Buffer Length

```
char dest_buf[10];
memset(dest_buf, '\0', sizeof(dest_buf));
if (size < sizeof(dest_buf) && sizeof(src_buf) >= size) //Assuming size is provided by user
input
{
      strncpy(dest_buf, src_buf, size);
}
else
{
      //...
}
```

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 StrcpyStrcat

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 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

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));
    }
}
```

MemoryFree on StackVariable

Risk

What might happen

Undefined Behavior may result with a crash. Crashes may give an attacker valuable information about the system and the program internals. Furthermore, it may leave unprotected files (e.g memory) that may be exploited.

Cause

How does it happen

Calling free() on a variable that was not dynamically allocated (e.g. malloc) will result with an Undefined Behavior.

General Recommendations

How to avoid it

Use free() only on dynamically allocated variables in order to prevent unexpected behavior from the compiler.

Source Code Examples

CPP

Bad - Calling free() on a static variable

```
void clean_up(){
  char temp[256];
  do_something();
  free(tmp);
  return;
}
```

Good - Calling free() only on variables that were dynamically allocated

```
void clean_up(){
  char *buff;
  buff = (char*) malloc(1024);
  free(buff);
  return;
}
```

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;
}
```

Improper Sanitization of Special Elements used in a Command ('Command Injection')

Weakness ID: 77 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software constructs all or part of a command using externally-influenced input from an upstream component, but it does not sanitize or incorrectly sanitizes special elements that could modify the intended command when it is sent to a downstream component.

Extended Description

Command injection vulnerabilities typically occur when:

1. Data enters the application from an untrusted source.
2. The data is part of a string that is executed as a command by the application.
3. By executing the command, the application gives an attacker a privilege or capability that the attacker would not otherwise have.

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

All

Common Consequences

Scope	Effect
Access Control	Command injection allows for the execution of arbitrary commands and code by the attacker.
Integrity	If a malicious user injects a character (such as a semi-colon) that delimits the end of one command and the beginning of another, it may be possible to then insert an entirely new and unrelated command that was not intended to be executed.

Likelihood of Exploit

Very High

Demonstrative Examples

Example 1

The following simple program accepts a filename as a command line argument and displays the contents of the file back to the user. The program is installed setuid root because it is intended for use as a learning tool to allow system administrators in-training to inspect privileged system files without giving them the ability to modify them or damage the system.

Example Language: **C**

```
int main(char* argc, char** argv) {
char cmd[CMD_MAX] = "/usr/bin/cat ";
strcat(cmd, argv[1]);
system(cmd);
}
```

Because the program runs with root privileges, the call to system() also executes with root privileges. If a user specifies a standard filename, the call works as expected. However, if an attacker passes a string of the form ";rm -rf /", then the call to system() fails to execute cat due to a lack of arguments and then plows on to recursively delete the contents of the root partition.

Example 2

The following code is from an administrative web application designed to allow users to kick off a backup of an Oracle database using a batch-file wrapper around the rman utility and then run a cleanup.bat script to delete some temporary files. The script rmanDB.bat accepts a single command line parameter, which specifies what type of backup to perform. Because access to the database is restricted, the application runs the backup as a privileged user.

(Bad Code)

Example Language: **Java**

```
...
String btype = request.getParameter("backuptype");
String cmd = new String("cmd.exe /K \"
c:\\util\\rmanDB.bat "
+btype+
"&&c:\\utl\\cleanup.bat\"")
System.Runtime.getRuntime().exec(cmd);
...
```

The problem here is that the program does not do any validation on the backuptype parameter read from the user. Typically the Runtime.exec() function will not execute multiple commands, but in this case the program first runs the cmd.exe shell in order to run multiple commands with a single call to Runtime.exec(). Once the shell is invoked, it will happily execute multiple commands separated by two ampersands. If an attacker passes a string of the form "& del c:\\dbms*.*", then the application will execute this command along with the others specified by the program. Because of the nature of the application, it runs with the privileges necessary to interact with the database, which means whatever command the attacker injects will run with those privileges as well.

Example 3

The following code from a system utility uses the system property APPHOME to determine the directory in which it is installed and then executes an initialization script based on a relative path from the specified directory.

(Bad Code)

Example Language: **Java**

```
...
String home = System.getProperty("APPHOME");
String cmd = home + INITCMD;
java.lang.Runtime.getRuntime().exec(cmd);
...
```

The code above allows an attacker to execute arbitrary commands with the elevated privilege of the application by modifying the system property APPHOME to point to a different path containing a malicious version of INITCMD. Because the program does not validate the value read from the environment, if an attacker can control the value of the system property APPHOME, then they can fool the application into running malicious code and take control of the system.

Example 4

The following code is from a web application that allows users access to an interface through which they can update their password on the system. Part of the process for updating passwords in certain network environments is to run a make command in the /var/yp directory, the code for which is shown below.

(Bad Code)

Example Language: **Java**

```
...
System.Runtime.getRuntime().exec("make");
...
```

The problem here is that the program does not specify an absolute path for make and

fails to clean its environment prior to executing the call to Runtime.exec(). If an attacker can modify the $PATH variable to point to a malicious binary called make and cause the program to be executed in their environment, then the malicious binary will be loaded instead of the one intended. Because of the nature of the application, it runs with the privileges necessary to perform system operations, which means the attacker's make will now be run with these privileges, possibly giving the attacker complete control of the system.

Example 5

The following code is a wrapper around the UNIX command cat which prints the contents of a file to standard out. It is also injectable:

(Bad Code)

Example Language: **C**

```
#include <stdio.h>
#include <unistd.h>

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

char cat[] = "cat ";
char *command;
size_t commandLength;

commandLength = strlen(cat) + strlen(argv[1]) + 1;
command = (char *) malloc(commandLength);
strncpy(command, cat, commandLength);
strncat(command, argv[1], (commandLength - strlen(cat)) );

system(command);
return (0);
}
```

Used normally, the output is simply the contents of the file requested:

```
$ ./catWrapper Story.txt
When last we left our heroes...
```

However, if we add a semicolon and another command to the end of this line, the command is executed by catWrapper with no complaint:

(Attack)

```
$ ./catWrapper Story.txt; ls
When last we left our heroes...
Story.txt
SensitiveFile.txt
PrivateData.db
a.out*
```

If catWrapper had been set to have a higher privilege level than the standard user, arbitrary commands could be executed with that higher privilege.

Potential Mitigations

Phase: Architecture and Design

If at all possible, use library calls rather than external processes to recreate the desired functionality

Phase: Implementation

If possible, ensure that all external commands called from the program are statically created.

Phase: Implementation

Strategy: Input Validation

Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright.

When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue."

Run time: Run time policy enforcement may be used in a white-list fashion to prevent use of any non-sanctioned commands.

Assign permissions to the software system that prevents the user from accessing/opening privileged files.

Other Notes

Command injection is a common problem with wrapper programs.

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	Weakness Class	20	Improper Input Validation	**Seven Pernicious Kingdoms (primary)700**
ChildOf	Weakness Class	74	Failure to Sanitize Data into a Different Plane ('Injection')	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	713	OWASP Top Ten 2007 Category A2 - Injection Flaws	**Weaknesses in OWASP Top Ten (2007) (primary)629**
ChildOf	Category	722	OWASP Top Ten 2004 Category A1 - Unvalidated Input	Weaknesses in OWASP Top Ten (2004)711
ChildOf	Category	727	OWASP Top Ten 2004 Category A6 - Injection Flaws	**Weaknesses in OWASP Top Ten (2004) (primary)711**
ParentOf	Weakness Base	78	Improper Sanitization of Special Elements used in an OS Command ('OS Command Injection')	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	88	Argument Injection or Modification	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	89	Improper Sanitization of Special Elements used in an SQL Command ('SQL Injection')	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	90	Failure to Sanitize Data into LDAP Queries ('LDAP Injection')	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	624	Executable Regular Expression Error	**Development Concepts (primary)699 Research Concepts (primary)1000**

f Causal Nature

Explicit

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
7 Pernicious Kingdoms			Command Injection
CLASP			Command injection

OWASP Top Ten 2007	A2	CWE More Specific	Injection Flaws
OWASP Top Ten 2004	A1	CWE More Specific	Unvalidated Input
OWASP Top Ten 2004	A6	CWE More Specific	Injection Flaws

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
15	Command Delimiters	
23	File System Function Injection, Content Based	
43	Exploiting Multiple Input Interpretation Layers	
75	Manipulating Writeable Configuration Files	
6	Argument Injection	
11	Cause Web Server Misclassification	
76	Manipulating Input to File System Calls	

References

G. Hoglund and G. McGraw. "Exploiting Software: How to Break Code". Addison-Wesley. February 2004.

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 Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples, Name		
2009-07-27	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples, Description, Name		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Common Consequences, Description, Other Notes, Potential Mitigations		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Potential Mitigations, Relationships		

Previous Entry Names

Change Date	Previous Entry Name
2008-04-11	Command Injection
2009-05-27	Failure to Sanitize Data into a Control Plane (aka 'Command Injection')
2009-07-27	Failure to Sanitize Data into a Control Plane ('Command Injection')

BACK TO TOP

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

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

Description

Description Summary

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

Extended Description

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

Terminology Notes

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

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C

C++

Modes of Introduction

Memory leaks have two common and sometimes overlapping causes:

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

Common Consequences

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

Likelihood of Exploit

Medium

Demonstrative Examples

Example 1

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

(Bad Code)

Example Language: **C**

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

return NULL;
}
```

```
return buf;
}
```

Example 2

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

(Bad Code)

Example Language: **C**

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

free(foo);
}
int main() {

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

Observed Examples

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

Potential Mitigations

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

Phase: Architecture and Design

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

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

Relationships

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

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

Relationship Notes

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

Affected Resources

- Memory

Functional Areas

- Memory management

Taxonomy Mappings

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

White Box Definitions

A weakness where the code path has:

1. start statement that allocates dynamically allocated memory resource

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

Where "loses" is defined through the following scenarios:

1. identity of the dynamic allocated memory resource never obtained

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

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

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

References

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

Content History

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

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

Previous Entry Names

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

BACK TO TOP

Use After Free

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

---

Source Code Examples

CPP

Use of Variable after It was Freed

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

Use of Pointer to Local Variable That Was Freed On Return

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

void func2()

{

	int j;
	j = 5;
```

```
}

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

Use of Zero Initialized Pointer

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

Source Code Examples

CPP

Explicit NULL Dereference

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

Implicit NULL Dereference

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

Java

Explicit Null Dereference

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

Use of Function with Inconsistent Implementations

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

Description

Description Summary

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

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C: *(Often)*

PHP: *(Often)*

All

Potential Mitigations

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

Other Notes

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

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

Relationships

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

Taxonomy Mappings

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

Content History

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

BACK TO TOP

Use of Insufficiently Random Values

Risk

What might happen

Random values are often used as a mechanism to prevent malicious users from guessing a value, such as a password, encryption key, or session identifier. Depending on what this random value is used for, an attacker would be able to predict the next numbers generated, or previously generated values. This could enable the attacker to hijack another user's session, impersonate another user, or crack an encryption key (depending on what the pseudo-random value was used for).

Cause

How does it happen

The application uses a weak method of generating pseudo-random values, such that other numbers could be determined from a relatively small sample size. Since the pseudo-random number generator used is designed for statistically uniform distribution of values, it is approximately deterministic. Thus, after collecting a few generated values (e.g. by creating a few individual sessions, and collecting the sessionids), it would be possible for an attacker to calculate another sessionid.
Specifically, if this pseudo-random value is used in any security context, such as passwords, keys, or secret identifiers, an attacker would be able to predict the next numbers generated, or previously generated values.

General Recommendations

How to avoid it

Generic Guidance:

- Whenever unpredicatable numbers are required in a security context, use a cryptographically strong random number generator, instead of a statistical pseudo-random generator.
- Use the cryptorandom generator that is built-in to your language or platform, and ensure it is securely seeded. Do not seed the generator with a weak, non-random seed. (In most cases, the default is securely random).
- Ensure you use a long enough random value, to make brute-force attacks unfeasible.

Specific Recommendations:

- Do not use the statistical pseudo-random number generator, use the cryptorandom generator instead. In Java, this is the SecureRandom class.

Source Code Examples

Java

Use of a weak pseudo-random number generator

```
Random random = new Random();

long sessNum = random.nextLong();

String sessionId = sessNum.toString();
```

```
```

Cryptographically secure random number generator

```
SecureRandom random = new SecureRandom();

byte sessBytes[] = new byte[32];

random.nextBytes(sessBytes);

String sessionId = new String(sessBytes);
```

Objc

Use of a weak pseudo-random number generator

```
long sessNum = rand();
NSString* sessionId = [NSString stringWithFormat:@"%ld", sessNum];
```

Cryptographically secure random number generator

```
UInt32 sessBytes;
SecRandomCopyBytes(kSecRandomDefault, sizeof(sessBytes), (uint8_t*)&sessBytes);

NSString* sessionId = [NSString stringWithFormat:@"%llu", sessBytes];
```

Swift

Use of a weak pseudo-random number generator

```
let sessNum = rand();
let sessionId = String(format:"%ld", sessNum)
```

Cryptographically secure random number generator

```
var sessBytes: UInt32 = 0
withUnsafeMutablePointer(&sessBytes, { (sessBytesPointer) -> Void in
    let castedPointer = unsafeBitCast(sessBytesPointer, UnsafeMutablePointer<UInt8>.self)
    SecRandomCopyBytes(kSecRandomDefault, sizeof(UInt32), castedPointer)
})

let sessionId = String(format:"%llu", sessBytes)
```

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

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

Heuristic Buffer Overflow malloc

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 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

Incorrect Permission Assignment for Critical Resource

Weakness ID: 732 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software specifies permissions for a security-critical resource in a way that allows that resource to be read or modified by unintended actors.

Extended Description

When a resource is given a permissions setting that provides access to a wider range of actors than required, it could lead to the disclosure of sensitive information, or the modification of that resource by unintended parties. This is especially dangerous when the resource is related to program configuration, execution or sensitive user data.

Time of Introduction

- Architecture and Design
- Implementation
- Installation
- Operation

Applicable Platforms

Languages

Language-independent

Modes of Introduction

The developer may set loose permissions in order to minimize problems when the user first runs the program, then create documentation stating that permissions should be tightened. Since system administrators and users do not always read the documentation, this can result in insecure permissions being left unchanged.

The developer might make certain assumptions about the environment in which the software runs - e.g., that the software is running on a single-user system, or the software is only accessible to trusted administrators. When the software is running in a different environment, the permissions become a problem.

Common Consequences

Scope	Effect
Confidentiality	An attacker may be able to read sensitive information from the associated resource, such as credentials or configuration information stored in a file.
Integrity	An attacker may be able to modify critical properties of the associated resource to gain privileges, such as replacing a world-writable executable with a Trojan horse.
Availability	An attacker may be able to destroy or corrupt critical data in the associated resource, such as deletion of records from a database.

Likelihood of Exploit

Medium to High

Detection Methods

Automated Static Analysis

Automated static analysis may be effective in detecting permission problems for system resources such as files, directories, shared memory, device interfaces, etc. Automated techniques may be able to detect the use of library functions that modify permissions, then analyze function calls for arguments that contain potentially insecure values.

However, since the software's intended security policy might allow loose permissions for certain operations (such as publishing a file on a web server), automated static analysis may produce some false positives - i.e., warnings that do not have any security consequences or require any code changes.

When custom permissions models are used - such as defining who can read messages in a particular forum in a bulletin board system - these can be difficult to detect using automated static analysis. It may be possible to define custom signatures that

identify any custom functions that implement the permission checks and assignments.

Automated Dynamic Analysis

Automated dynamic analysis may be effective in detecting permission problems for system resources such as files, directories, shared memory, device interfaces, etc.

However, since the software's intended security policy might allow loose permissions for certain operations (such as publishing a file on a web server), automated dynamic analysis may produce some false positives - i.e., warnings that do not have any security consequences or require any code changes.

When custom permissions models are used - such as defining who can read messages in a particular forum in a bulletin board system - these can be difficult to detect using automated dynamic analysis. It may be possible to define custom signatures that identify any custom functions that implement the permission checks and assignments.

Manual Static Analysis

Manual static analysis may be effective in detecting the use of custom permissions models and functions. The code could then be examined to identifying usage of the related functions. Then the human analyst could evaluate permission assignments in the context of the intended security model of the software.

Manual Dynamic Analysis

Manual dynamic analysis may be effective in detecting the use of custom permissions models and functions. The program could then be executed with a focus on exercising code paths that are related to the custom permissions. Then the human analyst could evaluate permission assignments in the context of the intended security model of the software.

Fuzzing

Fuzzing is not effective in detecting this weakness.

Demonstrative Examples

Example 1

The following code sets the umask of the process to 0 before creating a file and writing "Hello world" into the file.

(Bad Code)

Example Language: **C**

```
#define OUTFILE "hello.out"

umask(0);
FILE *out;
/* Ignore CWE-59 (link following) for brevity */
out = fopen(OUTFILE, "w");
if (out) {
fprintf(out, "hello world!\n");
fclose(out);
}
```

After running this program on a UNIX system, running the "ls -l" command might return the following output:

(Result)

```
-rw-rw-rw- 1 username 13 Nov 24 17:58 hello.out
```

The "rw-rw-rw-" string indicates that the owner, group, and world (all users) can read the file and write to it.

Example 2

The following code snippet might be used as a monitor to periodically record whether a web site is alive. To ensure that the file can always be modified, the code uses chmod() to make the file world-writable.

(Bad Code)

Example Language: **Perl**

```
$fileName = "secretFile.out";

if (-e $fileName) {
chmod 0777, $fileName;
}
```

```
my $outFH;
if (! open($outFH, ">>$fileName")) {
ExitError("Couldn't append to $fileName: $!");
}
my $dateString = FormatCurrentTime();
my $status = IsHostAlive("cwe.mitre.org");
print $outFH "$dateString cwe status: $status!\n";
close($outFH);
```

The first time the program runs, it might create a new file that inherits the permissions from its environment. A file listing might look like:

(Result)

```
-rw-r--r-- 1 username 13 Nov 24 17:58 secretFile.out
```

This listing might occur when the user has a default umask of 022, which is a common setting. Depending on the nature of the file, the user might not have intended to make it readable by everyone on the system.

The next time the program runs, however - and all subsequent executions - the chmod will set the file's permissions so that the owner, group, and world (all users) can read the file and write to it:

(Result)

```
-rw-rw-rw- 1 username 13 Nov 24 17:58 secretFile.out
```

Perhaps the programmer tried to do this because a different process uses different permissions that might prevent the file from being updated.

Example 3

The following command recursively sets world-readable permissions for a directory and all of its children:

(Bad Code)

Example Language: **Shell**

```
chmod -R ugo+r DIRNAME
```

If this command is run from a program, the person calling the program might not expect that all the files under the directory will be world-readable. If the directory is expected to contain private data, this could become a security problem.

Observed Examples

Reference	Description
CVE-2009-3482	Anti-virus product sets insecure "Everyone: Full Control" permissions for files under the "Program Files" folder, allowing attackers to replace executables with Trojan horses.
CVE-2009-3897	Product creates directories with 0777 permissions at installation, allowing users to gain privileges and access a socket used for authentication.
CVE-2009-3489	Photo editor installs a service with an insecure security descriptor, allowing users to stop or start the service, or execute commands as SYSTEM.
CVE-2009-3289	Library function copies a file to a new target and uses the source file's permissions for the target, which is incorrect when the source file is a symbolic link, which typically has 0777 permissions.
CVE-2009-0115	Device driver uses world-writable permissions for a socket file, allowing attackers to inject arbitrary commands.
CVE-2009-1073	LDAP server stores a cleartext password in a world-readable file.
CVE-2009-0141	Terminal emulator creates TTY devices with world-writable permissions, allowing an attacker to write to the terminals of other users.

CVE-2008-0662	VPN product stores user credentials in a registry key with "Everyone: Full Control" permissions, allowing attackers to steal the credentials.
CVE-2008-0322	Driver installs its device interface with "Everyone: Write" permissions.
CVE-2009-3939	Driver installs a file with world-writable permissions.
CVE-2009-3611	Product changes permissions to 0777 before deleting a backup; the permissions stay insecure for subsequent backups.
CVE-2007-6033	Product creates a share with "Everyone: Full Control" permissions, allowing arbitrary program execution.
CVE-2007-5544	Product uses "Everyone: Full Control" permissions for memory-mapped files (shared memory) in inter-process communication, allowing attackers to tamper with a session.
CVE-2005-4868	Database product uses read/write permissions for everyone for its shared memory, allowing theft of credentials.
CVE-2004-1714	Security product uses "Everyone: Full Control" permissions for its configuration files.
CVE-2001-0006	"Everyone: Full Control" permissions assigned to a mutex allows users to disable network connectivity.
CVE-2002-0969	Chain: database product contains buffer overflow that is only reachable through a .ini configuration file - which has "Everyone: Full Control" permissions.

Potential Mitigations

Phase: Implementation

When using a critical resource such as a configuration file, check to see if the resource has insecure permissions (such as being modifiable by any regular user), and generate an error or even exit the software if there is a possibility that the resource could have been modified by an unauthorized party.

Phase: Architecture and Design

Divide your application into anonymous, normal, privileged, and administrative areas. Reduce the attack surface by carefully defining distinct user groups, privileges, and/or roles. Map these against data, functionality, and the related resources. Then set the permissions accordingly. This will allow you to maintain more fine-grained control over your resources.

Phases: Implementation; Installation

During program startup, explicitly set the default permissions or umask to the most restrictive setting possible. Also set the appropriate permissions during program installation. This will prevent you from inheriting insecure permissions from any user who installs or runs the program.

Phase: System Configuration

For all configuration files, executables, and libraries, make sure that they are only readable and writable by the software's administrator.

Phase: Documentation

Do not suggest insecure configuration changes in your documentation, especially if those configurations can extend to resources and other software that are outside the scope of your own software.

Phase: Installation

Do not assume that the system administrator will manually change the configuration to the settings that you recommend in the manual.

Phase: Testing

Use 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. These may be more effective than strictly automated techniques. This is especially the case with weaknesses that are related to design and business rules.

Phase: Testing

Use monitoring tools that examine the software's process as it interacts with the operating system and the network. This technique is useful in cases when source code is unavailable, if the software was not developed by you, or if you want to verify that the build phase did not introduce any new weaknesses. Examples include debuggers that directly attach to the running process; system-call tracing utilities such as truss (Solaris) and strace (Linux); system activity monitors such as FileMon, RegMon, Process Monitor, and other Sysinternals utilities (Windows); and sniffers and protocol analyzers that monitor network traffic.

Attach the monitor to the process and watch for library functions or system calls on OS resources such as files, directories, and shared memory. Examine the arguments to these calls to infer which permissions are being used.

Note that this technique is only useful for permissions issues related to system resources. It is not likely to detect application-level business rules that are related to permissions, such as if a user of a blog system marks a post as "private," but the blog system inadvertently marks it as "public."

Phases: Testing; System Configuration

Ensure that your software runs properly under the Federal Desktop Core Configuration (FDCC) or an equivalent hardening configuration guide, which many organizations use to limit the attack surface and potential risk of deployed software.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	275	Permission Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	668	Exposure of Resource to Wrong Sphere	**Research Concepts (primary)1000**
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**
RequiredBy	Compound Element: Composite	689	Permission Race Condition During Resource Copy	Research Concepts1000
ParentOf	Weakness Variant	276	Incorrect Default Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	277	Insecure Inherited Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	278	Insecure Preserved Inherited Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	279	Incorrect Execution-Assigned Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Base	281	Improper Preservation of Permissions	**Research Concepts (primary)1000**

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
232	Exploitation of Privilege/Trust	
1	Accessing Functionality Not Properly Constrained by ACLs	
17	Accessing, Modifying or Executing Executable Files	
60	Reusing Session IDs (aka Session Replay)	
61	Session Fixation	
62	Cross Site Request Forgery (aka Session Riding)	
122	Exploitation of Authorization	
180	Exploiting Incorrectly Configured Access Control Security Levels	
234	Hijacking a privileged process	

References

Mark Dowd, John McDonald and Justin Schuh. "The Art of Software Security Assessment". Chapter 9, "File Permissions." Page 495.. 1st Edition. Addison Wesley. 2006.

John Viega and Gary McGraw. "Building Secure Software". Chapter 8, "Access Control." Page 194.. 1st Edition. Addison-Wesley. 2002.

Maintenance Notes

The relationships between privileges, permissions, and actors (e.g. users and groups) need further refinement within the Research view. One complication is that these concepts apply to two different pillars, related to control of resources (CWE-664) and protection mechanism failures (CWE-396).

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
2008-09-08			Internal CWE Team
	new weakness-focused entry for Research view.		
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2009-01-12	CWE Content Team	MITRE	Internal
	updated Description, Likelihood of Exploit, Name, Potential Mitigations, Relationships		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Potential Mitigations, Related Attack Patterns		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Name		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Demonstrative Examples, Detection Factors, Modes of Introduction, Observed Examples, Potential Mitigations, References		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Potential Mitigations, Related Attack Patterns		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2009-01-12	Insecure Permission Assignment for Resource		
2009-05-27	Insecure Permission Assignment for Critical Resource		

BACK TO TOP

TOCTOU

Risk

What might happen

At best, a Race Condition may cause errors in accuracy, overidden values or unexpected behavior that may result in denial-of-service. At worst, it may allow attackers to retrieve data or bypass security processes by replaying a controllable Race Condition until it plays out in their favor.

Cause

How does it happen

Race Conditions occur when a public, single instance of a resource is used by multiple concurrent logical processes. If the these logical processes attempt to retrieve and update the resource without a timely management system, such as a lock, a Race Condition will occur.

An example for when a Race Condition occurs is a resource that may return a certain value to a process for further editing, and then updated by a second process, resulting in the original process' data no longer being valid. Once the original process edits and updates the incorrect value back into the resource, the second process' update has been overwritten and lost.

General Recommendations

How to avoid it

When sharing resources between concurrent processes across the application ensure that these resources are either thread-safe, or implement a locking mechanism to ensure expected concurrent activity.

Source Code Examples

Java

Different Threads Increment and Decrement The Same Counter Repeatedly, Resulting in a Race Condition

```
	public static int counter = 0;
	public static void start() throws InterruptedException {
		incrementCounter ic;
		decrementCounter dc;
		while(counter == 0) {
			counter = 0;
			ic = new incrementCounter();
			dc = new decrementCounter();
			ic.start();
			dc.start();
			ic.join();
			dc.join();
		}
		System.out.println(counter); //Will stop and return either -1 or 1 due to race
condition over counter
	}

	public static class incrementCounter extends Thread {
	    public void run() {
		counter++;
	    }
```

```
}

public static class decrementCounter extends Thread {
    public void run() {
       counter--;
    }
}
```

Different Threads Increment and Decrement The Same Thread-Safe Counter Repeatedly, Never Resulting in a Race Condition

```
public static int counter = 0;
public static Object lock = new Object();

public static void start() throws InterruptedException {
       incrementCounter ic;
       decrementCounter dc;
       while(counter == 0) { // because of proper locking, this condition is never false
              counter = 0;
              ic = new incrementCounter();
              dc = new decrementCounter();
              ic.start();
              dc.start();
              ic.join();
              dc.join();
       }
       System.out.println(counter); // Never reached
}

public static class incrementCounter extends Thread {
    public void run() {
       synchronized (lock) {
              counter++;
       }
    }
}

public static class decrementCounter extends Thread {
    public void run() {
       synchronized (lock) {
              counter--;
       }
    }
}
```

Scanned Languages

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