zfs Scan Report

Project Name	zfs
Scan Start	Friday, June 21, 2024 12:06:45 PM
Preset	Checkmarx Default
Scan Time	00h:02m:33s
Lines Of Code Scanned	20217
Files Scanned	14
Report Creation Time	Friday, June 21, 2024 12:10:56 PM
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013
Team	CxServer
Checkmarx Version	8.7.0
Scan Type	Full
Source Origin	LocalPath
Density	1/100 (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

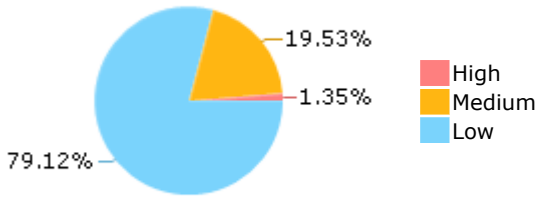
Results Limit

Results limit per query was set to 50

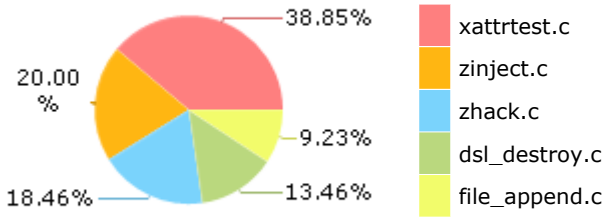
Selected Queries

Selected queries are listed in Result Summary

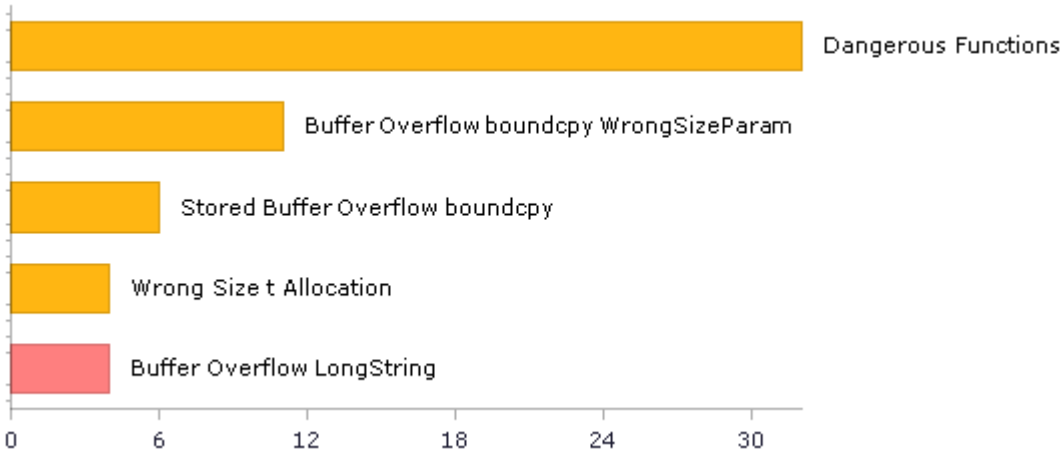
Result Summary



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	59	16
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	142	142
A3-Sensitive Data Exposure	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	2	2
A4-XML External Entities (XXE)	App. Specific	AVERAGE	COMMON	EASY	SEVERE	App. Specific	0	0
A5-Broken Access Control*	App. Specific	AVERAGE	COMMON	AVERAGE	SEVERE	App. Specific	1	1
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	32	32
A10-Insufficient Logging & Monitoring	App. Specific	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	App. Specific	0	0

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

Scan Summary - OWASP Top 10 2013

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

Category	Threat Agent	Attack Vectors	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	AVERAGE	SEVERE	ALL DATA	0	0
A2-Broken Authentication and Session Management	EXTERNAL, INTERNAL USERS	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	AFFECTED DATA AND FUNCTIONS	0	0
A3-Cross-Site Scripting (XSS)	EXTERNAL, INTERNAL, ADMIN USERS	AVERAGE	VERY WIDESPREAD	EASY	MODERATE	AFFECTED DATA AND SYSTEM	0	0
A4-Insecure Direct Object References	SYSTEM USERS	EASY	COMMON	EASY	MODERATE	EXPOSED DATA	1	1
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	32	32
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	20	15
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.	1	1
Audit And Accountability*	Organizations must: (i) create, protect, and retain information system audit records to the extent needed to enable the monitoring, analysis, investigation, and reporting of unlawful, unauthorized, or inappropriate information system activity; and (ii) ensure that the actions of individual information system users can be uniquely traced to those users so they can be held accountable for their actions.	0	0
Configuration Management	Organizations must: (i) establish and maintain baseline configurations and inventories of organizational information systems (including hardware, software, firmware, and documentation) throughout the respective system development life cycles; and (ii) establish and enforce security configuration settings for information technology products employed in organizational information systems.	27	18
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.	141	141
Media Protection	Organizations must: (i) protect information system media, both paper and digital; (ii) limit access to information on information system media to authorized users; and (iii) sanitize or destroy information system media before disposal or release for reuse.	2	2
System And Communications Protection	Organizations must: (i) monitor, control, and protect organizational communications (i.e., information transmitted or received by organizational information systems) at the external boundaries and key internal boundaries of the information systems; and (ii) employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational information systems.	0	0
System And Information Integrity	Organizations must: (i) identify, report, and correct information and information system flaws in a timely manner; (ii) provide protection from malicious code at appropriate locations within organizational information systems; and (iii) monitor information system security alerts and advisories and take appropriate actions in response.	2	2

***** 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)	169	160
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)	2	2
SC-4 Information in Shared Resources (P1)	0	0
SC-5 Denial of Service Protection (P1)*	37	4
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*	17	7
SI-11 Error Handling (P2)*	4	4
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

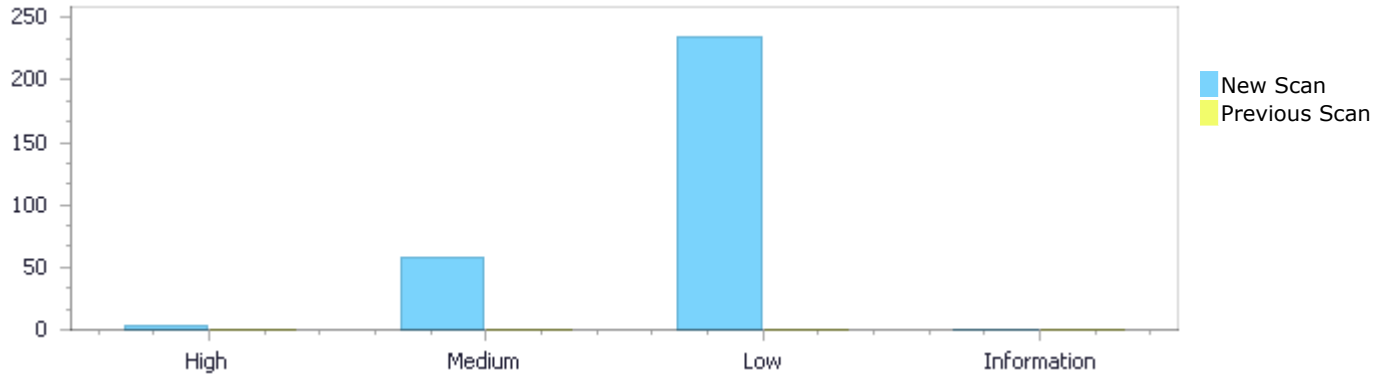
	modify the application's data and resources. This can provide the attacker a direct method of subverting the intended use of the software for personal or monetary gain.		
M9-Reverse Engineering	This category includes analysis of the final core binary to determine its source code, libraries, algorithms, 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	4	58	235	0	297
Recurrent Issues	0	0	0	0	0
Total	4	58	235	0	297
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	4	58	235	0	297
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	4	58	235	0	297

Result Summary

Vulnerability Type	Occurrences	Severity
Buffer Overflow LongString	4	High
Dangerous Functions	32	Medium
Buffer Overflow boundcpy WrongSizeParam	11	Medium
Stored Buffer Overflow boundcpy	6	Medium
Wrong Size t Allocation	4	Medium

Integer Overflow	2	Medium
Memory Leak	1	Medium
MemoryFree on StackVariable	1	Medium
Use of Zero Initialized Pointer	1	Medium
Improper Resource Access Authorization	141	Low
NULL Pointer Dereference	35	Low
Exposure of System Data to Unauthorized Control Sphere	27	Low
TOCTOU	9	Low
Inconsistent Implementations	7	Low
Unchecked Return Value	4	Low
Heuristic 2nd Order Buffer Overflow read	3	Low
Use of Sizeof On a Pointer Type	3	Low
Unchecked Array Index	2	Low
Use of Insufficiently Random Values	2	Low
Incorrect Permission Assignment For Critical Resources	1	Low
Potential Path Traversal	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
zfs/xattrtest.c	21
zfs/lobject.c	14
zfs/file_append.c	9
zfs/zfs_namecheck.c	5
zfs/lvm.c	4
zfs/zinject.c	4
zfs/lz4_zfs.c	3
zfs/zhack.c	2

Scan Results Details

Buffer Overflow LongString

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

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=1
Status	New

The size of the buffer used by post_hook in argv, at line 292 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that unlink_files passes to "post", at line 617 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	659	294
Object	"post"	argv

Code Snippet
File Name zfs/xattrtest.c
Method unlink_files(void)

```
....
659.        rc = post_hook("post");
```

File Name zfs/xattrtest.c
Method post_hook(const char *phase)

```
....
294.        char *argv[3] = { (char *)script, (char *)phase, NULL };
```

Buffer Overflow LongString\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=2
Status	New

The size of the buffer used by post_hook in argv, at line 292 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that create_files passes to "post", at line 350 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	407	294
Object	"post"	argv

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
407.        rc = post_hook("post");
```

File Name zfs/xattrtest.c
Method post_hook(const char *phase)

```
....
294.        char *argv[3] = { (char *)script, (char *)phase, NULL };
```

Buffer Overflow LongString\Path 3:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=3
Status	New

The size of the buffer used by post_hook in argv, at line 292 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that setxattrs passes to "post", at line 439 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	501	294
Object	"post"	argv

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
501.        rc = post_hook("post");
```

File Name	zfs/xattrtest.c
Method	post_hook(const char *phase)

```
....
294.        char *argv[3] = { (char *)script, (char *)phase, NULL };
```

Buffer Overflow LongString\Path 4:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=4
Status	New

The size of the buffer used by post_hook in argv, at line 292 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that getxattrs passes to "post", at line 513 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	602	294
Object	"post"	argv

Code Snippet

File Name	zfs/xattrtest.c
Method	getxattrs(void)

```
....
602.        rc = post_hook("post");
```

File Name	zfs/xattrtest.c
Method	post_hook(const char *phase)

```
....
294.        char *argv[3] = { (char *)script, (char *)phase, NULL };
```

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=1020017&projectid=20013&pathid=80
Status	New

The dangerous function, memcpy, was found in use at line 133 in zfs/file_append.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	167	167
Object	memcpy	memcpy

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
167.              memcpy(&buf[buf_offset], datapattern, amt);
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=81
Status	New

The dangerous function, memcpy, was found in use at line 246 in zfs/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	250	250
Object	memcpy	memcpy

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

```
....
250.        memcpy(out, source + 1, l * sizeof(char));
```

Dangerous Functions\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=82
Status	New

The dangerous function, memcpy, was found in use at line 246 in zfs/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	258	258
Object	memcpy	memcpy

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

```
....
258.        memcpy(out, source + 1, l * sizeof(char));
```

Dangerous Functions\Path 4:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=83
Status New

The dangerous function, memcpy, was found in use at line 246 in zfs/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	262	262
Object	memcpy	memcpy

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

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

Dangerous Functions\Path 5:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=84
Status New

The dangerous function, memcpy, was found in use at line 246 in zfs/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	278	278
Object	memcpy	memcpy

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

```
....
278.      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=1020017&projectid=20013&pathid=85
Status New

The dangerous function, memcpy, was found in use at line 293 in zfs/lvm.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/lvm.c	zfs/lvm.c
Line	324	324
Object	memcpy	memcpy

Code Snippet
File Name zfs/lvm.c
Method void luaV_concat (lua_State *L, int total) {

```
....
324.          memcpy(buffer+tl, svalue(top-i), l * sizeof(char));
```

Dangerous Functions\Path 7:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=86
Status New

The dangerous function, memcpy, was found in use at line 473 in zfs/lz4_zfs.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/lz4_zfs.c	zfs/lz4_zfs.c
Line	644	644
Object	memcpy	memcpy

Code Snippet
File Name zfs/lz4_zfs.c
Method LZ4_compressCtx(void *ctx, const char *source, char *dest, int isize,

```
....
644.                (void) memcpy(op, anchor, iend - anchor);
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=87
Status	New

The dangerous function, memcpy, was found in use at line 663 in zfs/lz4_zfs.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/lz4_zfs.c	zfs/lz4_zfs.c
Line	831	831
Object	memcpy	memcpy

Code Snippet
File Name zfs/lz4_zfs.c
Method LZ4_compress64kCtx(void *ctx, const char *source, char *dest, int isize,

```
....
831.                (void) memcpy(op, anchor, iend - anchor);
```

Dangerous Functions\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=88
Status	New

The dangerous function, memcpy, was found in use at line 439 in zfs/xattrtest.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c

Line	484	484
Object	memcpy	memcpy

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
484.                    memcpy(value + shift, xattrbytes,
```

Dangerous Functions\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=89
Status	New

The dangerous function, memcpy, was found in use at line 513 in zfs/xattrtest.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	583	583
Object	memcpy	memcpy

Code Snippet
File Name zfs/xattrtest.c
Method getxattrs(void)

```
....
583.                    memcpy(verify_value + shift, xattrbytes,
```

Dangerous Functions\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=90
Status	New

The dangerous function, sprintf, was found in use at line 439 in zfs/xattrtest.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	482	482
Object	sprintf	sprintf

Code Snippet	
File Name	zfs/xattrtest.c
Method	setxattrs(void)

```
....
482.                    (void) sprintf(name, "user.%d", j);
```

Dangerous Functions\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=91
Status	New

The dangerous function, sprintf, was found in use at line 439 in zfs/xattrtest.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	483	483
Object	sprintf	sprintf

Code Snippet	
File Name	zfs/xattrtest.c
Method	setxattrs(void)

```
....
483.                    shift = sprintf(value, "size=%d ", rnd_size);
```

Dangerous Functions\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=92
Status	New

The dangerous function, sprintf, was found in use at line 513 in zfs/xattrtest.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	568	568
Object	sprintf	sprintf

Code Snippet	
File Name	zfs/xattrtest.c

Method	getxattrs(void)

```
....
568.                    (void) sprintf(name, "user.%d", j);
```

Dangerous Functions\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=93
Status	New

The dangerous function, sprintf, was found in use at line 513 in zfs/xattrtest.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	581	581
Object	sprintf	sprintf

Code Snippet	
File Name	zfs/xattrtest.c
Method	getxattrs(void)

```
....
581.                    shift = sprintf(verify_value, "size=%d ",
```

Dangerous Functions\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=94
Status	New

The dangerous function, sscanf, was found in use at line 513 in zfs/xattrtest.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	580	580
Object	sscanf	sscanf

Code Snippet	
File Name	zfs/xattrtest.c
Method	getxattrs(void)

```
....
580.                    sscanf(value, "size=%d [a-z]", &rnd_size);
```

Dangerous Functions\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=95
Status	New

The dangerous function, sscanf, was found in use at line 634 in zfs/zinject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	639	639
Object	sscanf	sscanf

Code Snippet
File Name zfs/zinject.c
Method parse_delay(char *str, uint64_t *delay, uint64_t *nlanes)

```
....
639.        if (sscanf(str, "%lu:%lu", &scan_delay, &scan_nlanes) != 2)
```

Dangerous Functions\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=96
Status	New

The dangerous function, strlen, was found in use at line 133 in zfs/file_append.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	166	166
Object	strlen	strlen

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
166.              size_t amt = MIN(strlen(datapattern), left);
```

Dangerous Functions\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=97
Status	New

The dangerous function, strlen, was found in use at line 246 in zfs/lobject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	247	247
Object	strlen	strlen

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

```
....
247.    size_t l = strlen(source);
```

Dangerous Functions\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=98
Status	New

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

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	184	184
Object	strlen	strlen

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

```
....
184.          pushstr(L, s, strlen(s));
```

Dangerous Functions\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=99
Status	New

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

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	221	221
Object	strlen	strlen

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

```
....
221.    pushstr(L, fmt, strlen(fmt));
```

Dangerous Functions\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=100
Status	New

The dangerous function, strlen, was found in use at line 209 in zfs/lvm.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/lvm.c	zfs/lvm.c
Line	218	218
Object	strlen	strlen

Code Snippet
File Name zfs/lvm.c
Method static int l_strcmp (const TString *ls, const TString *rs) {

```
....
218.        size_t len = strlen(l);  /* index of first `\0' in both
strings */
```

Dangerous Functions\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=101
Status	New

The dangerous function, strlen, was found in use at line 98 in zfs/zfs_namecheck.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/zfs_namecheck.c	zfs/zfs_namecheck.c
Line	102	102
Object	strlen	strlen

Code Snippet
File Name zfs/zfs_namecheck.c
Method zfs_component_namecheck(const char *path, namecheck_err_t *why, char *what)

```
....
102.        if (strlen(path) >= ZFS_MAX_DATASET_NAME_LEN) {
```

Dangerous Functions\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=102
Status	New

The dangerous function, strlen, was found in use at line 135 in zfs/zfs_namecheck.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/zfs_namecheck.c	zfs/zfs_namecheck.c
Line	137	137
Object	strlen	strlen

Code Snippet
File Name zfs/zfs_namecheck.c
Method permset_namecheck(const char *path, namecheck_err_t *why, char *what)

```
....
137.        if (strlen(path) >= ZFS_PERMSET_MAXLEN) {
```

Dangerous Functions\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=103
Status	New

The dangerous function, strlen, was found in use at line 182 in zfs/zfs_namecheck.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/zfs_namecheck.c	zfs/zfs_namecheck.c
Line	191	191
Object	strlen	strlen

Code Snippet
File Name zfs/zfs_namecheck.c
Method entity_namecheck(const char *path, namecheck_err_t *why, char *what)

```
....
191.        if (strlen(path) >= ZFS_MAX_DATASET_NAME_LEN) {
```

Dangerous Functions\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=104
Status	New

The dangerous function, strlen, was found in use at line 407 in zfs/zfs_namecheck.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/zfs_namecheck.c	zfs/zfs_namecheck.c
Line	419	419
Object	strlen	strlen

Code Snippet
File Name zfs/zfs_namecheck.c
Method pool_namecheck(const char *pool, namecheck_err_t *why, char *what)

```
....
419.        if (strlen(pool) >= (ZFS_MAX_DATASET_NAME_LEN - 2 -
```

Dangerous Functions\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=105
Status	New

The dangerous function, strlen, was found in use at line 407 in zfs/zfs_namecheck.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/zfs_namecheck.c	zfs/zfs_namecheck.c
Line	420	420
Object	strlen	strlen

Code Snippet
File Name zfs/zfs_namecheck.c
Method pool_namecheck(const char *pool, namecheck_err_t *why, char *what)

```
....
420.            strlen(ORIGIN_DIR_NAME) * 2)) {
```

Dangerous Functions\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=106
Status	New

The dangerous function, strlen, was found in use at line 692 in zfs/zinject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	699	699
Object	strlen	strlen

Code Snippet
File Name zfs/zinject.c
Method parse_dvas(const char *str, uint32_t *dvas_out)

```
....
699.        if (strlen(str) > 5 || strlen(str) == 0)
```

Dangerous Functions\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=107
Status	New

The dangerous function, strlen, was found in use at line 692 in zfs/zinject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	699	699
Object	strlen	strlen

Code Snippet
File Name zfs/zinject.c
Method parse_dvas(const char *str, uint32_t *dvas_out)

```
....
699.        if (strlen(str) > 5 || strlen(str) == 0)
```

Dangerous Functions\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=108
Status	New

The dangerous function, atoi, was found in use at line 72 in zfs/file_append.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	82	82
Object	atoi	atoi

Code Snippet
File Name zfs/file_append.c
Method parse_options(int argc, char *argv[])

```
....
82.                     blocksize = atoi(optarg);
```

Dangerous Functions\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=109
Status	New

The dangerous function, atoi, was found in use at line 72 in zfs/file_append.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	88	88
Object	atoi	atoi

Code Snippet
File Name zfs/file_append.c
Method parse_options(int argc, char *argv[])

```
....
88.                     expected_offset = atoi(optarg);
```

Dangerous Functions\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=110
Status	New

The dangerous function, atoi, was found in use at line 72 in zfs/file_append.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	97	97
Object	atoi	atoi

Code Snippet
File Name zfs/file_append.c
Method parse_options(int argc, char *argv[])

```
....
97.                     numblocks = atoi(optarg);
```

Dangerous Functions\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=111
Status	New

The dangerous function, atoi, was found in use at line 737 in zfs/zinject.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	1149	1149
Object	atoi	atoi

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

```
....
1149.                   record.zi_type = atoi(argv[1]);
```

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=1020017&projectid=20013&pathid=22
Status	New

The size of the buffer used by real_LZ4_compress in refTables, at line 840 of zfs/lz4_zfs.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that real_LZ4_compress passes to refTables, at line 840 of zfs/lz4_zfs.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lz4_zfs.c	zfs/lz4_zfs.c
Line	855	855

Object	refTables	refTables

Code Snippet
File Name zfs/lz4_zfs.c
Method real_LZ4_compress(const char *source, char *dest, int isize, int osize)

```
....
855.        memset(ctx, 0, sizeof (struct refTables));
```

Buffer Overflow boundcpy WrongSizeParam\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=23
Status	New

The size of the buffer used by luaO_chunkid in l, at line 246 of zfs/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 l, at line 246 of zfs/lobject.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	250	250
Object	l	l

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

```
....
250.        memcpy(out, source + 1, l * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=24
Status	New

The size of the buffer used by luaO_chunkid in char, at line 246 of zfs/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 246 of zfs/lobject.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	250	250
Object	char	char

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

```
....
250.        memcpy(out, source + 1, l * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=25
Status	New

The size of the buffer used by luaO_chunkid in l, at line 246 of zfs/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 l, at line 246 of zfs/lobject.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	258	258
Object	l	l

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

```
....
258.        memcpy(out, source + 1, l * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=26
Status	New

The size of the buffer used by luaO_chunkid in char, at line 246 of zfs/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 246 of zfs/lobject.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	258	258
Object	char	char

Code Snippet	
File Name	zfs/lobject.c

Method	void luaO_chunkid (char *out, const char *source, size_t bufflen) {

```
....
258.        memcpy(out, source + 1, l * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=27
Status	New

The size of the buffer used by luaO_chunkid in bufflen, at line 246 of zfs/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 246 of zfs/lobject.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	262	262
Object	bufflen	bufflen

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

```
....
262.        memcpy(out, source + 1 + l - 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=1020017&projectid=20013&pathid=28
Status	New

The size of the buffer used by luaO_chunkid in char, at line 246 of zfs/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 246 of zfs/lobject.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	262	262
Object	char	char

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=29
Status	New

The size of the buffer used by luaO_chunkid in char, at line 246 of zfs/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 246 of zfs/lobject.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	278	278
Object	char	char

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=30
Status	New

The size of the buffer used by luaV_concat in l, at line 293 of zfs/lvm.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaV_concat passes to l, at line 293 of zfs/lvm.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lvm.c	zfs/lvm.c
Line	324	324
Object	l	l

Code Snippet
File Name zfs/lvm.c
Method void luaV_concat (lua_State *L, int total) {

```
....
324.          memcpy(buffer+tl, svalue(top-i), l * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=31
Status	New

The size of the buffer used by luaV_concat in char, at line 293 of zfs/lvm.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaV_concat passes to char, at line 293 of zfs/lvm.c, to overwrite the target buffer.

	Source	Destination
File	zfs/lvm.c	zfs/lvm.c
Line	324	324
Object	char	char

Code Snippet
File Name zfs/lvm.c
Method void luaV_concat (lua_State *L, int total) {

```
....
324.          memcpy(buffer+tl, svalue(top-i), l * sizeof(char));
```

Buffer Overflow boundcpy WrongSizeParam\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=32
Status	New

The size of the buffer used by main in amt, at line 133 of zfs/file_append.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 amt, at line 133 of zfs/file_append.c, to overwrite the target buffer.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	167	167
Object	amt	amt

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
167.              memcpy(&buf[buf_offset], datapattern, amt);
```

Stored Buffer Overflow boundcpy

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

Categories

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

Description

Stored Buffer Overflow boundcpy\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=114
Status	New

The size of the buffer used by getxattrs in xattrbytes, at line 513 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that get_random_bytes passes to BinaryExpr, at line 416 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	584
Object	BinaryExpr	xattrbytes

Code Snippet
File Name zfs/xattrtest.c
Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

File Name zfs/xattrtest.c

Method getxattrs(void)

```
....
584.                          sizeof (xattrbytes) - shift);
```

Stored Buffer Overflow boundcpy\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=115

Status	New

The size of the buffer used by getxattrs in sizeof, at line 513 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that get_random_bytes passes to BinaryExpr, at line 416 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	584
Object	BinaryExpr	sizeof

Code Snippet
File Name zfs/xattrtest.c
Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

File Name zfs/xattrtest.c

Method getxattrs(void)

```
....
584.                        sizeof (xattrbytes) - shift);
```

Stored Buffer Overflow boundcpy\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=116
Status	New

The size of the buffer used by getxattrs in BinaryExpr, at line 513 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that get_random_bytes passes to BinaryExpr, at line 416 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	584
Object	BinaryExpr	BinaryExpr

Code Snippet
File Name zfs/xattrtest.c
Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

File Name zfs/xattrtest.c

Method getxattrs(void)

```
....
584.                          sizeof (xattrbytes) - shift);
```

Stored Buffer Overflow boundcpy\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=117
Status	New

The size of the buffer used by setxattrs in xattrbytes, at line 439 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that get_random_bytes passes to BinaryExpr, at line 416 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	485
Object	BinaryExpr	xattrbytes

Code Snippet

File Name zfs/xattrtest.c

Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

File Name zfs/xattrtest.c

Method setxattrs(void)

```
....
485.                          sizeof (xattrbytes) - shift);
```

Stored Buffer Overflow boundcpy\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=118
Status	New

The size of the buffer used by setxattrs in sizeof, at line 439 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that get_random_bytes passes to BinaryExpr, at line 416 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	485
Object	BinaryExpr	sizeof

Code Snippet
File Name zfs/xattrtest.c
Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

File Name zfs/xattrtest.c

Method setxattrs(void)

```
....
485.                          sizeof (xattrbytes) - shift);
```

Stored Buffer Overflow boundcpy\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=119
Status	New

The size of the buffer used by setxattrs in BinaryExpr, at line 439 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that get_random_bytes passes to BinaryExpr, at line 416 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	485
Object	BinaryExpr	BinaryExpr

Code Snippet
File Name zfs/xattrtest.c
Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

File Name zfs/xattrtest.c

Method setxattrs(void)

```
....
485.                        sizeof (xattrbytes) - shift);
```

Wrong Size t Allocation

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

Description

Wrong Size t Allocation\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=34
Status	New

The function fsize in zfs/xattrtest.c at line 350 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	359	359
Object	fsize	fsize

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
359.        file = malloc(fsize);
```

Wrong Size t Allocation\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=35
Status	New

The function fsize in zfs/xattrtest.c at line 439 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	458	458
Object	fsize	fsize

Code Snippet
File Name zfs/xattrtest.c

Method	setxattrs(void)

```
....
458.        file = malloc(fsize);
```

Wrong Size t Allocation\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=36
Status	New

The function fsize in zfs/xattrtest.c at line 513 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	546	546
Object	fsize	fsize

Code Snippet	
File Name	zfs/xattrtest.c
Method	getxattrs(void)

```
....
546.        file = malloc(fsize);
```

Wrong Size t Allocation\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=37
Status	New

The function fsize in zfs/xattrtest.c at line 617 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	626	626
Object	fsize	fsize

Code Snippet	
File Name	zfs/xattrtest.c
Method	unlink_files(void)

```
....
626.        file = malloc(fsize);
```

Integer Overflow

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

Categories

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

Description

Integer Overflow\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=76
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 133 of zfs/file_append.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	168	168
Object	AssignExpr	AssignExpr

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
168.              buf_offset += amt;
```

Integer Overflow\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=77
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 133 of zfs/file_append.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	169	169

Object	AssignExpr	AssignExpr

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
169.                left -= amt;
```

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=1020017&projectid=20013&pathid=33
Status New

Calling free() (line 133) on a variable that was not dynamically allocated (line 133) in file zfs/file_append.c may result with a crash.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	203	203
Object	buf	buf

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
203.        free(buf);
```

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=1020017&projectid=20013&pathid=112
Status New

	Source	Destination
File	zfs/zhack.c	zfs/zhack.c
Line	151	151
Object	g_pool	g_pool

Code Snippet
File Name zfs/zhack.c
Method zhack_import(char *target, boolean_t readonly)

```
....
151.        g_pool = strdup(target);
```

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=1020017&projectid=20013&pathid=113
Status	New

The variable declared in desc at zfs/zhack.c in line 286 is not initialized when it is used by fi_desc at zfs/zhack.c in line 286.

	Source	Destination
File	zfs/zhack.c	zfs/zhack.c
Line	299	324
Object	desc	fi_desc

Code Snippet
File Name zfs/zhack.c
Method zhack_do_feature_enable(int argc, char **argv)

```
....
299.        desc = NULL;
....
324.        feature.fi_desc = desc;
```

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=1020017&projectid=20013&pathid=120
Status	New

	Source	Destination
File	zfs/gethostid.c	zfs/gethostid.c
Line	50	50
Object	fscanf	fscanf

Code Snippet
File Name zfs/gethostid.c
Method get_spl_hostid(void)

```
....
50.   if (fscanf(f, "%lx", &hostid) != 1)
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=121
Status	New

	Source	Destination
File	zfs/gethostid.c	zfs/gethostid.c
Line	50	50
Object	Address	Address

Code Snippet
File Name zfs/gethostid.c
Method get_spl_hostid(void)

```
....
50.   if (fscanf(f, "%lx", &hostid) != 1)
```

Improper Resource Access Authorization\Path 3:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=122
Status	New

	Source	Destination
File	zfs/gethostid.c	zfs/gethostid.c
Line	73	73
Object	Address	Address

Code Snippet
File Name zfs/gethostid.c
Method get_system_hostid(void)

```
....
73.               if (read(fd, &system_hostid, sizeof (system_hostid))
```

Improper Resource Access Authorization\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=123
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	427
Object	BinaryExpr	BinaryExpr

Code Snippet
File Name zfs/xattrtest.c
Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

Improper Resource Access Authorization\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=124
Status	New

	Source	Destination

File	zfs/zhack.c	zfs/zhack.c
Line	510	510
Object	vl	vl

Code Snippet
File Name zfs/zhack.c
Method zhack_repair_read_label(const int fd, vdev_label_t *vl,

```
....
510.        const int err = pread64(fd, vl, sizeof (vdev_label_t),
label_offset);
```

Improper Resource Access Authorization\Path 6:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=125
Status New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	149	149
Object	fprintf	fprintf

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
149.                (void) fprintf(stderr, "%s: %s: ", execname,
filename);
```

Improper Resource Access Authorization\Path 7:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=126
Status New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	158	158
Object	fprintf	fprintf

Code Snippet

File Name	zfs/file_append.c
Method	main(int argc, char *argv[])

```
....
158.              (void) fprintf(stderr,
```

Improper Resource Access Authorization\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=127
Status	New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	179	179
Object	fprintf	fprintf

Code Snippet

File Name	zfs/file_append.c
Method	main(int argc, char *argv[])

```
....
179.                          (void) fprintf(stderr,
```

Improper Resource Access Authorization\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=128
Status	New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	195	195
Object	fprintf	fprintf

Code Snippet

File Name	zfs/file_append.c
Method	main(int argc, char *argv[])

```
....
195.              (void) fprintf(stderr,
```

Improper Resource Access Authorization\Path 10:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=129
Status	New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	40	40
Object	fprintf	fprintf

Code Snippet
File Name zfs/file_append.c
Method usage(void)

```
....
40.    (void) fprintf(stderr,
```

Improper Resource Access Authorization\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=130
Status	New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	100	100
Object	fprintf	fprintf

Code Snippet
File Name zfs/file_append.c
Method parse_options(int argc, char *argv[])

```
....
100.                            (void) fprintf(stderr,
```

Improper Resource Access Authorization\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=131
Status	New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c

Line	107	107
Object	fprintf	fprintf

Code Snippet
File Name zfs/file_append.c
Method parse_options(int argc, char *argv[])

```
....
107.                            (void) fprintf(stderr,
```

Improper Resource Access Authorization\Path 13:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=132
Status New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	107	107
Object	fprintf	fprintf

Code Snippet
File Name zfs/file_append.c
Method parse_options(int argc, char *argv[])

```
....
107.                            (void) fprintf(stderr,
```

Improper Resource Access Authorization\Path 14:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=133
Status New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	118	118
Object	fprintf	fprintf

Code Snippet
File Name zfs/file_append.c
Method parse_options(int argc, char *argv[])

```
....
118.              (void) fprintf(stderr,
```

Improper Resource Access Authorization\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=134
Status	New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	125	125
Object	fprintf	fprintf

Code Snippet
File Name zfs/file_append.c
Method parse_options(int argc, char *argv[])

```
....
125.              (void) fprintf(stderr,
```

Improper Resource Access Authorization\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=135
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	101	101
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method usage(char *argv0)

```
....
101.        fprintf(stderr,
```

Improper Resource Access Authorization\Path 17:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=136
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	106	106
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method usage(char *argv0)

```
....
106.        fprintf(stderr,
```

Improper Resource Access Authorization\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=137
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	158	158
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
158.                        fprintf(stderr, "Error: the -s value may
not "
```

Improper Resource Access Authorization\Path 19:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=138
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c

Line	190	190
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
190.                            fprintf(stderr, "Error: the -o value must
be "
```

Improper Resource Access Authorization\Path 20:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=139
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	208	208
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
208.                fprintf(stdout, "verbose:           %d\n", verbose);
```

Improper Resource Access Authorization\Path 21:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=140
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	209	209
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
209.            fprintf(stdout, "verify:          %d\n", verify);
```

Improper Resource Access Authorization\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=141
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	210	210
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
210.            fprintf(stdout, "nth:             %d\n", nth);
```

Improper Resource Access Authorization\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=142
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	211	211
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
211.            fprintf(stdout, "files:           %d\n", files);
```

Improper Resource Access Authorization\Path 24:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=143
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	212	212
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
212.                fprintf(stdout, "xattrs:          %d\n", xattrs);
```

Improper Resource Access Authorization\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=144
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	213	213
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
213.                fprintf(stdout, "size:            %d\n", size);
```

Improper Resource Access Authorization\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=145
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	214	214

Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
214.                fprintf(stdout, "path:              %s\n", path);
```

Improper Resource Access Authorization\Path 27:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=146
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	215	215
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
215.                fprintf(stdout, "synccaches:        %d\n", synccaches);
```

Improper Resource Access Authorization\Path 28:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=147
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	216	216
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
216.              fprintf(stdout, "dropcaches:       %d\n", dropcaches);
```

Improper Resource Access Authorization\Path 29:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=148
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	217	217
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
217.              fprintf(stdout, "script:           %s\n", script);
```

Improper Resource Access Authorization\Path 30:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=149
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	218	218
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
218.              fprintf(stdout, "seed:             %ld\n", seed);
```

Improper Resource Access Authorization\Path 31:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=150
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	219	219
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
219.                fprintf(stdout, "random size:       %d\n",
size_is_random);
```

Improper Resource Access Authorization\Path 32:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=151
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	220	220
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
220.                fprintf(stdout, "random value:      %d\n",
value_is_random);
```

Improper Resource Access Authorization\Path 33:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=152
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c

Line	221	221
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
221.                fprintf(stdout, "keep:              %d\n", keep_files);
```

Improper Resource Access Authorization\Path 34:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=153
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	222	222
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
222.                fprintf(stdout, "only:              %d\n", phase);
```

Improper Resource Access Authorization\Path 35:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=154
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	223	223
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
223.              fprintf(stdout, "%s", "\n");
```

Improper Resource Access Authorization\Path 36:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=155
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	237	237
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method drop_caches(void)

```
....
237.              ERROR("Error %d: open(\"%s\", O_WRONLY)\n", errno,
file);
```

Improper Resource Access Authorization\Path 37:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=156
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	243	243
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method drop_caches(void)

```
....
243.              ERROR("Error %d: write(%d, \"3\", 1)\n", errno, fd);
```

Improper Resource Access Authorization\Path 38:

Severity	Low
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=157
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	250	250
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method drop_caches(void)

```
....
250.              ERROR("Error %d: close(%d)\n", errno, fd);
```

Improper Resource Access Authorization\Path 39:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=158
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	362	362
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
362.              ERROR("Error %d: malloc(%d) bytes for file name\n",
rc,
```

Improper Resource Access Authorization\Path 40:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=159
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c

Line	372	372
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
372.                    ERROR("Error %d: path too long\n", rc);
```

Improper Resource Access Authorization\Path 41:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=160
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	377	377
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
377.                    fprintf(stdout, "create: %s\n", file);
```

Improper Resource Access Authorization\Path 42:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=161
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	381	381
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
381.                    ERROR("Error %d: unlink(%s)\n", errno, file);
```

Improper Resource Access Authorization\Path 43:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=162
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	388	388
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
388.                    ERROR("Error %d: open(%s, O_CREATE, 0644)\n",
```

Improper Resource Access Authorization\Path 44:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=163
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	396	396
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
396.                    ERROR("Error %d: close(%d)\n", errno, rc);
```

Improper Resource Access Authorization\Path 45:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=164
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	404	404
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
404.        fprintf(stdout, "create:   %f seconds %f creates/second\n",
```

Improper Resource Access Authorization\Path 46:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=165
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	452	452
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
452.             ERROR("Error %d: malloc(%d) bytes for xattr value\n",
rc,
```

Improper Resource Access Authorization\Path 47:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=166
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c

Line	461	461
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
461.              ERROR("Error %d: malloc(%d) bytes for file name\n",
rc,
```

Improper Resource Access Authorization\Path 48:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=167
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	471	471
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
471.                    ERROR("Error %d: path too long\n", rc);
```

Improper Resource Access Authorization\Path 49:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=168
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	476	476
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
476.                    fprintf(stdout, "setxattr: %s\n", file);
```

Improper Resource Access Authorization\Path 50:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=169
Status	New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	489	489
Object	fprintf	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
489.                        ERROR("Error %d: lsetxattr(%s, %s, ...,
%d)\n",
```

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=1020017&projectid=20013&pathid=38
Status	New

The variable declared in null at zfs/dsl_destroy.c in line 828 is not initialized when it is used by dd at zfs/dsl_destroy.c in line 828.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	837	869
Object	null	dd

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_dir_destroy_sync(uint64_t ddobj, dmu_tx_t *tx)

```
....
837.        VERIFY0(dsl_dir_hold_obj(dp, ddobj, NULL, FTAG, &dd));
....
869.            dd->dd_myname, tx));
```

NULL Pointer Dereference\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=39
Status	New

The variable declared in null at zfs/dsl_destroy.c in line 828 is not initialized when it is used by dd at zfs/dsl_destroy.c in line 828.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	837	868
Object	null	dd

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_dir_destroy_sync(uint64_t ddobj, dmu_tx_t *tx)

```
....
837.        VERIFY0(dsl_dir_hold_obj(dp, ddobj, NULL, FTAG, &dd));
....
868.            dsl_dir_phys(dd->dd_parent)->dd_child_dir_zapobj,
```

NULL Pointer Dereference\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=40
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_next at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	491
Object	0	ds_next

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
491.                           ds_next, &ds_next->ds_prev));
```

NULL Pointer Dereference\Path 4:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=41
Status New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_nextnext at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	460
Object	0	ds_nextnext

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
460.                dsl_deadlist_space_range(&ds_nextnext->ds_deadlist,
```

NULL Pointer Dereference\Path 5:

Severity Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=42
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_next at zfs/dsl_destroy.c in line 273.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	297
Object	0	ds_next

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_handle_remaps(dsl_dataset_t *ds, dsl_dataset_t *ds_next,

```
....
297.              mutex_exit(&ds_next->ds_remap_deadlist_lock);
```

NULL Pointer Dereference\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=43
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_next at zfs/dsl_destroy.c in line 273.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	294
Object	0	ds_next

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_handle_remaps(dsl_dataset_t *ds, dsl_dataset_t *ds_next,

```
....
294.              mutex_enter(&ds_next->ds_remap_deadlist_lock);
```

NULL Pointer Dereference\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=44
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_next at zfs/dsl_destroy.c in line 273.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	299
Object	0	ds_next

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_handle_remaps(dsl_dataset_t *ds, dsl_dataset_t *ds_next,

```
....
299.              dsl_deadlist_merge(&ds_next->ds_remap_deadlist,
```

NULL Pointer Dereference\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=45

Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_next at zfs/dsl_destroy.c in line 153.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	186
Object	0	ds_next

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c
Method process_old_deadlist(dsl_dataset_t *ds, dsl_dataset_t *ds_prev,

```
....
186.        dsl_deadlist_open(&ds_next->ds_deadlist, mos,
```

NULL Pointer Dereference\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=46
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_next at zfs/dsl_destroy.c in line 153.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	179
Object	0	ds_next

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name	zfs/dsl_destroy.c
Method	process_old_deadlist(dsl_dataset_t *ds, dsl_dataset_t *ds_prev,

```
....
179.        dsl_deadlist_close(&ds_next->ds_deadlist);
```

NULL Pointer Dereference\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=47
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_deadlist at zfs/dsl_destroy.c in line 153.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	168
Object	0	ds_deadlist

Code Snippet

File Name	zfs/dsl_destroy.c
Method	dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name	zfs/dsl_destroy.c
Method	process_old_deadlist(dsl_dataset_t *ds, dsl_dataset_t *ds_prev,

```
....
168.        VERIFY0(bpobj_iterate(&ds_next->ds_deadlist.dl_bpobj,
```

NULL Pointer Dereference\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=48
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_deadlist at zfs/dsl_destroy.c in line 153.

	Source	Destination

File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	162
Object	0	ds_deadlist

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c
Method process_old_deadlist(dsl_dataset_t *ds, dsl_dataset_t *ds_prev,

```
....
162.        ASSERT(ds_next->ds_deadlist.dl_oldfmt);
```

NULL Pointer Dereference\Path 12:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=49
Status New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	520
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c
Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
520.              dmu_objset_evict(ds->ds_objset);
```

NULL Pointer Dereference\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=50
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	519
Object	0	ds

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
519.        if (ds->ds_objset) {
```

NULL Pointer Dereference\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=51
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c

Line	1013	547
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
547.        spa_prop_clear_bootfs(dp->dp_spa, ds->ds_object, tx);
```

NULL Pointer Dereference\Path 15:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=52
Status New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	434
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
434.        dmu_buf_will_dirty(ds->ds_dbuf, tx);
```

NULL Pointer Dereference\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=53
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	528
Object	0	ds

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
528.            dsl_dir_phys(ds->ds_dir)->dd_head_dataset_obj, FTAG,
&ds_head));
```

NULL Pointer Dereference\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=54
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c

Line	1013	563
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
563.        dsl_dir_rele(ds->ds_dir, ds);
```

NULL Pointer Dereference\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=55
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	471
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
471.                    dsl_dir_phys(ds->ds_dir)->dd_head_dataset_obj,
```

NULL Pointer Dereference\Path 19:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=56
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by hds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	475
Object	0	hds

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
475.                    dsl_deadlist_remove_key(&hds->ds_deadlist,
```

NULL Pointer Dereference\Path 20:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=57
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c

Line	1013	509
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
509.                    dsl_dir_diduse_space(ds->ds_dir,
```

NULL Pointer Dereference\Path 21:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=58
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	441
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
441.                dsl_dir_remove_clones_key(ds->ds_dir,
```

NULL Pointer Dereference\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=59
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by dd at zfs/dsl_destroy.c in line 204.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	225
Object	0	dd

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_dir_remove_clones_key_impl(dsl_dir_t *dd, uint64_t mintxg, dmu_tx_t *tx,

```
....
225.                VERIFY0(dsl_dataset_hold_obj(dd->dd_pool,
```

NULL Pointer Dereference\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=60
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by dd_pool at zfs/dsl_destroy.c in line 204.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	207

Object	0	dd_pool

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c
Method dsl_dir_remove_clones_key_impl(dsl_dir_t *dd, uint64_t mintxg, dmu_tx_t *tx,

```
....
207.        objset_t *mos = dd->dd_pool->dp_meta_objset;
```

NULL Pointer Dereference\Path 24:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=61
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	432
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c
Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
432.        dsl_deadlist_close(&ds->ds_deadlist);
```

NULL Pointer Dereference\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=62
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_dir at zfs/dsl_destroy.c in line 273.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	276
Object	0	ds_dir

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_handle_remaps(dsl_dataset_t *ds, dsl_dataset_t *ds_next,

```
....
276.        dsl_pool_t *dp = ds->ds_dir->dd_pool;
```

NULL Pointer Dereference\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=63
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 153.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	174
Object	0	ds

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method process_old_deadlist(dsl_dataset_t *ds, dsl_dataset_t *ds_prev,

```
....
174.        dsl_dir_diduse_space(ds->ds_dir, DD_USED_SNAP,
```

NULL Pointer Dereference\Path 27:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=64
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_dir at zfs/dsl_destroy.c in line 153.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	157
Object	0	ds_dir

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method process_old_deadlist(dsl_dataset_t *ds, dsl_dataset_t *ds_prev,

```
....
157.        dsl_pool_t *dp = ds->ds_dir->dd_pool;
```

NULL Pointer Dereference\Path 28:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=65
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 153.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	184
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method process_old_deadlist(dsl_dataset_t *ds, dsl_dataset_t *ds_prev,

```
....
184.        dsl_deadlist_open(&ds->ds_deadlist, mos,
```

NULL Pointer Dereference\Path 29:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=66
Status New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 153.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	178
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method	process_old_deadlist(dsl_dataset_t *ds, dsl_dataset_t *ds_prev,

```
....
178.        dsl_deadlist_close(&ds->ds_deadlist);
```

NULL Pointer Dereference\Path 30:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=67
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_deadlist at zfs/dsl_destroy.c in line 153.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	161
Object	0	ds_deadlist

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method process_old_deadlist(dsl_dataset_t *ds, dsl_dataset_t *ds_prev,

```
....
161.        ASSERT(ds->ds_deadlist.dl_oldfmt);
```

NULL Pointer Dereference\Path 31:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=68
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c

Line	1013	318
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
318.        ASSERT(zfs_refcount_is_zero(&ds->ds_longholds));
```

NULL Pointer Dereference\Path 32:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=69
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	317
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
317.        rrw_exit(&ds->ds_bp_rwlock, FTAG);
```

NULL Pointer Dereference\Path 33:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=70
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	315
Object	0	ds

Code Snippet

File Name zfs/dsl_destroy.c

Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
315.        rrw_enter(&ds->ds_bp_rwlock, RW_READER, FTAG);
```

NULL Pointer Dereference\Path 34:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=71
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c

Line	1013	410
Object	0	ds

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
410.              dsl_dir_diduse_space(ds->ds_dir, DD_USED_SNAP,
```

NULL Pointer Dereference\Path 35:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=72
Status	New

The variable declared in 0 at zfs/dsl_destroy.c in line 1009 is not initialized when it is used by ds_dir at zfs/dsl_destroy.c in line 306.

	Source	Destination
File	zfs/dsl_destroy.c	zfs/dsl_destroy.c
Line	1013	309
Object	0	ds_dir

Code Snippet
File Name zfs/dsl_destroy.c
Method dsl_destroy_head_sync_impl(dsl_dataset_t *ds, dmu_tx_t *tx)

```
....
1013.       uint64_t obj, ddobj, prevobj = 0;
```

File Name zfs/dsl_destroy.c

Method dsl_destroy_snapshot_sync_impl(dsl_dataset_t *ds, boolean_t defer, dmu_tx_t *tx)

```
....
309.        dsl_pool_t *dp = ds->ds_dir->dd_pool;
```

Exposure of System Data to Unauthorized Control Sphere

Query Path:
CPP\Cx\CPP Low Visibility\Exposure of System Data to Unauthorized Control Sphere Version:1

Categories

FISMA 2014: Configuration Management
NIST SP 800-53: AC-3 Access Enforcement (P1)

Description

Exposure of System Data to Unauthorized Control Sphere\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=262
Status	New

The system data read by main in the file zfs/file_append.c at line 133 is potentially exposed by main found in zfs/file_append.c at line 133.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	150	150
Object	perror	perror

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
150.                perror("open");
```

Exposure of System Data to Unauthorized Control Sphere\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=263
Status	New

The system data read by main in the file zfs/file_append.c at line 133 is potentially exposed by main found in zfs/file_append.c at line 133.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	177	177

Object	perror	perror

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
177.                        perror("write");
```

Exposure of System Data to Unauthorized Control Sphere\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=264
Status	New

The system data read by main in the file zfs/file_append.c at line 133 is potentially exposed by main found in zfs/file_append.c at line 133.

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	192	192
Object	perror	perror

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
192.             perror("output seek");
```

Exposure of System Data to Unauthorized Control Sphere\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=265
Status	New

The system data read by drop_caches in the file zfs/xattrtest.c at line 230 is potentially exposed by drop_caches found in zfs/xattrtest.c at line 230.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	237	237
Object	errno	fprintf

Code Snippet	
File Name	zfs/xattrtest.c
Method	drop_caches(void)

```
....
237.                ERROR("Error %d: open(\"%s\", O_WRONLY)\n", errno,
file);
```

Exposure of System Data to Unauthorized Control Sphere\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=266
Status	New

The system data read by drop_caches in the file zfs/xattrtest.c at line 230 is potentially exposed by drop_caches found in zfs/xattrtest.c at line 230.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	243	243
Object	errno	fprintf

Code Snippet	
File Name	zfs/xattrtest.c
Method	drop_caches(void)

```
....
243.                ERROR("Error %d: write(%d, \"3\", 1)\n", errno, fd);
```

Exposure of System Data to Unauthorized Control Sphere\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=267
Status	New

The system data read by drop_caches in the file zfs/xattrtest.c at line 230 is potentially exposed by drop_caches found in zfs/xattrtest.c at line 230.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	250	250
Object	errno	fprintf

Code Snippet	
File Name	zfs/xattrtest.c

Method	drop_caches(void)

```
....
250.              ERROR("Error %d: close(%d)\n", errno, fd);
```

Exposure of System Data to Unauthorized Control Sphere\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=268
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	381	396
Object	errno	fprintf

Code Snippet	
File Name	zfs/xattrtest.c
Method	create_files(void)

```
....
381.                    ERROR("Error %d: unlink(%s)\n", errno, file);
....
396.                    ERROR("Error %d: close(%d)\n", errno, rc);
```

Exposure of System Data to Unauthorized Control Sphere\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=269
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	388	396
Object	errno	fprintf

Code Snippet	
File Name	zfs/xattrtest.c
Method	create_files(void)

```
....
388.                    ERROR("Error %d: open(%s, O_CREATE, 0644)\n",
....
396.                    ERROR("Error %d: close(%d)\n", errno, rc);
```

Exposure of System Data to Unauthorized Control Sphere\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=270
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	390	396
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
390.                    rc = errno;
....
396.                    ERROR("Error %d: close(%d)\n", errno, rc);
```

Exposure of System Data to Unauthorized Control Sphere\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=271
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	396	396
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
396.                    ERROR("Error %d: close(%d)\n", errno, rc);
```

Exposure of System Data to Unauthorized Control Sphere\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=272
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	397	396
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
397.                    rc = errno;
....
396.                    ERROR("Error %d: close(%d)\n", errno, rc);
```

Exposure of System Data to Unauthorized Control Sphere\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=273
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	381	388
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
381.                    ERROR("Error %d: unlink(%s)\n", errno, file);
....
388.                    ERROR("Error %d: open(%s, O_CREATE, 0644)\n",
```

Exposure of System Data to Unauthorized Control Sphere\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=274
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	388	388
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
388.                    ERROR("Error %d: open(%s, O_CREATE, 0644)\n",
```

Exposure of System Data to Unauthorized Control Sphere\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=275
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	396	388
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
396.                    ERROR("Error %d: close(%d)\n", errno, rc);
....
388.                    ERROR("Error %d: open(%s, O_CREATE, 0644)\n",
```

Exposure of System Data to Unauthorized Control Sphere\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=276
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	381	381
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
381.                    ERROR("Error %d: unlink(%s)\n", errno, file);
```

Exposure of System Data to Unauthorized Control Sphere\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=277
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	388	381
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
388.                    ERROR("Error %d: open(%s, O_CREATE, 0644)\n",
....
381.                    ERROR("Error %d: unlink(%s)\n", errno, file);
```

Exposure of System Data to Unauthorized Control Sphere\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=278
Status	New

The system data read by create_files in the file zfs/xattrtest.c at line 350 is potentially exposed by create_files found in zfs/xattrtest.c at line 350.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	396	381
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method create_files(void)

```
....
396.                    ERROR("Error %d: close(%d)\n", errno, rc);
....
381.                    ERROR("Error %d: unlink(%s)\n", errno, file);
```

Exposure of System Data to Unauthorized Control Sphere\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=279
Status	New

The system data read by setxattrs in the file zfs/xattrtest.c at line 439 is potentially exposed by setxattrs found in zfs/xattrtest.c at line 439.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	489	489
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
489.                            ERROR("Error %d: lsetxattr(%s, %s, ...,
%d)\n",
```

Exposure of System Data to Unauthorized Control Sphere\Path 19:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=280
Status	New

The system data read by getxattrs in the file zfs/xattrtest.c at line 513 is potentially exposed by getxattrs found in zfs/xattrtest.c at line 513.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	572	572
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method getxattrs(void)

```
....
572.                            ERROR("Error %d: lgetxattr(%s, %s, ...,
%d)\n",
```

Exposure of System Data to Unauthorized Control Sphere\Path 20:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=281
Status	New

The system data read by unlink_files in the file zfs/xattrtest.c at line 617 is potentially exposed by unlink_files found in zfs/xattrtest.c at line 617.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	648	648
Object	errno	fprintf

Code Snippet
File Name zfs/xattrtest.c
Method unlink_files(void)

```
....
648.                    ERROR("Error %d: unlink(%s)\n", errno, file);
```

Exposure of System Data to Unauthorized Control Sphere\Path 21:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=282
Status	New

The system data read by zhack_repair_read_label in the file zfs/zhack.c at line 507 is potentially exposed by zhack_repair_read_label found in zfs/zhack.c at line 507.

	Source	Destination
File	zfs/zhack.c	zfs/zhack.c
Line	515	513
Object	errno	fprintf

Code Snippet
File Name zfs/zhack.c
Method zhack_repair_read_label(const int fd, vdev_label_t *vl,

```
....
515.                    l, strerror(errno));
....
513.               (void) fprintf(stderr,
```

Exposure of System Data to Unauthorized Control Sphere\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=283
Status	New

The system data read by zhack_repair_write_label in the file zfs/zhack.c at line 648 is potentially exposed by zhack_repair_write_label found in zfs/zhack.c at line 648.

	Source	Destination
File	zfs/zhack.c	zfs/zhack.c
Line	665	664
Object	errno	fprintf

Code Snippet
File Name zfs/zhack.c
Method zhack_repair_write_label(const int l, const int fd, const int byteswap,

```
....
665.                    l, strerror(errno));
....
664.                (void) fprintf(stderr, "error: cannot write label %d:
%s\n",
```

Exposure of System Data to Unauthorized Control Sphere\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=284
Status	New

The system data read by main in the file zfs/zinject.c at line 737 is potentially exposed by main found in zfs/zinject.c at line 737.

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	765	765
Object	errno	fprintf

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

```
....
765.                (void) fprintf(stderr, "%s\n",
libzfs_error_init(errno));
```

Exposure of System Data to Unauthorized Control Sphere\Path 24:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=285
Status	New

The system data read by iter_handlers in the file zfs/zinject.c at line 336 is potentially exposed by iter_handlers found in zfs/zinject.c at line 336.

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	349	348
Object	errno	fprintf

Code Snippet
File Name zfs/zinject.c
Method iter_handlers(int (*func)(int, const char *, zinject_record_t *, void *),

```
....
349.                    strerror(errno));
....
348.                (void) fprintf(stderr, "Unable to list handlers:
%s\n",
```

Exposure of System Data to Unauthorized Control Sphere\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=286
Status	New

The system data read by cancel_one_handler in the file zfs/zinject.c at line 502 is potentially exposed by cancel_one_handler found in zfs/zinject.c at line 502.

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	512	511
Object	errno	fprintf

Code Snippet
File Name zfs/zinject.c
Method cancel_one_handler(int id, const char *pool, zinject_record_t *record,

```
....
512.                    id, strerror(errno));
....
511.                (void) fprintf(stderr, "failed to remove handler %d:
%s\n",
```

Exposure of System Data to Unauthorized Control Sphere\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=287
Status	New

The system data read by cancel_handler in the file zfs/zinject.c at line 537 is potentially exposed by cancel_handler found in zfs/zinject.c at line 537.

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	545	544
Object	errno	fprintf

Code Snippet

File Name	zfs/zinject.c
Method	cancel_handler(int id)

```
....
545.                    id, strerror(errno));
....
544.                (void) fprintf(stderr, "failed to remove handler %d:
%s\n",
```

Exposure of System Data to Unauthorized Control Sphere\Path 27:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=288
Status	New

The system data read by register_handler in the file zfs/zinject.c at line 558 is potentially exposed by register_handler found in zfs/zinject.c at line 558.

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	570	568
Object	errno	fprintf

Code Snippet	
File Name	zfs/zinject.c
Method	register_handler(const char *pool, int flags, zinject_record_t *record,

```
....
570.                    strerror(errno));
....
568.                (void) fprintf(stderr, "failed to add handler: %s\n",
```

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=1020017&projectid=20013&pathid=289
Status	New

The get_spl_hostid method in zfs/gethostid.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	zfs/gethostid.c	zfs/gethostid.c

Line	46	46
Object	fopen	fopen

Code Snippet
File Name zfs/gethostid.c
Method get_spl_hostid(void)

```
....
46.    f = fopen("/proc/sys/kernel/spl/hostid", "re");
```

TOCTOU\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=290
Status	New

The main method in zfs/file_append.c file utilizes open 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	zfs/file_append.c	zfs/file_append.c
Line	147	147
Object	open	open

Code Snippet
File Name zfs/file_append.c
Method main(int argc, char *argv[])

```
....
147.        fd = open(filename, fd_flags, 0666);
```

TOCTOU\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=291
Status	New

The get_system_hostid method in zfs/gethostid.c file utilizes open 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	zfs/gethostid.c	zfs/gethostid.c
Line	71	71
Object	open	open

Code Snippet	
File Name	zfs/gethostid.c
Method	get_system_hostid(void)

```
....
71.         int fd = open("/etc/hostid", O_RDONLY | O_CLOEXEC);
```

TOCTOU\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=292
Status	New

The drop_caches method in zfs/xattrtest.c file utilizes open 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	zfs/xattrtest.c	zfs/xattrtest.c
Line	235	235
Object	open	open

Code Snippet	
File Name	zfs/xattrtest.c
Method	drop_caches(void)

```
....
235.        fd = open(file, O_WRONLY);
```

TOCTOU\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=293
Status	New

The run_process method in zfs/xattrtest.c file utilizes open 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	zfs/xattrtest.c	zfs/xattrtest.c
Line	265	265
Object	open	open

Code Snippet	
File Name	zfs/xattrtest.c

Method	run_process(const char *path, char *argv[])

```
....
265.              devnull_fd = open("/dev/null", O_WRONLY);
```

TOCTOU\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=294
Status	New

The create_files method in zfs/xattrtest.c file utilizes open 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	zfs/xattrtest.c	zfs/xattrtest.c
Line	386	386
Object	open	open

Code Snippet	
File Name	zfs/xattrtest.c
Method	create_files(void)

```
....
386.              rc = open(file, O_CREAT, 0644);
```

TOCTOU\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=295
Status	New

The get_random_bytes method in zfs/xattrtest.c file utilizes open 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	zfs/xattrtest.c	zfs/xattrtest.c
Line	421	421
Object	open	open

Code Snippet	
File Name	zfs/xattrtest.c
Method	get_random_bytes(char *buf, size_t bytes)

```
....
421.        rand = open("/dev/urandom", O_RDONLY);
```

TOCTOU\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=296
Status	New

The zhack_label_repair method in zfs/zhack.c file utilizes open 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	zfs/zhack.c	zfs/zhack.c
Line	870	870
Object	open	open

Code Snippet
File Name zfs/zhack.c
Method zhack_label_repair(const zhack_repair_op_t op, const int argc, char **argv)

```
....
870.        if ((fd = open(argv[0], O_RDWR)) == -1)
```

TOCTOU\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=297
Status	New

The main method in zfs/zinject.c file utilizes open 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	zfs/zinject.c	zfs/zinject.c
Line	771	771
Object	open	open

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

```
....
771.        if ((zfs_fd = open(ZFS_DEV, O_RDWR)) < 0) {
```

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=1020017&projectid=20013&pathid=5
Status	New

	Source	Destination
File	zfs/file_append.c	zfs/file_append.c
Line	79	79
Object	getopt	getopt

Code Snippet
File Name zfs/file_append.c
Method parse_options(int argc, char *argv[])

```
....
79.    while ((c = getopt(argc, argv, "b:de:f:hn:")) != -1) {
```

Inconsistent Implementations\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=6
Status	New

	Source	Destination
File	zfs/zhack.c	zfs/zhack.c
Line	982	982
Object	getopt	getopt

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

```
....
982.        while ((c = getopt(argc, argv, "+c:d:")) != -1) {
```

Inconsistent Implementations\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=7
Status	New

	Source	Destination
File	zfs/zhack.c	zfs/zhack.c
Line	306	306
Object	getopt	getopt

Code Snippet
File Name zfs/zhack.c
Method zhack_do_feature_enable(int argc, char **argv)

```
....
306.         while ((c = getopt(argc, argv, "+rd:")) != -1) {
```

Inconsistent Implementations\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=8
Status	New

	Source	Destination
File	zfs/zhack.c	zfs/zhack.c
Line	407	407
Object	getopt	getopt

Code Snippet
File Name zfs/zhack.c
Method zhack_do_feature_ref(int argc, char **argv)

```
....
407.         while ((c = getopt(argc, argv, "+md")) != -1) {
```

Inconsistent Implementations\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=9
Status	New

	Source	Destination
File	zfs/zhack.c	zfs/zhack.c
Line	918	918
Object	getopt	getopt

Code Snippet
File Name zfs/zhack.c
Method zhack_do_label_repair(int argc, char **argv)

```
....
918.         while ((c = getopt(argc, argv, "+cu")) != -1) {
```

Inconsistent Implementations\Path 6:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=10
Status New

	Source	Destination
File	zfs/zinject.c	zfs/zinject.c
Line	792	792
Object	getopt	getopt

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

```
....
792.         while ((c = getopt(argc, argv,
```

Inconsistent Implementations\Path 7:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=11
Status New

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	136	136
Object	getopt_long	getopt_long

Code Snippet

File Name	zfs/xattrtest.c
Method	parse_args(int argc, char **argv)

```
....
136.        while ((c = getopt_long(argc, argv, shortopts, longopts,
NULL)) != -1) {
```

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=1020017&projectid=20013&pathid=15
Status	New

The setxattrs method calls the sprintf function, at line 439 of zfs/xattrtest.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	zfs/xattrtest.c	zfs/xattrtest.c
Line	482	482
Object	sprintf	sprintf

Code Snippet	
File Name	zfs/xattrtest.c
Method	setxattrs(void)

```
....
482.                    (void) sprintf(name, "user.%d", j);
```

Unchecked Return Value\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=16
Status	New

The getxattrs method calls the sprintf function, at line 513 of zfs/xattrtest.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	zfs/xattrtest.c	zfs/xattrtest.c
Line	568	568
Object	sprintf	sprintf

Code Snippet
File Name zfs/xattrtest.c
Method getxattrs(void)

```
....
568.                    (void) sprintf(name, "user.%d", j);
```

Unchecked Return Value\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=17
Status	New

The setxattrs method calls the shift function, at line 439 of zfs/xattrtest.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	zfs/xattrtest.c	zfs/xattrtest.c
Line	483	483
Object	shift	shift

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
483.                    shift = sprintf(value, "size=%d ", rnd_size);
```

Unchecked Return Value\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=18
Status	New

The getxattrs method calls the shift function, at line 513 of zfs/xattrtest.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	zfs/xattrtest.c	zfs/xattrtest.c
Line	581	581

Object	shift	shift

Code Snippet
File Name zfs/xattrtest.c
Method getxattrs(void)

```
....
581.                    shift = sprintf(verify_value, "size=%d ",
```

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=1020017&projectid=20013&pathid=19
Status New

	Source	Destination
File	zfs/lgc.c	zfs/lgc.c
Line	469	469
Object	sizeof	sizeof

Code Snippet
File Name zfs/lgc.c
Method static lu_mem traversetable (global_State *g, Table *h) {

```
....
469.                            sizeof(Proto *) * f->sizep +
```

Use of Sizeof On a Pointer Type\Path 2:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=20
Status New

	Source	Destination
File	zfs/lgc.c	zfs/lgc.c
Line	1043	1043
Object	sizeof	sizeof

Code Snippet
File Name zfs/lgc.c

Method	static lu_mem singlestep (lua_State *L) {

```
....
1043.        g->GCmemtrav = g->strt.size * sizeof(GCObject*);
```

Use of Sizeof On a Pointer Type\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=21
Status	New

	Source	Destination
File	zfs/lobject.c	zfs/lobject.c
Line	202	202
Object	sizeof	sizeof

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

```
....
202.          char buff[4*sizeof(void *) + 8]; /* should be enough space
for a `%p' */
```

Heuristic 2nd Order Buffer Overflow read

Query Path:
CPP\Cx\CPP Heuristic\Heuristic 2nd Order Buffer Overflow read Version:0

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
NIST SP 800-53: SI-10 Information Input Validation (P1)
OWASP Top 10 2017: A1-Injection

Description

Heuristic 2nd Order Buffer Overflow read\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=73
Status	New

The size of the buffer used by get_random_bytes in BinaryExpr, at line 416 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that get_random_bytes passes to BinaryExpr, at line 416 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	427

Object	BinaryExpr	BinaryExpr

Code Snippet
File Name zfs/xattrtest.c
Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

Heuristic 2nd Order Buffer Overflow read\Path 2:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=74
Status New

The size of the buffer used by get_random_bytes in bytes, at line 416 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that get_random_bytes passes to BinaryExpr, at line 416 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	427
Object	BinaryExpr	bytes

Code Snippet
File Name zfs/xattrtest.c
Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

Heuristic 2nd Order Buffer Overflow read\Path 3:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=75
Status New

The size of the buffer used by get_random_bytes in bytes_read, at line 416 of zfs/xattrtest.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that get_random_bytes passes to BinaryExpr, at line 416 of zfs/xattrtest.c, to overwrite the target buffer.

	Source	Destination
File	zfs/xattrtest.c	zfs/xattrtest.c
Line	427	427

Object	BinaryExpr	bytes_read

Code Snippet
File Name zfs/xattrtest.c
Method get_random_bytes(char *buf, size_t bytes)

```
....
427.              ssize_t rc = read(rand, buf + bytes_read, bytes -
bytes_read);
```

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=1020017&projectid=20013&pathid=12
Status	New

Method setxattrs at line 439 of zfs/xattrtest.c uses a weak method random 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	zfs/xattrtest.c	zfs/xattrtest.c
Line	480	480
Object	random	random

Code Snippet
File Name zfs/xattrtest.c
Method setxattrs(void)

```
....
480.                          rnd_size = (random() % (size - 16)) + 16;
```

Use of Insufficiently Random Values\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=13
Status	New

Method parse_args at line 130 of zfs/xattrtest.c uses a weak method srandom 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	zfs/xattrtest.c	zfs/xattrtest.c
Line	205	205
Object	srandom	srandom

Code Snippet
File Name zfs/xattrtest.c
Method parse_args(int argc, char **argv)

```
....
205.        srandom(seed);
```

Unchecked Array Index

Query Path:
CPP\Cx\CPP Low Visibility\Unchecked Array Index Version:1

Categories

NIST SP 800-53: SI-10 Information Input Validation (P1)

Description

Unchecked Array Index\Path 1:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=78
Status New

	Source	Destination
File	zfs/lz4_zfs.c	zfs/lz4_zfs.c
Line	523	523
Object	h	h

Code Snippet
File Name zfs/lz4_zfs.c
Method LZ4_compressCtx(void *ctx, const char *source, char *dest, int isize,

```
....
523.                    HashTable[h] = ip - base;
```

Unchecked Array Index\Path 2:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=79

Status	New

	Source	Destination
File	zfs/lz4_zfs.c	zfs/lz4_zfs.c
Line	711	711
Object	h	h

Code Snippet
File Name zfs/lz4_zfs.c
Method LZ4_compress64kCtx(void *ctx, const char *source, char *dest, int isize,

```
....
711.                    HashTable[h] = ip - base;
```

Potential Path Traversal

Query Path:
CPP\Cx\CPP Low Visibility\Potential Path Traversal Version:0

Categories

OWASP Top 10 2013: A4-Insecure Direct Object References
OWASP Top 10 2017: A5-Broken Access Control

Description

Potential Path Traversal\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1020017&projectid=20013&pathid=14
Status	New

Method main at line 970 of zfs/zhack.c gets user input from the argv element. This element's value then flows through the code and is eventually used in a file path for local disk access in zhack_label_repair at line 854 of zfs/zhack.c. This may cause a Path Traversal vulnerability.

	Source	Destination
File	zfs/zhack.c	zfs/zhack.c
Line	970	870
Object	argv	argv

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

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

File Name zfs/zhack.c

Method	zhack_label_repair(const zhack_repair_op_t op, const int argc, char **argv)

```
....
870.        if ((fd = open(argv[0], O_RDWR)) == -1)
```

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=1020017&projectid=20013&pathid=261
Status	New

	Source	Destination
File	zfs/gethostid.c	zfs/gethostid.c
Line	46	46
Object	f	f

Code Snippet	
File Name	zfs/gethostid.c
Method	get_spl_hostid(void)

```
....
46.    f = fopen("/proc/sys/kernel/spl/hostid", "re");
```

Buffer Overflow LongString

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

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;
}
```

Wrong Size t Allocation

Risk

What might happen

Incorrect allocation of memory may result in unexpected behavior by either overwriting sections of memory with unexpected values. Under certain conditions where both an incorrect allocation of memory and the values being written can be controlled by an attacker, such an issue may result in execution of malicious code.

Cause

How does it happen

Some memory allocation functions require a size value to be provided as a parameter. The allocated size should be derived from the provided value, by providing the length value of the intended source, multiplied by the size of that length. Failure to perform the correct arithmetic to obtain the exact size of the value will likely result in the source overflowing its destination.

General Recommendations

How to avoid it

- Always perform the correct arithmetic to determine size.
- Specifically for memory allocation, calculate the allocation size from the allocation source:
    - Derive the size value from the length of intended source to determine the amount of units to be processed.
    - Always programmatically consider the size of the each unit and their conversion to memory units - for example, by using sizeof() on the unit's type.
    - Memory allocation should be a multiplication of the amount of units being written, times the size of each unit.

Source Code Examples

CPP

Allocating and Assigning Memory without Sizeof Arithmetic

```
int *ptr;
ptr = (int*)malloc(5);
for (int i = 0; i < 5; i++)
{
      ptr[i] = i * 2 + 1;
}
```

Allocating and Assigning Memory with Sizeof Arithmetic

```
int *ptr;
ptr = (int*)malloc(5 * sizeof(int));
```

```
for (int i = 0; i < 5; i++)
{
      ptr[i] = i * 2 + 1;
}
```

Incorrect Arithmetic of Multi-Byte String Allocation

```
wchar_t * dest;
dest = (wchar_t *)malloc(wcslen(source) + 1); // Would not crash for a short "source"
wcscpy((wchar_t *)dest, source);
wprintf(L"Dest: %s\r\n", dest);
```

Correct Arithmetic of Multi-Byte String Allocation

```
wchar_t * dest;
dest = (wchar_t *)malloc((wcslen(source) + 1) * sizeof(wchar_t));
wcscpy((wchar_t *)dest, source);
wprintf(L"Dest: %s\r\n", dest);
```

Integer Overflow

Risk

What might happen

Assigning large data types into smaller data types, without proper checks and explicit casting, will lead to undefined behavior and unintentional effects, such as data corruption (e.g. value wraparound, wherein maximum values become minimum values); system crashes; infinite loops; logic errors, such as bypassing of security mechanisms; or even buffer overflows leading to arbitrary code execution.

Cause

How does it happen

This flaw can occur when implicitly casting numerical data types of a larger size, into a variable with a data type of a smaller size. This forces the program to discard some bits of information from the number. Depending on how the numerical data types are stored in memory, this is often the bits with the highest value, causing substantial corruption of the stored number. Alternatively, the sign bit of a signed integer could be lost, completely reversing the intention of the number.

General Recommendations

How to avoid it

- Avoid casting larger data types to smaller types.
- Prefer promoting the target variable to a large enough data type.
- If downcasting is necessary, always check that values are valid and in range of the target type, before casting

---

Source Code Examples

CPP

Unsafe Downsize Casting

```
int unsafe_addition(short op1, int op2) {

    // op2 gets forced from int into a short
    short total = op1 + op2;

    return total;
}
```

Safer Use of Proper Data Types

```
int safe_addition(short op1, int op2) {

    // total variable is of type int, the largest type that is needed
    int total = 0;

    // check if total will overflow available integer size
    if (INT_MAX - abs(op2) > op1)
```

```
    {
        total = op1 + op2;
    }
    else
    {
        // instead of overflow, saturate (but this is not always a good thing)
        total = INT_MAX
    }

    return total;
}
```

Dangerous Functions

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

Source Code Examples

CPP

Buffer Overflow in gets()

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

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

Safe reading from user

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

	printf("Please enter your name: ");
	fgets(buf, sizeof(buf), stdin); //setting the amount of bytes to read
	if (buf == ACCEPTED_NAME)
	{
		//Do something
	}
	return 0;
}
```

Unsafe function for string copy

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

	return 0;
}
```

Safe string copy

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

	return 0;
}
```

Unsafe format string

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

Safe format string

```
int main(int argc, char* argv[])
{
      printf("%s", argv[1]); // Second parameter is not a formattable string
      return 0;
}
```

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

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

Description

Description Summary

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

Extended Description

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

Terminology Notes

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

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C

C++

Modes of Introduction

Memory leaks have two common and sometimes overlapping causes:

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

Common Consequences

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

Likelihood of Exploit

Medium

Demonstrative Examples

Example 1

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

(Bad Code)

Example Language: **C**

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

return NULL;
}
```

```
return buf;
}
```

Example 2

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

(Bad Code)

Example Language: **C**

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

free(foo);
}
int main() {

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

Observed Examples

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

Potential Mitigations

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

Phase: Architecture and Design

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

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

Relationships

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

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

Relationship Notes

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

Affected Resources

- Memory

Functional Areas

- Memory management

Taxonomy Mappings

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

White Box Definitions

A weakness where the code path has:

1. start statement that allocates dynamically allocated memory resource

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

Where "loses" is defined through the following scenarios:

1. identity of the dynamic allocated memory resource never obtained

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

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

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

References

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

Content History

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

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

Previous Entry Names

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

BACK TO TOP

Use of Zero Initialized Pointer

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

Source Code Examples

CPP

Explicit NULL Dereference

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

Implicit NULL Dereference

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

Java

Explicit Null Dereference

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

Stored Buffer Overflow boundcpy

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

CPP

Overflowing Buffers

```
const int BUFFER_SIZE = 10;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
{
    strcpy(buffer, inputString);
}
```

Checked Buffers

```
const int BUFFER_SIZE = 10;
const int MAX_INPUT_SIZE = 256;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
```

```
{
    if (strnlen(inputString, MAX_INPUT_SIZE) < sizeof(buffer))
    {
        strncpy(buffer, inputString, sizeof(buffer));
    }
}
```

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)
```

Potential Path Traversal

Risk

What might happen

An attacker could define any arbitrary file path for the application to use, potentially leading to:

- Stealing sensitive files, such as configuration or system files
- Overwriting files such as program binaries, configuration files, or system files
- Deleting critical files, causing a denial of service (DoS).

Cause

How does it happen

The application uses user input in the file path for accessing files on the application server's local disk. This enables an attacker to arbitrarily determine the file path.

General Recommendations

How to avoid it

1. Ideally, avoid depending on user input for file selection.
2. Validate all input, regardless of source. Validation should be based on a whitelist: accept only data fitting a specified structure, rather than reject bad patterns. Check for:
   - Data type
   - Size
   - Range
   - Format
   - Expected values
3. Accept user input only for the filename, not for the path and folders.
4. Ensure that file path is fully canonicalized.
5. Explicitly limit the application to using a designated folder that separate from the applications binary folder.
6. Restrict the privileges of the application's OS user to necessary files and folders. The application should not be able to write to the application binary folder, and should not read anything outside of the application folder and data folder.

Source Code Examples

CSharp

Using unvalidated user input as the file name may enable the user to access arbitrary files on the server local disk

```
public class PathTraversal
{
        private void foo(TextBox textbox1)
    {
                string fileNum = textbox1.Text;
                string path = "c:\files\file" + fileNum;
                FileStream f = new FileStream(path, FileMode.Open);
                byte[] output = new byte[10];
                f.Read(output,0, 10);
```

```
        }
}
```

Potentially hazardous characters are removed from the user input before use

```
public class PathTraversalFixed
{
        private void foo(TextBox textbox1)
    {
                string fileNum = textbox1.Text.Replace("\", "").Replace("..", "");
         string path = "c:\files\file" + fileNum;
                FileStream f = new FileStream(path, FileMode.Open);
                byte[] output = new byte[10];
                f.Read(output,0, 10);
        }
}
```

Java

Using unvalidated user input as the file name may enable the user to access arbitrary files on the server local disk

```
public class Absolute_Path_Traversal {
    public static void main(String[] args) {
        Scanner userInputScanner = new Scanner(System.in);
        System.out.print("\nEnter file name: ");
        String name = userInputScanner.nextLine();
        String path = "c:\files\file" + name;
        try {
            BufferedReader reader = new BufferedReader(new FileReader(path));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Potentially hazardous characters are removed from the user input before use

```
public class Absolute_Path_Traversal_Fixed {
    public static void main(String[] args) {
        Scanner userInputScanner = new Scanner(System.in);
        System.out.print("\nEnter file name: ");
        String name = userInputScanner.nextLine();
        name = name.replace("/", "").replace("..", "");
        String path = "c:\files\file" + name;
        try {
            BufferedReader reader = new BufferedReader(new FileReader(path));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Unchecked Return Value

Risk

What might happen

A program that does not check function return values could cause the application to enter an undefined state. This could lead to unexpected behavior and unintended consequences, including inconsistent data, system crashes or other error-based exploits.

Cause

How does it happen

The application calls a system function, but does not receive or check the result of this funciton. These functions often return error codes in the result, or share other status codes with it's caller. The application simply ignores this result value, losing this vital information.

General Recommendations

How to avoid it

- Always check the result of any called function that returns a value, and verify the result is an expected value.

- Ensure the calling function responds to all possible return values.

- Expect runtime errors and handle them gracefully. Explicitly define a mechanism for handling unexpected errors.

Source Code Examples

CPP

Unchecked Memory Allocation

```
buff = (char*) malloc(size);
strncpy(buff, source, size);
```

Safer Memory Allocation

```
buff = (char*) malloc(size+1);
if (buff==NULL) exit(1);

strncpy(buff, source, size);
buff[size] = '\0';
```

Use of sizeof() on a Pointer Type

Weakness ID: 467 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The code calls sizeof() on a malloced pointer type, which always returns the wordsize/8. This can produce an unexpected result if the programmer intended to determine how much memory has been allocated.

Time of Introduction

- Implementation

Applicable Platforms

Languages

C

C++

Common Consequences

Scope	Effect
Integrity	This error can often cause one to allocate a buffer that is much smaller than what is needed, leading to resultant weaknesses such as buffer overflows.

Likelihood of Exploit

High

Demonstrative Examples

Example 1

Care should be taken to ensure sizeof returns the size of the data structure itself, and not the size of the pointer to the data structure.

In this example, sizeof(foo) returns the size of the pointer.

(Bad Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(foo));
```

In this example, sizeof(*foo) returns the size of the data structure and not the size of the pointer.

(Good Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(*foo));
```

Example 2

This example defines a fixed username and password. The AuthenticateUser() function is intended to accept a username and a password from an untrusted user, and check to ensure that it matches the username and password. If the username and password match, AuthenticateUser() is intended to indicate that authentication succeeded.

(Bad Code)

```
/* Ignore CWE-259 (hard-coded password) and CWE-309 (use of password system for authentication) for this example. */

char *username = "admin";
char *pass = "password";

int AuthenticateUser(char *inUser, char *inPass) {
```

```
printf("Sizeof username = %d\n", sizeof(username));
printf("Sizeof pass = %d\n", sizeof(pass));

if (strncmp(username, inUser, sizeof(username))) {
printf("Auth failure of username using sizeof\n");
return(AUTH_FAIL);
}
/* Because of CWE-467, the sizeof returns 4 on many platforms and architectures. */
if (! strncmp(pass, inPass, sizeof(pass))) {
printf("Auth success of password using sizeof\n");
return(AUTH_SUCCESS);
}
else {
printf("Auth fail of password using sizeof\n");
return(AUTH_FAIL);
}
}

int main (int argc, char **argv)
{
int authResult;

if (argc < 3) {
ExitError("Usage: Provide a username and password");
}
authResult = AuthenticateUser(argv[1], argv[2]);
if (authResult != AUTH_SUCCESS) {
ExitError("Authentication failed");
}
else {
DoAuthenticatedTask(argv[1]);
}
}
```

In AuthenticateUser(), because sizeof() is applied to a parameter with an array type, the sizeof() call might return 4 on many modern architectures. As a result, the strncmp() call only checks the first four characters of the input password, resulting in a partial comparison (CWE-187), leading to improper authentication (CWE-287).

Because of the partial comparison, any of these passwords would still cause authentication to succeed for the "admin" user:

(Attack)

```
pass5
passABCDEFGH
passWORD
```

Because only 4 characters are checked, this significantly reduces the search space for an attacker, making brute force attacks more feasible.

The same problem also applies to the username, so values such as "adminXYZ" and "administrator" will succeed for the username.

Potential Mitigations

Phase: Implementation

Use expressions such as "sizeof(*pointer)" instead of "sizeof(pointer)", unless you intend to run sizeof() on a pointer type to gain some platform independence or if you are allocating a variable on the stack.

Other Notes

The use of sizeof() on a pointer can sometimes generate useful information. An obvious case is to find out the wordsize on a platform. More often than not, the appearance of sizeof(pointer) indicates a bug.

Weakness Ordinalities

Ordinality	Description
Primary	*(where the weakness exists independent of other weaknesses)*

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	465	Pointer Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	682	Incorrect Calculation	**Research Concepts (primary)1000**
ChildOf	Category	737	CERT C Secure Coding Section 03 - Expressions (EXP)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
CanPrecede	Weakness Base	131	Incorrect Calculation of Buffer Size	Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Use of sizeof() on a pointer type
CERT C Secure Coding	ARR01-C		Do not apply the sizeof operator to a pointer when taking the size of an array
CERT C Secure Coding	EXP01-C		Do not take the size of a pointer to determine the size of the pointed-to type

White Box Definitions

A weakness where code path has:

1. end statement that passes an identity of a dynamically allocated memory resource to a sizeof operator
2. start statement that allocates the dynamically allocated memory resource

References

Robert Seacord. "EXP01-A. Do not take the sizeof a pointer to determine the size of a type". <https://www.securecoding.cert.org/confluence/display/seccode/EXP01-A.+Do+not+take+the+sizeof+a+pointer+to+determine+the+size+of+a+type>.

Content History

Submissions

Submission Date	Submitter	Organization	Source
	CLASP		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-01		KDM Analytics	External
	added/updated white box definitions		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

BACK TO TOP

NULL Pointer Dereference

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

Source Code Examples

Heuristic 2nd Order Buffer Overflow read

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Improper Validation of Array Index

Weakness ID: 129 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The product uses untrusted input when calculating or using an array index, but the product does not validate or incorrectly validates the index to ensure the index references a valid position within the array.

Alternate Terms

out-of-bounds array index

index-out-of-range

array index underflow

Time of Introduction

- Implementation

Applicable Platforms

Languages

C: *(Often)*

C++: *(Often)*

Language-independent

Common Consequences

Scope	Effect
Integrity Availability	Unchecked array indexing will very likely result in the corruption of relevant memory and perhaps instructions, leading to a crash, if the values are outside of the valid memory area.
Integrity	If the memory corrupted is data, rather than instructions, the system will continue to function with improper values.
Confidentiality Integrity	Unchecked array indexing can also trigger out-of-bounds read or write operations, or operations on the wrong objects; i.e., "buffer overflows" are not always the result. This may result in the exposure or modification of sensitive data.
Integrity	If the memory accessible by the attacker can be effectively controlled, it may be possible to execute arbitrary code, as with a standard buffer overflow and possibly without the use of large inputs if a precise index can be controlled.
Integrity Availability Confidentiality	A single fault could allow either an overflow (CWE-788) or underflow (CWE-786) of the array index. What happens next will depend on the type of operation being performed out of bounds, but can expose sensitive information, cause a system crash, or possibly lead to arbitrary code execution.

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

This weakness can often be detected using automated static analysis tools. Many modern tools use data flow analysis or constraint-based techniques to minimize the number of false positives.

Automated static analysis generally does not account for environmental considerations when reporting out-of-bounds memory operations. This can make it difficult for users to determine which warnings should be investigated first. For example, an analysis tool might report array index errors that originate from command line arguments in a program that is not expected to run with setuid or other special privileges.

Effectiveness: High

This is not a perfect solution, since 100% accuracy and coverage are not feasible.

Automated Dynamic Analysis

This weakness can be detected using dynamic tools and techniques that interact with the software using large test suites with many diverse inputs, such as fuzz testing (fuzzing), robustness testing, and fault injection. The software's operation may slow down, but it should not become unstable, crash, or generate incorrect results.

Black Box

Black box methods might not get the needed code coverage within limited time constraints, and a dynamic test might not produce any noticeable side effects even if it is successful.

Demonstrative Examples

Example 1

The following C/C++ example retrieves the sizes of messages for a pop3 mail server. The message sizes are retrieved from a socket that returns in a buffer the message number and the message size, the message number (num) and size (size) are extracted from the buffer and the message size is placed into an array using the message number for the array index.

(Bad Code)

Example Language: **C**

```
/* capture the sizes of all messages */
int getsizes(int sock, int count, int *sizes) {
...
char buf[BUFFER_SIZE];
int ok;
int num, size;

// read values from socket and added to sizes array
while ((ok = gen_recv(sock, buf, sizeof(buf))) == 0)
{

// continue read from socket until buf only contains '.'
if (DOTLINE(buf))
break;
else if (sscanf(buf, "%d %d", &num, &size) == 2)
sizes[num - 1] = size;
}
...
}
```

In this example the message number retrieved from the buffer could be a value that is outside the allowable range of indices for the array and could possibly be a negative number. Without proper validation of the value to be used for the array index an array overflow could occur and could potentially lead to unauthorized access to memory addresses and system crashes. The value of the array index should be validated to ensure that it is within the allowable range of indices for the array as in the following code.

(Good Code)

Example Language: **C**

```
/* capture the sizes of all messages */
int getsizes(int sock, int count, int *sizes) {
...
char buf[BUFFER_SIZE];
int ok;
int num, size;

// read values from socket and added to sizes array
while ((ok = gen_recv(sock, buf, sizeof(buf))) == 0)
{

// continue read from socket until buf only contains '.'
if (DOTLINE(buf))
```

```
break;
else if (sscanf(buf, "%d %d", &num, &size) == 2) {
if (num > 0 && num <= (unsigned)count)
sizes[num - 1] = size;
else
/* warn about possible attempt to induce buffer overflow */
report(stderr, "Warning: ignoring bogus data for message sizes returned by server.\n");
}
}
...
}
```

Example 2

In the code snippet below, an unchecked integer value is used to reference an object in an array.

(Bad Code)

Example Language: **Java**

```
public String getValue(int index) {
return array[index];
}
```

If index is outside of the range of the array, this may result in an ArrayIndexOutOfBounds Exception being raised.

Example 3

In the following Java example the method displayProductSummary is called from a Web service servlet to retrieve product summary information for display to the user. The servlet obtains the integer value of the product number from the user and passes it to the displayProductSummary method. The displayProductSummary method passes the integer value of the product number to the getProductSummary method which obtains the product summary from the array object containing the project summaries using the integer value of the product number as the array index.

(Bad Code)

Example Language: **Java**

```
// Method called from servlet to obtain product information
public String displayProductSummary(int index) {

String productSummary = new String("");

try {
String productSummary = getProductSummary(index);

} catch (Exception ex) {...}

return productSummary;
}

public String getProductSummary(int index) {
return products[index];
}
```

In this example the integer value used as the array index that is provided by the user may be outside the allowable range of indices for the array which may provide unexpected results or may comes the application to fail. The integer value used for the array index should be validated to ensure that it is within the allowable range of indices for the array as in the following code.

(Good Code)

Example Language: **Java**

```
// Method called from servlet to obtain product information
public String displayProductSummary(int index) {

String productSummary = new String("");
```

```
try {
String productSummary = getProductSummary(index);

} catch (Exception ex) {...}

return productSummary;
}

public String getProductSummary(int index) {
String productSummary = "";

if ((index >= 0) && (index < MAX_PRODUCTS)) {
productSummary = products[index];
}
else {
System.err.println("index is out of bounds");
throw new IndexOutOfBoundsException();
}

return productSummary;
}
```

An alternative in Java would be to use one of the collection objects such as ArrayList that will automatically generate an exception if an attempt is made to access an array index that is out of bounds.

(Good Code)

Example Language: **Java**

```
ArrayList productArray = new ArrayList(MAX_PRODUCTS);
...
try {
productSummary = (String) productArray.get(index);
} catch (IndexOutOfBoundsException ex) {...}
```

Observed Examples

Reference	Description
CVE-2005-0369	large ID in packet used as array index
CVE-2001-1009	negative array index as argument to POP LIST command
CVE-2003-0721	Integer signedness error leads to negative array index
CVE-2004-1189	product does not properly track a count and a maximum number, which can lead to resultant array index overflow.
CVE-2007-5756	chain: device driver for packet-capturing software allows access to an unintended IOCTL with resultant array index error.

Potential Mitigations

Phase: Architecture and Design

Strategies: Input Validation; Libraries or Frameworks

Use an input validation framework such as Struts or the OWASP ESAPI Validation API. If you use Struts, be mindful of weaknesses covered by the CWE-101 category.

Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.

Even though client-side checks provide minimal benefits with respect to server-side security, they are still useful. First, they can support intrusion detection. If the server receives input that should have been rejected by the client, then it may be an indication of an attack. Second, client-side error-checking can provide helpful feedback to the user about the expectations for valid input. Third, there may be a reduction in server-side processing time for accidental input errors, although this is typically a small savings.

Phase: Requirements

Strategy: Language Selection

Use a language with features that can automatically mitigate or eliminate out-of-bounds indexing errors.

For example, Ada allows the programmer to constrain the values of a variable and languages such as Java and Ruby will allow the programmer to handle exceptions when an out-of-bounds index is accessed.

Phase: Implementation

Strategy: Input Validation

Assume all input is malicious. Use an "accept known good" input validation strategy (i.e., use a whitelist). Reject any input that does not strictly conform to specifications, or transform it into something that does. Use a blacklist to reject any unexpected inputs and detect potential attacks.

When accessing a user-controlled array index, use a stringent range of values that are within the target array. Make sure that you do not allow negative values to be used. That is, verify the minimum as well as the maximum of the range of acceptable values.

Phase: Implementation

Be especially careful to validate your input when you invoke code that crosses language boundaries, such as from an interpreted language to native code. This could create an unexpected interaction between the language boundaries. Ensure that you are not violating any of the expectations of the language with which you are interfacing. For example, even though Java may not be susceptible to buffer overflows, providing a large argument in a call to native code might trigger an overflow.

Weakness Ordinalities

Ordinality	Description
Resultant	The most common condition situation leading to unchecked array indexing is the use of loop index variables as buffer indexes. If the end condition for the loop is subject to a flaw, the index can grow or shrink unbounded, therefore causing a buffer overflow or underflow. Another common situation leading to this condition is the use of a function's return value, or the resulting value of a calculation directly as an index in to a buffer.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	20	Improper Input Validation	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	189	Numeric Errors	Development Concepts699
ChildOf	Category	633	Weaknesses that Affect Memory	**Resource-specific Weaknesses (primary)631**
ChildOf	Category	738	CERT C Secure Coding Section 04 - Integers (INT)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
ChildOf	Category	802	2010 Top 25 - Risky Resource Management	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
CanPrecede	Weakness Class	119	Failure to Constrain Operations within the Bounds of a Memory Buffer	Research Concepts1000
CanPrecede	Weakness Variant	789	Uncontrolled Memory Allocation	Research Concepts1000
PeerOf	Weakness Base	124	Buffer Underwrite ('Buffer Underflow')	Research Concepts1000

Theoretical Notes

An improperly validated array index might lead directly to the always-incorrect behavior of "access of array using out-of-bounds index."

Affected Resources

- Memory

f Causal Nature

Explicit

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Unchecked array indexing
PLOVER			INDEX - Array index overflow
CERT C Secure Coding	ARR00-C		Understand how arrays work
CERT C Secure Coding	ARR30-C		Guarantee that array indices are within the valid range
CERT C Secure Coding	ARR38-C		Do not add or subtract an integer to a pointer if the resulting value does not refer to a valid array element
CERT C Secure Coding	INT32-C		Ensure that operations on signed integers do not result in overflow

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
100	Overflow Buffers	

References

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 5, "Array Indexing Errors" Page 144. 2nd Edition. Microsoft. 2002.

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	CLASP		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Sean Eidemiller	Cigital	External
	added/updated demonstrative examples		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Common Consequences		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Description, Name, Relationships		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Observed Examples, Other Notes, Potential Mitigations, Theoretical Notes, Weakness Ordinalities		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Demonstrative Examples, Detection Factors, Likelihood of Exploit, Potential Mitigations, References, Related Attack Patterns, Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Related Attack Patterns		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2009-10-29	Unchecked Array Indexing		

BACK TO TOP

Improper Access Control (Authorization)

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

Description

Description Summary

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

Extended Description

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

Alternate Terms

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

Time of Introduction

- Architecture and Design
- Implementation
- Operation

Applicable Platforms

Languages

Language-independent

Technology Classes

Web-Server: *(Often)*

Database-Server: *(Often)*

Modes of Introduction

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

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

Common Consequences

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

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

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

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

Effectiveness: Limited

Automated Dynamic Analysis

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

Manual Analysis

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

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

Effectiveness: Moderate

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

Demonstrative Examples

Example 1

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

(Bad Code)

Example Language: **Perl**

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

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

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

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

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

Observed Examples

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

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

Potential Mitigations

Phase: Architecture and Design

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

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

Phase: Architecture and Design

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

Phase: Architecture and Design

Strategy: Libraries or Frameworks

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

easier to avoid.

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

Phase: Architecture and Design

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

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

Phases: System Configuration; Installation

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

Relationships

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

Taxonomy Mappings

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

Related Attack Patterns

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

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

References

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

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

Content History

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

BACK TO TOP

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

Exposure of System Data to Unauthorized Control Sphere

Risk

What might happen

System data can provide attackers with valuable insights on systems and services they are targeting - any type of system data, from service version to operating system fingerprints, can assist attackers to hone their attack, correlate data with known vulnerabilities or focus efforts on developing new attacks against specific technologies.

Cause

How does it happen

System data is read and subsequently exposed where it might be read by untrusted entities.

General Recommendations

How to avoid it

Consider the implications of exposure of the specified input, and expected level of access to the specified output. If not required, consider removing this code, or modifying exposed information to exclude potentially sensitive system data.

Source Code Examples

Java

Leaking Environment Variables in JSP Web-Page

```
String envVarValue = System.getenv(envVar);
if (envVarValue == null) {
    out.println("Environment variable is not defined:");
    out.println(System.getenv());
} else {
    //[..]
};
```

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