disque Scan Report

Project Name	disque
Scan Start	Thursday, June 20, 2024 11:50:40 PM
Preset	Checkmarx Default
Scan Time	00h:07m:02s
Lines Of Code Scanned	12049
Files Scanned	9
Report Creation Time	Friday, June 21, 2024 12:06:43 AM
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005
Team	CxServer
Checkmarx Version	8.7.0
Scan Type	Full
Source Origin	LocalPath
Density	3/100 (Vulnerabilities/LOC)
Visibility	Public

Filter Settings

Severity

Included: High, Medium, Low, Information

Excluded: None

Result State

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

Excluded: None

Assigned to

Included: All

Categories

Included:

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

Excluded:

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

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

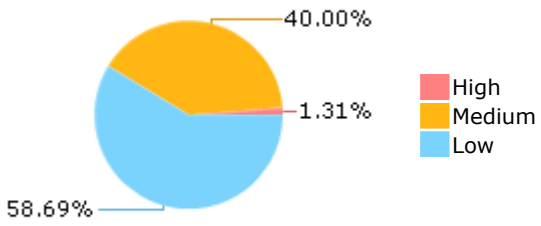
Results Limit

Results limit per query was set to 50

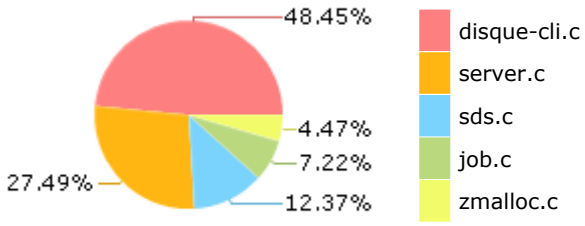
Selected Queries

Selected queries are listed in Result Summary

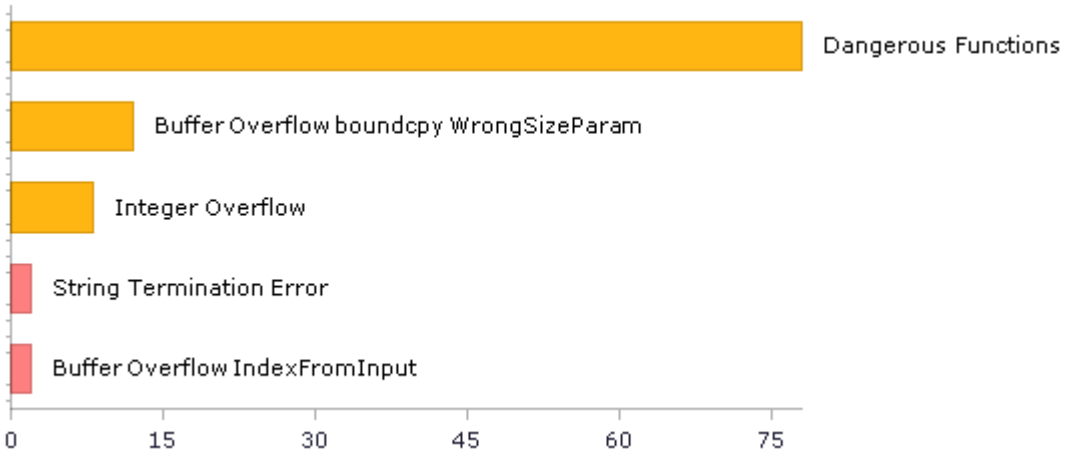
Result Summary



Most Vulnerable Files



Top 5 Vulnerabilities



Scan Summary - OWASP Top 10 2017

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

Category	Threat Agent	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection	App. Specific	EASY	COMMON	EASY	SEVERE	App. Specific	32	21
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	85	85
A3-Sensitive Data Exposure	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	4	4
A4-XML External Entities (XXE)	App. Specific	AVERAGE	COMMON	EASY	SEVERE	App. Specific	0	0
A5-Broken Access Control*	App. Specific	AVERAGE	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A6-Security Misconfiguration	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A7-Cross-Site Scripting (XSS)	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A8-Insecure Deserialization	App. Specific	DIFFICULT	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A9-Using Components with Known Vulnerabilities*	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	MODERATE	App. Specific	78	78
A10-Insufficient Logging & Monitoring	App. Specific	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	App. Specific	0	0

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

Scan Summary - OWASP Top 10 2013

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

Category	Threat Agent	Attack Vectors	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	AVERAGE	SEVERE	ALL DATA	0	0
A2-Broken Authentication and Session Management	EXTERNAL, INTERNAL USERS	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	AFFECTED DATA AND FUNCTIONS	0	0
A3-Cross-Site Scripting (XSS)	EXTERNAL, INTERNAL, ADMIN USERS	AVERAGE	VERY WIDESPREAD	EASY	MODERATE	AFFECTED DATA AND SYSTEM	0	0
A4-Insecure Direct Object References	SYSTEM USERS	EASY	COMMON	EASY	MODERATE	EXPOSED DATA	0	0
A5-Security Misconfiguration	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	EASY	MODERATE	ALL DATA AND SYSTEM	0	0
A6-Sensitive Data Exposure	EXTERNAL, INTERNAL, ADMIN USERS, USERS BROWSERS	DIFFICULT	UNCOMMON	AVERAGE	SEVERE	EXPOSED DATA	0	0
A7-Missing Function Level Access Control*	EXTERNAL, INTERNAL USERS	EASY	COMMON	AVERAGE	MODERATE	EXPOSED DATA AND FUNCTIONS	0	0
A8-Cross-Site Request Forgery (CSRF)	USERS BROWSERS	AVERAGE	COMMON	EASY	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0
A9-Using Components with Known Vulnerabilities*	EXTERNAL USERS, AUTOMATED TOOLS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	78	78
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	35	29
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.	7	7
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.	14	4
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.	78	78
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.	4	4
System And Communications Protection	Organizations must: (i) monitor, control, and protect organizational communications (i.e., information transmitted or received by organizational information systems) at the external boundaries and key internal boundaries of the information systems; and (ii) employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational information systems.	0	0
System And Information Integrity	Organizations must: (i) identify, report, and correct information and information system flaws in a timely manner; (ii) provide protection from malicious code at appropriate locations within organizational information systems; and (iii) monitor information system security alerts and advisories and take appropriate actions in response.	8	8

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

Scan Summary - NIST SP 800-53

Category	Issues Found	Best Fix Locations
AC-12 Session Termination (P2)	0	0
AC-3 Access Enforcement (P1)	99	89
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)	4	4
SC-4 Information in Shared Resources (P1)	0	0
SC-5 Denial of Service Protection (P1)*	17	10
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*	31	25
SI-11 Error Handling (P2)*	29	29
SI-15 Information Output Filtering (P0)	0	0
SI-16 Memory Protection (P1)	1	1

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

Scan Summary - 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

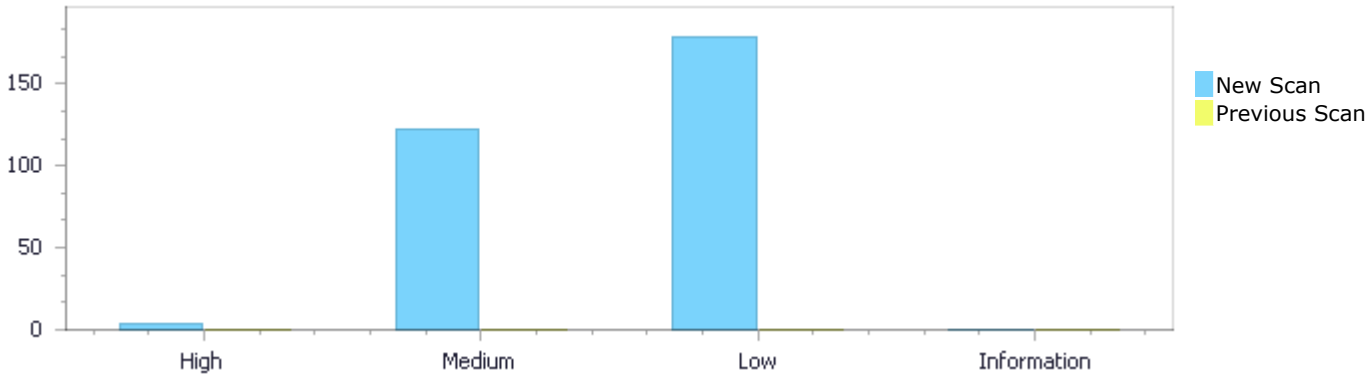
Scan Summary - Custom

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	122	179	0	305
Recurrent Issues	0	0	0	0	0
Total	4	122	179	0	305
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	122	179	0	305
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	4	122	179	0	305

Result Summary

Vulnerability Type	Occurrences	Severity
Buffer Overflow IndexFromInput	2	High
String Termination Error	2	High
Dangerous Functions	78	Medium
Buffer Overflow boundcpy WrongSizeParam	12	Medium
Integer Overflow	8	Medium

Divide By Zero	6	Medium
Use of Zero Initialized Pointer	6	Medium
Char Overflow	3	Medium
Memory Leak	3	Medium
MemoryFree on StackVariable	3	Medium
Buffer Overflow AddressOfLocalVarReturned	2	Medium
Double Free	1	Medium
Improper Resource Access Authorization	78	Low
Unchecked Return Value	29	Low
Exposure of System Data to Unauthorized Control Sphere	14	Low
TOCTOU	14	Low
Unchecked Array Index	10	Low
Use of Sizeof On a Pointer Type	8	Low
Incorrect Permission Assignment For Critical Resources	7	Low
Heuristic 2nd Order Buffer Overflow read	6	Low
NULL Pointer Dereference	6	Low
Use of Insufficiently Random Values	4	Low
Heuristic Buffer Overflow malloc	2	Low
Sizeof Pointer Argument	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
disque/disque-cli.c	39
disque/sds.c	31
disque/server.c	29
disque/job.c	21
disque/zmalloc.c	5
disque/queue.c	1

Scan Results Details

String Termination Error

Query Path:
CPP\Cx\CPP Buffer Overflow\String Termination Error 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

String Termination Error\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=65
Status	New

	Source	Destination
File	disque/zmalloc.c	disque/zmalloc.c
Line	242	251
Object	buf	strchr

Code Snippet
File Name disque/zmalloc.c
Method size_t zmalloc_get_rss(void) {

```
....
242.      if (read(fd,buf,4096) <= 0) {
....
251.          p = strchr(p,' ');
```

String Termination Error\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=66
Status	New

	Source	Destination
File	disque/zmalloc.c	disque/zmalloc.c
Line	242	255
Object	buf	strchr

Code Snippet

File Name	disque/zmalloc.c
Method	size_t zmalloc_get_rss(void) {

```
....
242.      if (read(fd,buf,4096) <= 0) {
....
255.      x = strchr(p,' ');
```

Buffer Overflow IndexFromInput

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

Categories

OWASP Top 10 2017: A1-Injection

Description

Buffer Overflow IndexFromInput\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=304
Status	New

The size of the buffer used by noninteractive in argc, at line 971 of disque/disque-cli.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 1876 of disque/disque-cli.c, to overwrite the target buffer.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1876	975
Object	argc	argc

Code Snippet

File Name	disque/disque-cli.c
Method	int main(int argc, char **argv) {

```
....
1876.  int main(int argc, char **argv) {
```

File Name	disque/disque-cli.c
Method	static int noninteractive(int argc, char **argv) {

```
....
975.          argv[argc] = readArgFromStdin();
```

Buffer Overflow IndexFromInput\Path 2:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=305
Status	New

The size of the buffer used by main in argc, at line 2507 of disque/server.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 2507 of disque/server.c, to overwrite the target buffer.

	Source	Destination
File	disque/server.c	disque/server.c
Line	2507	2527
Object	argc	argc

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

```
....
2507.  int main(int argc, char **argv) {
....
2527.      server.exec_argv[argc] = 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=1010004&projectid=10005&pathid=103
Status	New

The dangerous function, memcpy, was found in use at line 1194 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1307	1307
Object	memcpy	memcpy

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1307.                           memcpy(echo+21,magic,20);
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=104
Status	New

The dangerous function, memcpy, was found in use at line 1194 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1308	1308
Object	memcpy	memcpy

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1308.                           memcpy(obuf,echo,sizeof(echo)-1);
```

Dangerous Functions\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=105
Status	New

The dangerous function, memcpy, was found in use at line 1630 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1641	1641
Object	memcpy	memcpy

Code Snippet
File Name disque/disque-cli.c
Method static char *getInfoField(char *info, char *field) {

```
....
1641.       memcpy(result,p,(n1-p));
```

Dangerous Functions\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=106
Status	New

The dangerous function, memcpy, was found in use at line 168 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	171	171
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method int compareNodeIDsByJob(clusterNode *nodea, clusterNode *nodeb, job *j) {

```
....
171.       memcpy(ida,nodea->name,CLUSTER_NAMELEN);
```

Dangerous Functions\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=107
Status	New

The dangerous function, memcpy, was found in use at line 168 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	172	172
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method int compareNodeIDsByJob(clusterNode *nodea, clusterNode *nodeb, job *j) {

```
....
172.        memcpy(idb,nodeb->name,CLUSTER_NAMELEN);
```

Dangerous Functions\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=108
Status	New

The dangerous function, memcpy, was found in use at line 227 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	234	234
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method job *createJob(char *id, int state, int ttl, int retry) {

```
....
234.            memcpy(j->id,id,JOB_ID_LEN);
```

Dangerous Functions\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=109
Status	New

The dangerous function, memcpy, was found in use at line 606 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	610	610
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method char *serializeSdsString(char *p, sds s) {

```
....
610.        memcpy(p,&count,sizeof(count));
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=110
Status	New

The dangerous function, memcpy, was found in use at line 606 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	611	611
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method char *serializeSdsString(char *p, sds s) {

```
....
611.        if (s) memcpy(p+sizeof(count),s,len);
```

Dangerous Functions\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=111
Status	New

The dangerous function, memcpy, was found in use at line 665 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	688	688
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method sds serializeJob(sds jobs, job *j, int sertype) {

```
....
688.        memcpy(msg,&count,sizeof(count));
```

Dangerous Functions\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=112
Status	New

The dangerous function, memcpy, was found in use at line 665 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	693	693
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method sds serializeJob(sds jobs, job *j, int sertype) {

```
....
693.        memcpy(sj,j,JOB_STRUCT_SER_LEN);
```

Dangerous Functions\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=113
Status	New

The dangerous function, memcpy, was found in use at line 665 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	724	724
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method sds serializeJob(sds jobs, job *j, int sertype) {

```
....
724.      memcpy(p,&count,sizeof(count));
```

Dangerous Functions\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=114
Status	New

The dangerous function, memcpy, was found in use at line 665 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	731	731
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method sds serializeJob(sds jobs, job *j, int sertype) {

```
....
731.          memcpy(p,node->name,CLUSTER_NAMELEN);
```

Dangerous Functions\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=115
Status	New

The dangerous function, memcpy, was found in use at line 764 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	774	774
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method job *deserializeJob(unsigned char *p, size_t len, unsigned char **next, int sertype) {

```
....
774.      memcpy(&joblen,p,sizeof(joblen));
```

Dangerous Functions\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=116
Status	New

The dangerous function, memcpy, was found in use at line 764 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	781	781
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method job *deserializeJob(unsigned char *p, size_t len, unsigned char **next, int sertype) {

```
....
781.      memcpy(j,p,JOB_STRUCT_SER_LEN);
```

Dangerous Functions\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=117
Status	New

The dangerous function, memcpy, was found in use at line 764 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	815	815
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method job *deserializeJob(unsigned char *p, size_t len, unsigned char **next, int sertype) {

```
....
815.      memcpy(&aux,p,sizeof(aux));
```

Dangerous Functions\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=118
Status	New

The dangerous function, memcpy, was found in use at line 764 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	826	826
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method job *deserializeJob(unsigned char *p, size_t len, unsigned char **next, int sertype) {

```
....
826.      memcpy(&aux,p,sizeof(aux));
```

Dangerous Functions\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=119
Status	New

The dangerous function, memcpy, was found in use at line 764 in disque/job.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/job.c	disque/job.c
Line	837	837
Object	memcpy	memcpy

Code Snippet
File Name disque/job.c
Method job *deserializeJob(unsigned char *p, size_t len, unsigned char **next, int sertype) {

```
....
837.      memcpy(&aux,p,sizeof(aux));
```

Dangerous Functions\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=120
Status	New

The dangerous function, memcpy, was found in use at line 81 in disque/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	132	132
Object	memcpy	memcpy

Code Snippet
File Name disque/sds.c
Method sds sdsnewlen(const void *init, size_t initlen) {

```
....
132.          memcpy(s, init, initlen);
```

Dangerous Functions\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=121
Status	New

The dangerous function, memcpy, was found in use at line 194 in disque/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	229	229
Object	memcpy	memcpy

Code Snippet
File Name disque/sds.c
Method sds sdsMakeRoomFor(sds s, size_t addlen) {

```
....
229.          memcpy((char*)newsh+hdrlen, s, len+1);
```

Dangerous Functions\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=122
Status	New

The dangerous function, memcpy, was found in use at line 245 in disque/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	261	261
Object	memcpy	memcpy

Code Snippet
File Name disque/sds.c
Method sds sdsRemoveFreeSpace(sds s) {

```
....
261.          memcpy((char*)newsh+hdrlen, s, len+1);
```

Dangerous Functions\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=123
Status	New

The dangerous function, memcpy, was found in use at line 376 in disque/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	381	381
Object	memcpy	memcpy

Code Snippet
File Name disque/sds.c
Method sds sdscatlen(sds s, const void *t, size_t len) {

```
....
381.        memcpy(s+curlen, t, len);
```

Dangerous Functions\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=124
Status	New

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

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	410	410
Object	memcpy	memcpy

Code Snippet
File Name disque/sds.c
Method sds sdscpylen(sds s, const char *t, size_t len) {

```
....
410.        memcpy(s, t, len);
```

Dangerous Functions\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=125
Status	New

The dangerous function, memcpy, was found in use at line 579 in disque/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	611	611
Object	memcpy	memcpy

Code Snippet
File Name disque/sds.c
Method sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
611.                    memcpy(s+i,str,l);
```

Dangerous Functions\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=126
Status	New

The dangerous function, memcpy, was found in use at line 579 in disque/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	627	627
Object	memcpy	memcpy

Code Snippet
File Name disque/sds.c
Method sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
627.                        memcpy(s+i,buf,l);
```

Dangerous Functions\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=127
Status	New

The dangerous function, memcpy, was found in use at line 579 in disque/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	644	644
Object	memcpy	memcpy

Code Snippet
File Name disque/sds.c
Method sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
644.                        memcpy(s+i,buf,l);
```

Dangerous Functions\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=128
Status	New

The dangerous function, memcpy, was found in use at line 1715 in disque/server.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1735	1735
Object	memcpy	memcpy

Code Snippet
File Name disque/server.c
Method int time_independent_strcmp(char *a, char *b) {

```
....
1735.       memcpy(bufa,a,alen);
```

Dangerous Functions\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=129
Status	New

The dangerous function, memcpy, was found in use at line 1715 in disque/server.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1736	1736
Object	memcpy	memcpy

Code Snippet
File Name disque/server.c
Method int time_independent_strcmp(char *a, char *b) {

```
....
1736.       memcpy(bufb,b,blen);
```

Dangerous Functions\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=130
Status	New

The dangerous function, memcpy, was found in use at line 193 in disque/zmalloc.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/zmalloc.c	disque/zmalloc.c
Line	197	197
Object	memcpy	memcpy

Code Snippet
File Name disque/zmalloc.c
Method char *zstrdup(const char *s) {

```
....
197.       memcpy(p,s,l);
```

Dangerous Functions\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=131
Status	New

The dangerous function, sprintf, was found in use at line 1660 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1670	1670
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method void bytesToHuman(char *s, long long n) {

```
....
1670.           sprintf(s,"%lldB",n);
```

Dangerous Functions\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=132
Status	New

The dangerous function, sprintf, was found in use at line 1660 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1674	1674
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method void bytesToHuman(char *s, long long n) {

```
....
1674.           sprintf(s,"%.2fK",d);
```

Dangerous Functions\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=133
Status	New

The dangerous function, sprintf, was found in use at line 1660 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1677	1677
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method void bytesToHuman(char *s, long long n) {

```
....
1677.           sprintf(s,"%.2fM",d);
```

Dangerous Functions\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=134
Status	New

The dangerous function, sprintf, was found in use at line 1660 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1680	1680
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method void bytesToHuman(char *s, long long n) {

```
....
1680.           sprintf(s,"%.2fG",d);
```

Dangerous Functions\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=135
Status	New

The dangerous function, sprintf, was found in use at line 1684 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1710	1710
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1710.               sprintf(buf,"db%d:keys",j);
```

Dangerous Functions\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=136
Status	New

The dangerous function, sprintf, was found in use at line 1684 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1715	1715
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1715.           sprintf(buf,"%ld",aux);
```

Dangerous Functions\Path 35:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=137
Status	New

The dangerous function, sprintf, was found in use at line 1684 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1725	1725
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1725.           sprintf(buf,"%ld",aux);
```

Dangerous Functions\Path 36:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=138
Status	New

The dangerous function, sprintf, was found in use at line 1684 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1730	1730
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1730.           sprintf(buf,"%ld",aux);
```

Dangerous Functions\Path 37:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=139
Status	New

The dangerous function, sprintf, was found in use at line 1684 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1735	1735
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1735.           sprintf(buf,"%ld (+%ld)",aux,requests == 0 ? 0 : aux-
requests);
```

Dangerous Functions\Path 38:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=140
Status	New

The dangerous function, sprintf, was found in use at line 1684 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1741	1741
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1741.           sprintf(buf,"%ld",aux);
```

Dangerous Functions\Path 39:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=141
Status	New

The dangerous function, sprintf, was found in use at line 1876 in disque/server.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1881	1881
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1881.           sprintf(s,"%lluB",n);
```

Dangerous Functions\Path 40:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=142
Status	New

The dangerous function, sprintf, was found in use at line 1876 in disque/server.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1885	1885
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1885.           sprintf(s,"%.2fK",d);
```

Dangerous Functions\Path 41:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=143
Status	New

The dangerous function, sprintf, was found in use at line 1876 in disque/server.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1888	1888
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1888.           sprintf(s,"%.2fM",d);
```

Dangerous Functions\Path 42:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=144
Status	New

The dangerous function, sprintf, was found in use at line 1876 in disque/server.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1891	1891
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1891.           sprintf(s,"%.2fG",d);
```

Dangerous Functions\Path 43:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=145
Status	New

The dangerous function, sprintf, was found in use at line 1876 in disque/server.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1894	1894
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1894.          sprintf(s,"%.2fT",d);
```

Dangerous Functions\Path 44:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=146
Status	New

The dangerous function, sprintf, was found in use at line 1876 in disque/server.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1897	1897
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1897.          sprintf(s,"%.2fP",d);
```

Dangerous Functions\Path 45:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=147
Status	New

The dangerous function, sprintf, was found in use at line 1876 in disque/server.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1900	1900
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1900.          sprintf(s,"%lluB",n);
```

Dangerous Functions\Path 46:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=148
Status	New

The dangerous function, strlen, was found in use at line 272 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	290	290
Object	strlen	strlen

Code Snippet
File Name disque/disque-cli.c
Method static void completionCallback(const char *buf, linenoiseCompletions *lc) {

```
....
290.          matchlen = strlen(buf+startpos);
```

Dangerous Functions\Path 47:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=149
Status	New

The dangerous function, strlen, was found in use at line 477 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	484	484
Object	strlen	strlen

Code Snippet
File Name disque/disque-cli.c
Method static sds cliFormatReplyCSV(redisReply *r) {

```
....
484.          out = sdscatrepr(out,r->str,strlen(r->str));
```

Dangerous Functions\Path 48:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=150
Status	New

The dangerous function, strlen, was found in use at line 1630 in disque/disque-cli.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1636	1636
Object	strlen	strlen

Code Snippet
File Name disque/disque-cli.c
Method static char *getInfoField(char *info, char *field) {

```
....
1636.      p += strlen(field)+1;
```

Dangerous Functions\Path 49:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=151
Status	New

The dangerous function, strlen, was found in use at line 144 in disque/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	145	145
Object	strlen	strlen

Code Snippet
File Name disque/sds.c
Method sds sdsnew(const char *init) {

```
....
145.      size_t initlen = (init == NULL) ? 0 : strlen(init);
```

Dangerous Functions\Path 50:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=152
Status	New

The dangerous function, strlen, was found in use at line 174 in disque/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	175	175
Object	strlen	strlen

Code Snippet
File Name disque/sds.c
Method void sdsupdatelen(sds s) {

```
....
175.      int reallen = strlen(s);
```

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=1010004&projectid=10005&pathid=50
Status	New

The size of the buffer used by *serializeSdsString in count, at line 606 of disque/job.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *serializeSdsString passes to count, at line 606 of disque/job.c, to overwrite the target buffer.

	Source	Destination
File	disque/job.c	disque/job.c
Line	610	610

Object	count	count

Code Snippet
File Name disque/job.c
Method char *serializeSdsString(char *p, sds s) {

```
....
610.        memcpy(p,&count,sizeof(count));
```

Buffer Overflow boundcpy WrongSizeParam\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=51
Status	New

The size of the buffer used by serializeJob in count, at line 665 of disque/job.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that serializeJob passes to count, at line 665 of disque/job.c, to overwrite the target buffer.

	Source	Destination
File	disque/job.c	disque/job.c
Line	688	688
Object	count	count

Code Snippet
File Name disque/job.c
Method sds serializeJob(sds jobs, job *j, int sertype) {

```
....
688.        memcpy(msg,&count,sizeof(count));
```

Buffer Overflow boundcpy WrongSizeParam\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=52
Status	New

The size of the buffer used by serializeJob in count, at line 665 of disque/job.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that serializeJob passes to count, at line 665 of disque/job.c, to overwrite the target buffer.

	Source	Destination
File	disque/job.c	disque/job.c
Line	724	724
Object	count	count

Code Snippet	
File Name	disque/job.c
Method	sds serializeJob(sds jobs, job *j, int sertype) {

```
....
724.        memcpy(p,&count,sizeof(count));
```

Buffer Overflow boundcpy WrongSizeParam\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=53
Status	New

The size of the buffer used by sdscatlen in len, at line 376 of disque/sds.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that sdscatlen passes to len, at line 376 of disque/sds.c, to overwrite the target buffer.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	381	381
Object	len	len

Code Snippet	
File Name	disque/sds.c
Method	sds sdscatlen(sds s, const void *t, size_t len) {

```
....
381.        memcpy(s+curlen, t, len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=54
Status	New

The size of the buffer used by sdscpylen in len, at line 405 of disque/sds.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that sdscpylen passes to len, at line 405 of disque/sds.c, to overwrite the target buffer.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	410	410
Object	len	len

Code Snippet	
File Name	disque/sds.c

Method	sds sdscpylen(sds s, const char *t, size_t len) {

```
....
410.      memcpy(s, t, len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=55
Status	New

The size of the buffer used by sdscatfmt in l, at line 579 of disque/sds.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that sdscatfmt passes to l, at line 579 of disque/sds.c, to overwrite the target buffer.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	611	611
Object	l	l

Code Snippet	
File Name	disque/sds.c
Method	sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
611.                  memcpy(s+i,str,l);
```

Buffer Overflow boundcpy WrongSizeParam\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=56
Status	New

The size of the buffer used by sdscatfmt in l, at line 579 of disque/sds.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that sdscatfmt passes to l, at line 579 of disque/sds.c, to overwrite the target buffer.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	627	627
Object	l	l

Code Snippet	
File Name	disque/sds.c
Method	sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
627.                        memcpy(s+i,buf,l);
```

Buffer Overflow boundcpy WrongSizeParam\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=57
Status	New

The size of the buffer used by sdscatfmt in l, at line 579 of disque/sds.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that sdscatfmt passes to l, at line 579 of disque/sds.c, to overwrite the target buffer.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	644	644
Object	l	l

Code Snippet
File Name disque/sds.c
Method sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
644.                        memcpy(s+i,buf,l);
```

Buffer Overflow boundcpy WrongSizeParam\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=58
Status	New

The size of the buffer used by time_independent_strcmp in alen, at line 1715 of disque/server.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that time_independent_strcmp passes to alen, at line 1715 of disque/server.c, to overwrite the target buffer.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1735	1735
Object	alen	alen

Code Snippet
File Name disque/server.c
Method int time_independent_strcmp(char *a, char *b) {

```
....
1735.       memcpy(bufa,a,alen);
```

Buffer Overflow boundcpy WrongSizeParam\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=59
Status	New

The size of the buffer used by time_independent_strcmp in blen, at line 1715 of disque/server.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that time_independent_strcmp passes to blen, at line 1715 of disque/server.c, to overwrite the target buffer.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1736	1736
Object	blen	blen

Code Snippet
File Name disque/server.c
Method int time_independent_strcmp(char *a, char *b) {

```
....
1736.       memcpy(bufb,b,blen);
```

Buffer Overflow boundcpy WrongSizeParam\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=60
Status	New

The size of the buffer used by *zstrdup in l, at line 193 of disque/zmalloc.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *zstrdup passes to l, at line 193 of disque/zmalloc.c, to overwrite the target buffer.

	Source	Destination
File	disque/zmalloc.c	disque/zmalloc.c
Line	197	197
Object	l	l

Code Snippet
File Name disque/zmalloc.c
Method char *zstrdup(const char *s) {

```
....
197.        memcpy(p,s,l);
```

Buffer Overflow boundcpy WrongSizeParam\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=61
Status	New

The size of the buffer used by sdscmp in minlen, at line 767 of disque/sds.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that sdscmp passes to minlen, at line 767 of disque/sds.c, to overwrite the target buffer.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	774	774
Object	minlen	minlen

Code Snippet
File Name disque/sds.c
Method int sdscmp(const sds s1, const sds s2) {

```
....
774.        cmp = memcmp(s1,s2,minlen);
```

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=1010004&projectid=10005&pathid=84
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 579 of disque/sds.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/sds.c	disque/sds.c

Line	587	587
Object	AssignExpr	AssignExpr

Code Snippet
File Name disque/sds.c
Method sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
587.      i = initlen; /* Position of the next byte to write to dest
str. */
```

Integer Overflow\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=85
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 579 of disque/sds.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	613	613
Object	AssignExpr	AssignExpr

Code Snippet
File Name disque/sds.c
Method sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
613.                  i += l;
```

Integer Overflow\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=86
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 579 of disque/sds.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	629	629
Object	AssignExpr	AssignExpr

Code Snippet	
File Name	disque/sds.c
Method	sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
629.                        i += l;
```

Integer Overflow\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=87
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 579 of disque/sds.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	646	646
Object	AssignExpr	AssignExpr

Code Snippet	
File Name	disque/sds.c
Method	sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
646.                        i += l;
```

Integer Overflow\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=88
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 714 of disque/sds.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	719	719
Object	AssignExpr	AssignExpr

Code Snippet	
File Name	disque/sds.c
Method	void sdsrange(sds s, int start, int end) {

```
....
719.          start = len+start;
```

Integer Overflow\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=89
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 714 of disque/sds.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	723	723
Object	AssignExpr	AssignExpr

Code Snippet
File Name disque/sds.c
Method void sdsrange(sds s, int start, int end) {

```
....
723.          end = len+end;
```

Integer Overflow\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=90
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 714 of disque/sds.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	731	731
Object	AssignExpr	AssignExpr

Code Snippet
File Name disque/sds.c
Method void sdsrange(sds s, int start, int end) {

```
....
731.              end = len-1;
```

Integer Overflow\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=91
Status	New

A variable of a larger data type, additional_nodes, is being assigned to a smaller data type, in 1135 of disque/job.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/job.c	disque/job.c
Line	1229	1229
Object	additional_nodes	additional_nodes

Code Snippet
File Name disque/job.c
Method void addjobCommand(client *c) {

```
....
1229.       int additional_nodes = extrepl ? replicate : replicate-1;
```

Divide By Zero

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

Description

Divide By Zero\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=42
Status	New

The application performs an illegal operation in findBigKeys, in disque/disque-cli.c. In line 1492, the program attempts to divide by sampled, which might be evaluate to 0 (zero) at time of division. This value could be a hard-coded zero value, or received from external, untrusted input sampled in findBigKeys of disque/disque-cli.c, at line 1492.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1595	1595
Object	sampled	sampled

Code Snippet
File Name disque/disque-cli.c
Method static void findBigKeys(void) {

```
....
1595.         totlen, totlen ? (double)totlen/sampled : 0);
```

Divide By Zero\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=43
Status	New

The application performs an illegal operation in findBigKeys, in disque/disque-cli.c. In line 1492, the program attempts to divide by sampled, which might be evaluate to 0 (zero) at time of division. This value could be a hard-coded zero value, or received from external, untrusted input sampled in findBigKeys of disque/disque-cli.c, at line 1492.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1610	1610
Object	sampled	sampled

Code Snippet
File Name disque/disque-cli.c
Method static void findBigKeys(void) {

```
....
1610.             sampled ? 100 * (double)counts[i]/sampled : 0,
```

Divide By Zero\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=44
Status	New

The application performs an illegal operation in serverCron, in disque/server.c. In line 749, the program attempts to divide by base, which might be evaluate to 0 (zero) at time of division. This value could be a hard-coded zero value, or received from external, untrusted input base in serverCron of disque/server.c, at line 749.

	Source	Destination
File	disque/server.c	disque/server.c
Line	824	824
Object	base	base

Code Snippet
File Name disque/server.c
Method int serverCron(struct aeEventLoop *eventLoop, long long id, void *clientData) {

```
....
824.                long long growth = (server.aof_current_size*100/base)
- 100;
```

Divide By Zero\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=45
Status	New

The application performs an illegal operation in initServerConfig, in disque/server.c. In line 970, the program attempts to divide by R_Zero, which might be evaluate to 0 (zero) at time of division. This value could be a hard-coded zero value, or received from external, untrusted input R_Zero in initServerConfig of disque/server.c, at line 970.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1038	1038
Object	R_Zero	R_Zero

Code Snippet
File Name disque/server.c
Method void initServerConfig(void) {

```
....
1038.       R_PosInf = 1.0/R_Zero;
```

Divide By Zero\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=46
Status	New

The application performs an illegal operation in initServerConfig, in disque/server.c. In line 970, the program attempts to divide by R_Zero, which might be evaluate to 0 (zero) at time of division. This value could be a hard-coded zero value, or received from external, untrusted input R_Zero in initServerConfig of disque/server.c, at line 970.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1039	1039
Object	R_Zero	R_Zero

Code Snippet

File Name	disque/server.c
Method	void initServerConfig(void) {

```
....
1039.       R_NegInf = -1.0/R_Zero;
```

Divide By Zero\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=47
Status	New

The application performs an illegal operation in initServerConfig, in disque/server.c. In line 970, the program attempts to divide by R_Zero, which might be evaluate to 0 (zero) at time of division. This value could be a hard-coded zero value, or received from external, untrusted input R_Zero in initServerConfig of disque/server.c, at line 970.

	Source	Destination
File	disque/server.c	disque/server.c
Line	1040	1040
Object	R_Zero	R_Zero

Code Snippet	
File Name	disque/server.c
Method	void initServerConfig(void) {

```
....
1040.       R_Nan = R_Zero/R_Zero;
```

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=1010004&projectid=10005&pathid=185
Status	New

The variable declared in sizes at disque/disque-cli.c in line 1492 is not initialized when it is used by sizes at disque/disque-cli.c in line 1492.

	Source	Destination

File	disque/disque-cli.c	disque/disque-cli.c
Line	1494	1532
Object	sizes	sizes

Code Snippet
File Name disque/disque-cli.c
Method static void findBigKeys(void) {

```
....
1494.      unsigned long long sampled = 0, total_keys, totlen=0,
*sizes=NULL, it=0;
....
1532.              sizes = zrealloc(sizes, sizeof(unsigned long
long)*keys->elements);
```

Use of Zero Initialized Pointer\Path 2:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=186
Status New

The variable declared in context at disque/disque-cli.c in line 334 is not initialized when it is used by sizes at disque/disque-cli.c in line 1492.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	352	1532
Object	context	sizes

Code Snippet
File Name disque/disque-cli.c
Method static int cliConnect(int force) {

```
....
352.              context = NULL;
```

File Name disque/disque-cli.c
Method static void findBigKeys(void) {

```
....
1532.              sizes = zrealloc(sizes, sizeof(unsigned long
long)*keys->elements);
```

Use of Zero Initialized Pointer\Path 3:

Severity Medium
Result State To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=187
Status	New

The variable declared in BinaryExpr at disque/queue.c in line 738 is not initialized when it is used by job at disque/queue.c in line 738.

	Source	Destination
File	disque/queue.c	disque/queue.c
Line	743	792
Object	BinaryExpr	job

Code Snippet
File Name disque/queue.c
Method void getjobCommand(client *c) {

```
....
743.      robj **queues = NULL;
....
792.                  job = queueFetchJob(q,&qlen);
```

Use of Zero Initialized Pointer\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=188
Status	New

The variable declared in vector at disque/sds.c in line 933 is not initialized when it is used by vector at disque/sds.c in line 933.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	936	1023
Object	vector	vector

Code Snippet
File Name disque/sds.c
Method sds *sdssplitargs(const char *line, int *argc) {

```
....
936.      char **vector = NULL;
....
1023.             vector = s_realloc(vector,((*argc)+1)*sizeof(char*));
```

Use of Zero Initialized Pointer\Path 5:

Severity	Medium
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=189
Status	New

The variable declared in current at disque/sds.c in line 933 is not initialized when it is used by vector at disque/sds.c in line 933.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	1026	1024
Object	current	vector

Code Snippet
File Name disque/sds.c
Method sds *sdssplitargs(const char *line, int *argc) {

```
....
1026.               current = NULL;
....
1024.               vector[*argc] = current;
```

Use of Zero Initialized Pointer\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=190
Status	New

The variable declared in vector at disque/sds.c in line 933 is not initialized when it is used by vector at disque/sds.c in line 933.

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	936	1036
Object	vector	vector

Code Snippet
File Name disque/sds.c
Method sds *sdssplitargs(const char *line, int *argc) {

```
....
936.       char **vector = NULL;
....
1036.           sdsfree(vector[*argc]);
```

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=1010004&projectid=10005&pathid=62
Status	New

Calling free() (line 880) on a variable that was not dynamically allocated (line 880) in file disque/disque-cli.c may result with a crash.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	910	910
Object	line	line

Code Snippet
File Name disque/disque-cli.c
Method static void repl(void) {

```
....
910.                    free(line);
```

MemoryFree on StackVariable\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=63
Status	New

Calling free() (line 880) on a variable that was not dynamically allocated (line 880) in file disque/disque-cli.c may result with a crash.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	966	966
Object	line	line

Code Snippet
File Name disque/disque-cli.c
Method static void repl(void) {

```
....
966.            free(line);
```

MemoryFree on StackVariable\Path 3:

Severity	Medium

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=64
Status	New

Calling free() (line 1648) on a variable that was not dynamically allocated (line 1648) in file disque/disque-cli.c may result with a crash.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1654	1654
Object	value	value

Code Snippet	
File Name	disque/disque-cli.c
Method	static long getLongInfoField(char *info, char *field) {

```
....
1654.       free(value);
```

Char Overflow

Query Path:
CPP\Cx\CPP Integer Overflow\Char Overflow 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)

Description

Char Overflow\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=81
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 1810 of disque/disque-cli.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1815	1815
Object	AssignExpr	AssignExpr

Code Snippet	
File Name	disque/disque-cli.c
Method	unsigned long compute_something_fast(void) {

```
....
1815.       for (k = 0; k < 256; k++) s[k] = k;
```

Char Overflow\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=82
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 74 of disque/job.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/job.c	disque/job.c
Line	98	98
Object	AssignExpr	AssignExpr

Code Snippet
File Name disque/job.c
Method void generateJobID(char *id, int ttl, int retry) {

```
....
98.       ttlbytes[0] = (ttl&0xff00)>>8;
```

Char Overflow\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=83
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 74 of disque/job.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	disque/job.c	disque/job.c
Line	99	99
Object	AssignExpr	AssignExpr

Code Snippet
File Name disque/job.c
Method void generateJobID(char *id, int ttl, int retry) {

```
....
99.       ttlbytes[1] = ttl&0xff;
```

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=1010004&projectid=10005&pathid=182
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	182	182
Object	helpEntries	helpEntries

Code Snippet
File Name disque/disque-cli.c
Method static void cliInitHelp(void) {

```
....
182.      helpEntries = malloc(sizeof(helpEntry)*len);
```

Memory Leak\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=183
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	186	186
Object	argv	argv

Code Snippet
File Name disque/disque-cli.c
Method static void cliInitHelp(void) {

```
....
186.          tmp.argv = malloc(sizeof(sds));
```

Memory Leak\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=184
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1640	1640
Object	result	result

Code Snippet
File Name disque/disque-cli.c
Method static char *getInfoField(char *info, char *field) {

```
....
1640.       result = malloc(sizeof(char)*(n1-p)+1);
```

Buffer Overflow AddressOfLocalVarReturned

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

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
NIST SP 800-53: SC-5 Denial of Service Protection (P1)
OWASP Top 10 2017: A1-Injection

Description
Buffer Overflow AddressOfLocalVarReturned\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=48
Status	New

The pointer states at disque/job.c in line 340 is being used after it has been freed.

	Source	Destination
File	disque/job.c	disque/job.c
Line	343	343
Object	states	states

Code Snippet
File Name disque/job.c
Method char *jobStateToString(int state) {

```
....
343.      return states[state];
```

Buffer Overflow AddressOfLocalVarReturned\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=49
Status	New

The pointer o2 at disque/server.c in line 372 is being used after it has been freed.

	Source	Destination
File	disque/server.c	disque/server.c
Line	380	380
Object	o2	o2

Code Snippet
File Name disque/server.c
Method int dictEncObjKeyCompare(void *privdata, const void *key1,

```
....
380.              return o1->ptr == o2->ptr;
```

Double Free

Query Path:
CPP\Cx\CPP Medium Threat\Double Free Version:1

Categories

NIST SP 800-53: SI-16 Memory Protection (P1)

Description

Double Free\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=181
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	910	966
Object	line	line

Code Snippet
File Name disque/disque-cli.c

Method static void repl(void) {

```
....
910.                    free(line);
....
966.            free(line);
```

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=1010004&projectid=10005&pathid=191
Status	New

	Source	Destination
File	disque/debug.c	disque/debug.c
Line	448	448
Object	fgets	fgets

Code Snippet
File Name disque/debug.c
Method int memtest_test_linux_anonymous_maps(void) {

```
....
448.        while(fgets(line,sizeof(line),fp) != NULL) {
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=192
Status	New

	Source	Destination
File	disque/server.c	disque/server.c
Line	1203	1203
Object	fgets	fgets

Code Snippet
File Name disque/server.c
Method void checkTcpBacklogSettings(void) {

```
....
1203.       if (fgets(buf,sizeof(buf),fp) != NULL) {
```

Improper Resource Access Authorization\Path 3:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=193
Status New

	Source	Destination
File	disque/server.c	disque/server.c
Line	2319	2319
Object	fgets	fgets

Code Snippet
File Name disque/server.c
Method int linuxOvercommitMemoryValue(void) {

```
....
2319.       if (fgets(buf,64,fp) == NULL) {
```

Improper Resource Access Authorization\Path 4:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=194
Status New

	Source	Destination
File	disque/zmalloc.c	disque/zmalloc.c
Line	306	306
Object	fgets	fgets

Code Snippet
File Name disque/zmalloc.c
Method size_t zmalloc_get_private_dirty(void) {

```
....
306.       while(fgets(line,sizeof(line),fp) != NULL) {
```

Improper Resource Access Authorization\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=195
Status	New

	Source	Destination
File	disque/debug.c	disque/debug.c
Line	448	448
Object	line	line

Code Snippet
File Name disque/debug.c
Method int memtest_test_linux_anonymous_maps(void) {

```
....
448.      while(fgets(line,sizeof(line),fp) != NULL) {
```

Improper Resource Access Authorization\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=196
Status	New

	Source	Destination
File	disque/server.c	disque/server.c
Line	1203	1203
Object	buf	buf

Code Snippet
File Name disque/server.c
Method void checkTcpBacklogSettings(void) {

```
....
1203.      if (fgets(buf,sizeof(buf),fp) != NULL) {
```

Improper Resource Access Authorization\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=197
Status	New

	Source	Destination
File	disque/server.c	disque/server.c
Line	2319	2319
Object	buf	buf

Code Snippet
File Name disque/server.c
Method int linuxOvercommitMemoryValue(void) {

```
....
2319.       if (fgets(buf,64,fp) == NULL) {
```

Improper Resource Access Authorization\Path 8:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=198
Status New

	Source	Destination
File	disque/zmalloc.c	disque/zmalloc.c
Line	306	306
Object	line	line

Code Snippet
File Name disque/zmalloc.c
Method size_t zmalloc_get_private_dirty(void) {

```
....
306.       while(fgets(line,sizeof(line),fp) != NULL) {
```

Improper Resource Access Authorization\Path 9:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=199
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1003	1003
Object	buf	buf

Code Snippet

File Name	disque/disque-cli.c
Method	static int evalMode(int argc, char **argv) {

```
....
1003.       while((nread = fread(buf,1,sizeof(buf),fp)) != 0) {
```

Improper Resource Access Authorization\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=200
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	800	800
Object	buf	buf

Code Snippet

File Name	disque/disque-cli.c
Method	static sds readArgFromStdin(void) {

```
....
800.           int nread = read(fileno(stdin),buf,1024);
```

Improper Resource Access Authorization\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=201
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1097	1097
Object	p	p

Code Snippet

File Name	disque/disque-cli.c
Method	unsigned long long sendSync(int fd) {

```
....
1097.           nread = read(fd,p,1);
```

Improper Resource Access Authorization\Path 12:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=202
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1126	1126
Object	buf	buf

Code Snippet
File Name disque/disque-cli.c
Method static void slaveMode(void) {

```
....
1126.           nread = read(fd,buf,(payload > sizeof(buf)) ? sizeof(buf)
: payload);
```

Improper Resource Access Authorization\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=203
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1171	1171
Object	buf	buf

Code Snippet
File Name disque/disque-cli.c
Method static void getRDB(void) {

```
....
1171.           nread = read(s,buf,(payload > sizeof(buf)) ? sizeof(buf)
: payload);
```

Improper Resource Access Authorization\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=204
Status	New

	Source	Destination

File	disque/disque-cli.c	disque/disque-cli.c
Line	1229	1229
Object	ibuf	ibuf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1229.                   nread = read(fd,ibuf,sizeof(ibuf));
```

Improper Resource Access Authorization\Path 15:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=205
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1290	1290
Object	obuf	obuf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1290.                       ssize_t nread =
read(STDIN_FILENO,obuf,sizeof(obuf));
```

Improper Resource Access Authorization\Path 16:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=206
Status New

	Source	Destination
File	disque/zmalloc.c	disque/zmalloc.c
Line	242	242
Object	buf	buf

Code Snippet
File Name disque/zmalloc.c

Method	size_t zmalloc_get_rss(void) {

```
....
242.      if (read(fd,buf,4096) <= 0) {
```

Improper Resource Access Authorization\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=207
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	346	346
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static int cliConnect(int force) {

```
....
346.              fprintf(stderr,"Could not connect to Disque at ");
```

Improper Resource Access Authorization\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=208
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	348	348
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static int cliConnect(int force) {

```
....
348.                  fprintf(stderr,"%s:%d:
%s\n",config.hostip,config.hostport,context->errstr);
```

Improper Resource Access Authorization\Path 19:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=209
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	350	350
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static int cliConnect(int force) {

```
....
350.                    fprintf(stderr,"%s:
%s\n",config.hostsocket,context->errstr);
```

Improper Resource Access Authorization\Path 20:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=210
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	373	373
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void cliPrintContextError(void) {

```
....
373.      fprintf(stderr,"Error: %s\n",context->errstr);
```

Improper Resource Access Authorization\Path 21:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=211
Status	New

	Source	Destination

File	disque/disque-cli.c	disque/disque-cli.c
Line	437	437
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static sds cliFormatReplyTTY(redisReply *r, char *prefix) {

```
....
437.            fprintf(stderr,"Unknown reply type: %d\n", r->type);
```

Improper Resource Access Authorization\Path 22:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=212
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	471	471
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static sds cliFormatReplyRaw(redisReply *r) {

```
....
471.            fprintf(stderr,"Unknown reply type: %d\n", r->type);
```

Improper Resource Access Authorization\Path 23:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=213
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	507	507
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static sds cliFormatReplyCSV(redisReply *r) {

```
....
507.          fprintf(stderr,"Unknown reply type: %d\n", r->type);
```

Improper Resource Access Authorization\Path 24:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=214
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	694	694
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static redisReply *reconnectingInfo(void) {

```
....
694.              fprintf(stderr, "Error: %s\n", c->errstr);
```

Improper Resource Access Authorization\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=215
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	782	782
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static int parseOptions(int argc, char **argv) {

```
....
782.                  fprintf(stderr,
```

Improper Resource Access Authorization\Path 26:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=216
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	814	814
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void usage(void) {

```
....
814.        fprintf(stderr,
```

Improper Resource Access Authorization\Path 27:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=217
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	999	999
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static int evalMode(int argc, char **argv) {

```
....
999.            fprintf(stderr,
```

Improper Resource Access Authorization\Path 28:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=218
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1046	1046

Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void latencyMode(void) {

```
....
1046.               fprintf(stderr,"\nI/O error\n");
```

Improper Resource Access Authorization\Path 29:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=219
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1090	1090
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method unsigned long long sendSync(int fd) {

```
....
1090.           fprintf(stderr,"Error writing to master\n");
```

Improper Resource Access Authorization\Path 30:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=220
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1099	1099
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method unsigned long long sendSync(int fd) {

```
....
1099.               fprintf(stderr,"Error reading bulk length while
SYNCing\n");
```

Improper Resource Access Authorization\Path 31:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=221
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1119	1119
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void slaveMode(void) {

```
....
1119.       fprintf(stderr,"SYNC with master, discarding %llu "
```

Improper Resource Access Authorization\Path 32:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=222
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1128	1128
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void slaveMode(void) {

```
....
1128.               fprintf(stderr,"Error reading RDB payload while
SYNCing\n");
```

Improper Resource Access Authorization\Path 33:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=223
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1133	1133
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void slaveMode(void) {

```
....
1133.       fprintf(stderr,"SYNC done. Logging commands from master.\n");
```

Improper Resource Access Authorization\Path 34:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=224
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1153	1153
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void getRDB(void) {

```
....
1153.       fprintf(stderr,"SYNC sent to master, writing %llu bytes to
'%s'\n",
```

Improper Resource Access Authorization\Path 35:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=225
Status	New

	Source	Destination

File	disque/disque-cli.c	disque/disque-cli.c
Line	1162	1162
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void getRDB(void) {

```
....
1162.               fprintf(stderr, "Error opening '%s': %s\n",
config.rdb_filename,
```

Improper Resource Access Authorization\Path 36:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=226
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1173	1173
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void getRDB(void) {

```
....
1173.               fprintf(stderr,"I/O Error reading RDB payload from
socket\n");
```

Improper Resource Access Authorization\Path 37:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=227
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1178	1178
Object	fprintf	fprintf

Code Snippet

File Name	disque/disque-cli.c
Method	static void getRDB(void) {

```
....
1178.               fprintf(stderr,"Error writing data to file: %s\n",
```

Improper Resource Access Authorization\Path 38:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=228
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1186	1186
Object	fprintf	fprintf

Code Snippet

File Name	disque/disque-cli.c
Method	static void getRDB(void) {

```
....
1186.       fprintf(stderr,"Transfer finished with success.\n");
```

Improper Resource Access Authorization\Path 39:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=229
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1210	1210
Object	fprintf	fprintf

Code Snippet

File Name	disque/disque-cli.c
Method	static void pipeMode(void) {

```
....
1210.           fprintf(stderr, "Can't set the socket in non blocking
mode: %s\n",
```

Improper Resource Access Authorization\Path 40:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=230
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1231	1231
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1231.                       fprintf(stderr, "Error reading from the
server: %s\n",
```

Improper Resource Access Authorization\Path 41:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=231
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1244	1244
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1244.                       fprintf(stderr, "Error reading replies from
server\n");
```

Improper Resource Access Authorization\Path 42:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=232
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1249	1249
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1249.                           fprintf(stderr,"%s\n", reply->str);
```

Improper Resource Access Authorization\Path 43:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=233
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1277	1277
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1277.                           fprintf(stderr, "Error writing to the
server: %s\n",
```

Improper Resource Access Authorization\Path 44:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=234
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1313	1313
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1313.                           fprintf(stderr, "Error reading from
stdin: %s\n",
```

Improper Resource Access Authorization\Path 45:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=235
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1331	1331
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1331.               fprintf(stderr,"No replies for %d seconds:
exiting.\n",
```

Improper Resource Access Authorization\Path 46:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=236
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1361	1361
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static redisReply *sendScan(unsigned long long *it) {

```
....
1361.          fprintf(stderr, "\nI/O error\n");
```

Improper Resource Access Authorization\Path 47:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=237
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1364	1364
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static redisReply *sendScan(unsigned long long *it) {

```
....
1364.           fprintf(stderr, "SCAN error: %s\n", reply->str);
```

Improper Resource Access Authorization\Path 48:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=238
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1367	1367
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static redisReply *sendScan(unsigned long long *it) {

```
....
1367.           fprintf(stderr, "Non ARRAY response from SCAN!\n");
```

Improper Resource Access Authorization\Path 49:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=239
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1370	1370
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static redisReply *sendScan(unsigned long long *it) {

```
....
1370.           fprintf(stderr, "Invalid element count from SCAN!\n");
```

Improper Resource Access Authorization\Path 50:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=240
Status New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1391	1391
Object	fprintf	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static int getDbSize(void) {

```
....
1391.           fprintf(stderr, "Couldn't determine DBSIZE!\n");
```

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=1010004&projectid=10005&pathid=5
Status New

The _serverAssertPrintClientInfo method calls the snprintf function, at line 140 of disque/debug.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	disque/debug.c	disque/debug.c
Line	155	155
Object	snprintf	snprintf

Code Snippet
File Name disque/debug.c
Method void _serverAssertPrintClientInfo(client *c) {

```
....
155.              snprintf(buf,sizeof(buf),"Object type: %u, encoding:
%u",
```

Unchecked Return Value\Path 2:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=6
Status New

The cliRefreshPrompt method calls the snprintf function, at line 124 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	138	138
Object	snprintf	snprintf

Code Snippet
File Name disque/disque-cli.c
Method static void cliRefreshPrompt(void) {

```
....
138.      snprintf(config.prompt+len,sizeof(config.prompt)-len,"> ");
```

Unchecked Return Value\Path 3:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=7
Status New

The cliFormatReplyTTY method calls the snprintf function, at line 376 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	421	421
Object	snprintf	snprintf

Code Snippet
File Name disque/disque-cli.c
Method static sds cliFormatReplyTTY(redisReply *r, char *prefix) {

```
....
421.                snprintf(_prefixfmt,sizeof(_prefixfmt),"%%s%%%ud)
",idxlen);
```

Unchecked Return Value\Path 4:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=8
Status New

The bytesToHuman method calls the sprintf function, at line 1660 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1670	1670
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method void bytesToHuman(char *s, long long n) {

```
....
1670.           sprintf(s,"%lldB",n);
```

Unchecked Return Value\Path 5:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=9
Status New

The bytesToHuman method calls the sprintf function, at line 1660 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1674	1674
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method void bytesToHuman(char *s, long long n) {

```
....
1674.           sprintf(s,"%.2fK",d);
```

Unchecked Return Value\Path 6:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=10
Status New

The bytesToHuman method calls the sprintf function, at line 1660 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1677	1677
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method void bytesToHuman(char *s, long long n) {

```
....
1677.           sprintf(s,"%.2fM",d);
```

Unchecked Return Value\Path 7:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=11
Status New

The bytesToHuman method calls the sprintf function, at line 1660 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1680	1680
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method void bytesToHuman(char *s, long long n) {

```
....
1680.           sprintf(s,"%.2fG",d);
```

Unchecked Return Value\Path 8:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=12
Status New

The statMode method calls the sprintf function, at line 1684 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1710	1710
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1710.               sprintf(buf,"db%d:keys",j);
```

Unchecked Return Value\Path 9:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=13
Status New

The statMode method calls the sprintf function, at line 1684 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1715	1715
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1715.           sprintf(buf,"%ld",aux);
```

Unchecked Return Value\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=14
Status	New

The statMode method calls the sprintf function, at line 1684 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1725	1725
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1725.           sprintf(buf,"%ld",aux);
```

Unchecked Return Value\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=15
Status	New

The statMode method calls the sprintf function, at line 1684 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c

Line	1730	1730
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1730.           sprintf(buf,"%ld",aux);
```

Unchecked Return Value\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=16
Status	New

The statMode method calls the sprintf function, at line 1684 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1735	1735
Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1735.           sprintf(buf,"%ld (+%ld)",aux,requests == 0 ? 0 : aux-
requests);
```

Unchecked Return Value\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=17
Status	New

The statMode method calls the sprintf function, at line 1684 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1741	1741

Object	sprintf	sprintf

Code Snippet
File Name disque/disque-cli.c
Method static void statMode(void) {

```
....
1741.           sprintf(buf,"%ld",aux);
```

Unchecked Return Value\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=18
Status	New

The serverLogRaw method calls the snprintf function, at line 155 of disque/server.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	disque/server.c	disque/server.c
Line	179	179
Object	snprintf	snprintf

Code Snippet
File Name disque/server.c
Method void serverLogRaw(int level, const char *msg) {

```
....
179.           snprintf(buf+off,sizeof(buf)-
off,"%03d",(int)tv.tv_usec/1000);
```

Unchecked Return Value\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=19
Status	New

The serverDebug method calls the snprintf function, at line 211 of disque/server.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	disque/server.c	disque/server.c
Line	222	222

Object	snprintf	snprintf

Code Snippet
File Name disque/server.c
Method void serverDebug(const char *fmt, ...) {

```
....
222.      snprintf(buf+off,sizeof(buf)-
off,"%03d:%03d",(int)tv.tv_usec/1000,
```

Unchecked Return Value\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=20
Status	New

The bytesToHuman method calls the sprintf function, at line 1876 of disque/server.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	disque/server.c	disque/server.c
Line	1881	1881
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1881.          sprintf(s,"%lluB",n);
```

Unchecked Return Value\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=21
Status	New

The bytesToHuman method calls the sprintf function, at line 1876 of disque/server.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	disque/server.c	disque/server.c
Line	1885	1885

Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1885.           sprintf(s,"%.2fK",d);
```

Unchecked Return Value\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=22
Status	New

The bytesToHuman method calls the sprintf function, at line 1876 of disque/server.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	disque/server.c	disque/server.c
Line	1888	1888
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1888.           sprintf(s,"%.2fM",d);
```

Unchecked Return Value\Path 19:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=23
Status	New

The bytesToHuman method calls the sprintf function, at line 1876 of disque/server.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	disque/server.c	disque/server.c
Line	1891	1891

Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1891.           sprintf(s,"%.2fG",d);
```

Unchecked Return Value\Path 20:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=24
Status	New

The bytesToHuman method calls the sprintf function, at line 1876 of disque/server.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	disque/server.c	disque/server.c
Line	1894	1894
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1894.           sprintf(s,"%.2fT",d);
```

Unchecked Return Value\Path 21:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=25
Status	New

The bytesToHuman method calls the sprintf function, at line 1876 of disque/server.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	disque/server.c	disque/server.c
Line	1897	1897

Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1897.           sprintf(s,"%.2fP",d);
```

Unchecked Return Value\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=26
Status	New

The bytesToHuman method calls the sprintf function, at line 1876 of disque/server.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	disque/server.c	disque/server.c
Line	1900	1900
Object	sprintf	sprintf

Code Snippet
File Name disque/server.c
Method void bytesToHuman(char *s, unsigned long long n) {

```
....
1900.           sprintf(s,"%lluB",n);
```

Unchecked Return Value\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=27
Status	New

The disqueAsciiArt method calls the snprintf function, at line 2387 of disque/server.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	disque/server.c	disque/server.c
Line	2402	2402

Object	snprintf	snprintf

Code Snippet
File Name disque/server.c
Method void disqueAsciiArt(void) {

```
....
2402.           snprintf(buf,1024*16,ascii_logo,
```

Unchecked Return Value\Path 24:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=28
Status	New

The zmalloc_get_rss method calls the snprintf function, at line 232 of disque/zmalloc.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	disque/zmalloc.c	disque/zmalloc.c
Line	240	240
Object	snprintf	snprintf

Code Snippet
File Name disque/zmalloc.c
Method size_t zmalloc_get_rss(void) {

```
....
240.      snprintf(filename,256,"/proc/%d/stat",getpid());
```

Unchecked Return Value\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=29
Status	New

The cliRefreshPrompt method calls the len function, at line 124 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	128	128

Object	len	len

Code Snippet
File Name disque/disque-cli.c
Method static void cliRefreshPrompt(void) {

```
....
128.          len = snprintf(config.prompt,sizeof(config.prompt),"disque
%s",
```

Unchecked Return Value\Path 26:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=30
Status New

The cliRefreshPrompt method calls the len function, at line 124 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	131	131
Object	len	len

Code Snippet
File Name disque/disque-cli.c
Method static void cliRefreshPrompt(void) {

```
....
131.          len = snprintf(config.prompt,sizeof(config.prompt),
```

Unchecked Return Value\Path 27:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=31
Status New

The cliRefreshPrompt method calls the len function, at line 124 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	136	136

Object	len	len

Code Snippet
File Name disque/disque-cli.c
Method static void cliRefreshPrompt(void) {

```
....
136.          len += snprintf(config.prompt+len,sizeof(config.prompt)-
len,"[%d]",
```

Unchecked Return Value\Path 28:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=32
Status New

The cliSendCommand method calls the argvlen function, at line 593 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	628	628
Object	argvlen	argvlen

Code Snippet
File Name disque/disque-cli.c
Method static int cliSendCommand(int argc, char **argv, int repeat) {

```
....
628.      argvlen = malloc(argc*sizeof(size_t));
```

Unchecked Return Value\Path 29:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=33
Status New

The *getInfoField method calls the result function, at line 1630 of disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1640	1640

Object	result	result

Code Snippet
File Name disque/disque-cli.c
Method static char *getInfoField(char *info, char *field) {

```
....
1640.       result = malloc(sizeof(char)*(n1-p)+1);
```

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=1010004&projectid=10005&pathid=276
Status	New

The system data read by readArgFromStdin in the file disque/disque-cli.c at line 795 is potentially exposed by readArgFromStdin found in disque/disque-cli.c at line 795.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	804	804
Object	perror	perror

Code Snippet
File Name disque/disque-cli.c
Method static sds readArgFromStdin(void) {

```
....
804.                perror("Reading from standard input");
```

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=1010004&projectid=10005&pathid=277
Status	New

The system data read by evalMode in the file disque/disque-cli.c at line 988 is potentially exposed by evalMode found in disque/disque-cli.c at line 988.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1000	999
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static int evalMode(int argc, char **argv) {

```
....
1000.               "Can't open file '%s': %s\n", config.eval,
strerror(errno));
....
999.           fprintf(stderr,
```

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=1010004&projectid=10005&pathid=278
Status New

The system data read by getRDB in the file disque/disque-cli.c at line 1147 is potentially exposed by getRDB found in disque/disque-cli.c at line 1147.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1163	1178
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void getRDB(void) {

```
....
1163.                   strerror(errno));
....
1178.              fprintf(stderr,"Error writing data to file: %s\n",
```

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=1010004&projectid=10005&pathid=279
Status New

The system data read by getRDB in the file disque/disque-cli.c at line 1147 is potentially exposed by getRDB found in disque/disque-cli.c at line 1147.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1179	1178
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void getRDB(void) {

```
....
1179.                   strerror(errno));
....
1178.               fprintf(stderr,"Error writing data to file: %s\n",
```

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=1010004&projectid=10005&pathid=280
Status New

The system data read by getRDB in the file disque/disque-cli.c at line 1147 is potentially exposed by getRDB found in disque/disque-cli.c at line 1147.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1163	1162
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void getRDB(void) {

```
....
1163.                   strerror(errno));
....
1162.               fprintf(stderr, "Error opening '%s': %s\n",
config.rdb_filename,
```

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=1010004&projectid=10005&pathid=281
Status New

The system data read by pipeMode in the file disque/disque-cli.c at line 1194 is potentially exposed by pipeMode found in disque/disque-cli.c at line 1194.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1232	1313
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1232.                           strerror(errno));
....
1313.                           fprintf(stderr, "Error reading from
stdin: %s\n",
```

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=1010004&projectid=10005&pathid=282
Status	New

The system data read by pipeMode in the file disque/disque-cli.c at line 1194 is potentially exposed by pipeMode found in disque/disque-cli.c at line 1194.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1278	1313
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1278.                                   strerror(errno));
....
1313.                           fprintf(stderr, "Error reading from
stdin: %s\n",
```

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=1010004&projectid=10005&pathid=283
Status	New

The system data read by pipeMode in the file disque/disque-cli.c at line 1194 is potentially exposed by pipeMode found in disque/disque-cli.c at line 1194.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1314	1313
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1314.                               strerror(errno));
....
1313.                           fprintf(stderr, "Error reading from
stdin: %s\n",
```

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=1010004&projectid=10005&pathid=284
Status New

The system data read by pipeMode in the file disque/disque-cli.c at line 1194 is potentially exposed by pipeMode found in disque/disque-cli.c at line 1194.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1232	1231
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1232.                               strerror(errno));
....
1231.                           fprintf(stderr, "Error reading from the
server: %s\n",
```

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=1010004&projectid=10005&pathid=285

Status	New

The system data read by pipeMode in the file disque/disque-cli.c at line 1194 is potentially exposed by pipeMode found in disque/disque-cli.c at line 1194.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1278	1231
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1278.                                   strerror(errno));
....
1231.                           fprintf(stderr, "Error reading from the
server: %s\n",
```

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=1010004&projectid=10005&pathid=286
Status	New

The system data read by pipeMode in the file disque/disque-cli.c at line 1194 is potentially exposed by pipeMode found in disque/disque-cli.c at line 1194.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1314	1231
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1314.                                   strerror(errno));
....
1231.                           fprintf(stderr, "Error reading from the
server: %s\n",
```

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=1010004&projectid=10005&pathid=287
Status	New

The system data read by pipeMode in the file disque/disque-cli.c at line 1194 is potentially exposed by pipeMode found in disque/disque-cli.c at line 1194.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1232	1277
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1232.                           strerror(errno));
....
1277.                               fprintf(stderr, "Error writing to the
server: %s\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=1010004&projectid=10005&pathid=288
Status	New

The system data read by pipeMode in the file disque/disque-cli.c at line 1194 is potentially exposed by pipeMode found in disque/disque-cli.c at line 1194.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1278	1277
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1278.                                   strerror(errno));
....
1277.                               fprintf(stderr, "Error writing to the
server: %s\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=1010004&projectid=10005&pathid=289
Status	New

The system data read by pipeMode in the file disque/disque-cli.c at line 1194 is potentially exposed by pipeMode found in disque/disque-cli.c at line 1194.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1314	1277
Object	errno	fprintf

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1314.                                   strerror(errno));
....
1277.                                   fprintf(stderr, "Error writing to the
server: %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=1010004&projectid=10005&pathid=290
Status	New

The memtest_test_linux_anonymous_maps method in disque/debug.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	disque/debug.c	disque/debug.c
Line	440	440
Object	fopen	fopen

Code Snippet
File Name disque/debug.c
Method int memtest_test_linux_anonymous_maps(void) {

```
....
440.      FILE *fp = fopen("/proc/self/maps","r");
```

TOCTOU\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=291
Status	New

The evalMode method in disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	997	997
Object	fopen	fopen

Code Snippet
File Name disque/disque-cli.c
Method static int evalMode(int argc, char **argv) {

```
....
997.      fp = fopen(config.eval,"r");
```

TOCTOU\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=292
Status	New

The serverLogRaw method in disque/server.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	disque/server.c	disque/server.c
Line	166	166
Object	fopen	fopen

Code Snippet
File Name disque/server.c
Method void serverLogRaw(int level, const char *msg) {

```
....
166.      fp = log_to_stdout ? stdout : fopen(server.logfile,"a");
```

TOCTOU\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=293
Status	New

The serverDebug method in disque/server.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	disque/server.c	disque/server.c
Line	216	216
Object	fopen	fopen

Code Snippet
File Name disque/server.c
Method void serverDebug(const char *fmt, ...) {

```
....
216.      FILE *fp = fopen("/tmp/disque.dbg","a");
```

TOCTOU\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=294
Status	New

The checkTcpBacklogSettings method in disque/server.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	disque/server.c	disque/server.c
Line	1200	1200
Object	fopen	fopen

Code Snippet
File Name disque/server.c
Method void checkTcpBacklogSettings(void) {

```
....
1200.       FILE *fp = fopen("/proc/sys/net/core/somaxconn","r");
```

TOCTOU\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=295
Status	New

The linuxOvercommitMemoryValue method in disque/server.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	disque/server.c	disque/server.c
Line	2315	2315
Object	fopen	fopen

Code Snippet
File Name disque/server.c
Method int linuxOvercommitMemoryValue(void) {

```
....
2315.       FILE *fp = fopen("/proc/sys/vm/overcommit_memory","r");
```

TOCTOU\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=296
Status	New

The createPidFile method in disque/server.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	disque/server.c	disque/server.c
Line	2337	2337
Object	fopen	fopen

Code Snippet
File Name disque/server.c
Method void createPidFile(void) {

```
....
2337.       FILE *fp = fopen(server.pidfile,"w");
```

TOCTOU\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=297
Status	New

The zmalloc_get_private_dirty method in disque/zmalloc.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	disque/zmalloc.c	disque/zmalloc.c
Line	303	303
Object	fopen	fopen

Code Snippet
File Name disque/zmalloc.c
Method size_t zmalloc_get_private_dirty(void) {

```
....
303.       FILE *fp = fopen("/proc/self/smaps","r");
```

TOCTOU\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=298
Status	New

The logStackTrace method in disque/debug.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	disque/debug.c	disque/debug.c
Line	394	394
Object	open	open

Code Snippet
File Name disque/debug.c
Method void logStackTrace(ucontext_t *uc) {

```
....
394.          open(server.logfile, O_APPEND|O_CREAT|O_WRONLY, 0644);
```

TOCTOU\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=299
Status	New

The getRDB method in disque/disque-cli.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	disque/disque-cli.c	disque/disque-cli.c
Line	1160	1160
Object	open	open

Code Snippet
File Name disque/disque-cli.c
Method static void getRDB(void) {

```
....
1160.          fd = open(config.rdb_filename, O_CREAT|O_WRONLY, 0644);
```

TOCTOU\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=300
Status	New

The serverLogFromHandler method in disque/server.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	disque/server.c	disque/server.c
Line	248	248
Object	open	open

Code Snippet
File Name disque/server.c
Method void serverLogFromHandler(int level, const char *msg) {

```
....
248.                            open(server.logfile,
O_APPEND|O_CREAT|O_WRONLY, 0644);
```

TOCTOU\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=301
Status	New

The initServer method in disque/server.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	disque/server.c	disque/server.c
Line	1396	1396
Object	open	open

Code Snippet
File Name disque/server.c
Method void initServer(void) {

```
....
1396.          server.aof_fd = open(server.aof_filename,
```

TOCTOU\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=302
Status	New

The daemonize method in disque/server.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	disque/server.c	disque/server.c
Line	2353	2353
Object	open	open

Code Snippet
File Name disque/server.c
Method void daemonize(void) {

```
....
2353.       if ((fd = open("/dev/null", O_RDWR, 0)) != -1) {
```

TOCTOU\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=303
Status	New

The zmalloc_get_rss method in disque/zmalloc.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	disque/zmalloc.c	disque/zmalloc.c
Line	241	241
Object	open	open

Code Snippet
File Name disque/zmalloc.c
Method size_t zmalloc_get_rss(void) {

```
....
241.       if ((fd = open(filename,O_RDONLY)) == -1) return 0;
```

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=1010004&projectid=10005&pathid=93
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	975	975
Object	argc	argc

Code Snippet

File Name	disque/disque-cli.c
Method	static int noninteractive(int argc, char **argv) {

```
....
975.          argv[argc] = readArgFromStdin();
```

Unchecked Array Index\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=94
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1823	1823
Object	i	i

Code Snippet	
File Name	disque/disque-cli.c
Method	unsigned long compute_something_fast(void) {

```
....
1823.          s[i] = s[j];
```

Unchecked Array Index\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=95
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1824	1824
Object	j	j

Code Snippet	
File Name	disque/disque-cli.c
Method	unsigned long compute_something_fast(void) {

```
....
1824.          s[j] = t;
```

Unchecked Array Index\Path 4:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=96
Status	New

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	350	350
Object	len	len

Code Snippet
File Name disque/sds.c
Method void sdsIncrLen(sds s, int incr) {

```
....
350.        s[len] = '\0';
```

Unchecked Array Index\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=97
Status	New

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	665	665
Object	i	i

Code Snippet
File Name disque/sds.c
Method sds sdscatfmt(sds s, char const *fmt, ...) {

```
....
665.        s[i] = '\0';
```

Unchecked Array Index\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=98
Status	New

	Source	Destination
File	disque/sds.c	disque/sds.c

Line	693	693
Object	len	len

Code Snippet
File Name disque/sds.c
Method sds sdstrim(sds s, const char *cset) {

```
....
693.      s[len] = '\0';
```

Unchecked Array Index\Path 7:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=99
Status New

	Source	Destination
File	disque/server.c	disque/server.c
Line	589	589
Object	idx	idx

Code Snippet
File Name disque/server.c
Method void trackInstantaneousMetric(int metric, long long current_reading) {

```
....
589.
server.inst_metric[metric].samples[server.inst_metric[metric].idx] =
```

Unchecked Array Index\Path 8:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=100
Status New

	Source	Destination
File	disque/server.c	disque/server.c
Line	978	978
Object	CONFIG_RUN_ID_SIZE	CONFIG_RUN_ID_SIZE

Code Snippet
File Name disque/server.c
Method void initServerConfig(void) {

```
....
978.      server.runid[CONFIG_RUN_ID_SIZE] = '\0';
```

Unchecked Array Index\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=101
Status	New

	Source	Destination
File	disque/server.c	disque/server.c
Line	589	589
Object	metric	metric

Code Snippet
File Name disque/server.c
Method void trackInstantaneousMetric(int metric, long long current_reading) {

```
....
589.
server.inst_metric[metric].samples[server.inst_metric[metric].idx] =
```

Unchecked Array Index\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=102
Status	New

	Source	Destination
File	disque/server.c	disque/server.c
Line	589	589
Object	metric	metric

Code Snippet
File Name disque/server.c
Method void trackInstantaneousMetric(int metric, long long current_reading) {

```
....
589.
server.inst_metric[metric].samples[server.inst_metric[metric].idx] =
```

Use of Sizeof On a Pointer Type

Query Path:

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=1010004&projectid=10005&pathid=34
Status	New

	Source	Destination
File	disque/debug.c	disque/debug.c
Line	108	108
Object	sizeof	sizeof

Code Snippet
File Name disque/debug.c
Method void debugCommand(client *c) {

```
....
108.          sizes = sdscatprintf(sizes,"bits:%d ",(sizeof(void*) ==
8)?64:32);
```

Use of Sizeof On a Pointer Type\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=35
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	177	177
Object	sizeof	sizeof

Code Snippet
File Name disque/disque-cli.c
Method static void cliInitHelp(void) {

```
....
177.      int groupslen = sizeof(commandGroups)/sizeof(char*);
```

Use of Sizeof On a Pointer Type\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=36

Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	238	238
Object	sizeof	sizeof

Code Snippet
File Name disque/disque-cli.c
Method static void cliOutputHelp(int argc, char **argv) {

```
....
238.          len = sizeof(commandGroups)/sizeof(char*);
```

Use of Sizeof On a Pointer Type\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=37
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	872	872
Object	sizeof	sizeof

Code Snippet
File Name disque/disque-cli.c
Method static char **convertToSds(int count, char** args) {

```
....
872.     char **sds = zmalloc(sizeof(char*)*count);
```

Use of Sizeof On a Pointer Type\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=38
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	974	974
Object	sizeof	sizeof

Code Snippet
File Name disque/disque-cli.c
Method static int noninteractive(int argc, char **argv) {

```
....
974.          argv = zrealloc(argv, (argc+1)*sizeof(char*));
```

Use of Sizeof On a Pointer Type\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=39
Status	New

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	1023	1023
Object	sizeof	sizeof

Code Snippet
File Name disque/sds.c
Method sds *sdssplitargs(const char *line, int *argc) {

```
....
1023.              vector = s_realloc(vector,((*argc)+1)*sizeof(char*));
```

Use of Sizeof On a Pointer Type\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=40
Status	New

	Source	Destination
File	disque/sds.c	disque/sds.c
Line	1029	1029
Object	sizeof	sizeof

Code Snippet
File Name disque/sds.c
Method sds *sdssplitargs(const char *line, int *argc) {

```
....
1029.              if (vector == NULL) vector = s_malloc(sizeof(void*));
```

Use of Sizeof On a Pointer Type\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=41
Status	New

	Source	Destination
File	disque/server.c	disque/server.c
Line	2526	2526
Object	sizeof	sizeof

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

```
....
2526.       server.exec_argv = zmalloc(sizeof(char*)*(argc+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=1010004&projectid=10005&pathid=269
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	997	997
Object	fp	fp

Code Snippet
File Name disque/disque-cli.c
Method static int evalMode(int argc, char **argv) {

```
....
997.       fp = fopen(config.eval,"r");
```

Incorrect Permission Assignment For Critical Resources\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=270
Status	New

	Source	Destination
File	disque/debug.c	disque/debug.c
Line	440	440
Object	fp	fp

Code Snippet
File Name disque/debug.c
Method int memtest_test_linux_anonymous_maps(void) {

```
....
440.      FILE *fp = fopen("/proc/self/maps","r");
```

Incorrect Permission Assignment For Critical Resources\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=271
Status	New

	Source	Destination
File	disque/server.c	disque/server.c
Line	216	216
Object	fp	fp

Code Snippet
File Name disque/server.c
Method void serverDebug(const char *fmt, ...) {

```
....
216.      FILE *fp = fopen("/tmp/disque.dbg","a");
```

Incorrect Permission Assignment For Critical Resources\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=272
Status	New

	Source	Destination
File	disque/server.c	disque/server.c
Line	1200	1200
Object	fp	fp

Code Snippet
File Name disque/server.c
Method void checkTcpBacklogSettings(void) {

```
....
1200.       FILE *fp = fopen("/proc/sys/net/core/somaxconn","r");
```

Incorrect Permission Assignment For Critical Resources\Path 5:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=273
Status New

	Source	Destination
File	disque/server.c	disque/server.c
Line	2315	2315
Object	fp	fp

Code Snippet
File Name disque/server.c
Method int linuxOvercommitMemoryValue(void) {

```
....
2315.       FILE *fp = fopen("/proc/sys/vm/overcommit_memory","r");
```

Incorrect Permission Assignment For Critical Resources\Path 6:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=274
Status New

	Source	Destination
File	disque/server.c	disque/server.c
Line	2337	2337
Object	fp	fp

Code Snippet

File Name	disque/server.c
Method	void createPidFile(void) {

```
....
2337.       FILE *fp = fopen(server.pidfile,"w");
```

Incorrect Permission Assignment For Critical Resources\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=275
Status	New

	Source	Destination
File	disque/zmalloc.c	disque/zmalloc.c
Line	303	303
Object	fp	fp

Code Snippet

File Name	disque/zmalloc.c
Method	size_t zmalloc_get_private_dirty(void) {

```
....
303.       FILE *fp = fopen("/proc/self/smaps","r");
```

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=1010004&projectid=10005&pathid=67
Status	New

The variable declared in null at disque/queue.c in line 738 is not initialized when it is used by j at disque/queue.c in line 120.

	Source	Destination
File	disque/queue.c	disque/queue.c
Line	790	135

Object	null	j

Code Snippet
File Name disque/queue.c
Method void getjobCommand(client *c) {

```
....
790.                job *job = NULL;
```

File Name disque/queue.c

Method void addReplyJob(client *c, job *j, int flags) {

```
....
135.            addReplyLongLong(c,j->num_deliv);
```

NULL Pointer Dereference\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=68
Status	New

The variable declared in null at disque/queue.c in line 738 is not initialized when it is used by j at disque/queue.c in line 120.

	Source	Destination
File	disque/queue.c	disque/queue.c
Line	790	132
Object	null	j

Code Snippet
File Name disque/queue.c
Method void getjobCommand(client *c) {

```
....
790.                job *job = NULL;
```

File Name disque/queue.c

Method void addReplyJob(client *c, job *j, int flags) {

```
....
132.            addReplyLongLong(c,j->num_nacks);
```

NULL Pointer Dereference\Path 3:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=69
Status	New

The variable declared in null at disque/queue.c in line 738 is not initialized when it is used by j at disque/queue.c in line 120.

	Source	Destination
File	disque/queue.c	disque/queue.c
Line	790	128
Object	null	j

Code Snippet

File Name disque/queue.c

Method void getjobCommand(client *c) {

```
....
790.              job *job = NULL;
```

File Name disque/queue.c

Method void addReplyJob(client *c, job *j, int flags) {

```
....
128.      addReplyBulkCBuffer(c,j->body,sdslen(j->body));
```

NULL Pointer Dereference\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=70
Status	New

The variable declared in null at disque/queue.c in line 738 is not initialized when it is used by j at disque/queue.c in line 120.

	Source	Destination
File	disque/queue.c	disque/queue.c
Line	790	128
Object	null	j

Code Snippet

File Name disque/queue.c

Method void getjobCommand(client *c) {

```
....
790.              job *job = NULL;
```

File Name disque/queue.c

Method void addReplyJob(client *c, job *j, int flags) {

```
....
128.      addReplyBulkCBuffer(c,j->body,sdslen(j->body));
```

NULL Pointer Dereference\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=71
Status	New

The variable declared in null at disque/queue.c in line 738 is not initialized when it is used by j at disque/queue.c in line 120.

	Source	Destination
File	disque/queue.c	disque/queue.c
Line	790	126
Object	null	j

Code Snippet

File Name disque/queue.c

Method void getjobCommand(client *c) {

```
....
790.              job *job = NULL;
```

File Name disque/queue.c

Method void addReplyJob(client *c, job *j, int flags) {

```
....
126.      addReplyBulk(c,j->queue);
```

NULL Pointer Dereference\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=72
Status	New

The variable declared in null at disque/queue.c in line 738 is not initialized when it is used by j at disque/queue.c in line 120.

	Source	Destination
File	disque/queue.c	disque/queue.c
Line	790	127
Object	null	j

Code Snippet
File Name disque/queue.c
Method void getjobCommand(client *c) {

```
....
790.              job *job = NULL;
```

File Name disque/queue.c
Method void addReplyJob(client *c, job *j, int flags) {

```
....
127.      addReplyBulkCBuffer(c,j->id,JOB_ID_LEN);
```

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=1010004&projectid=10005&pathid=73
Status	New

The size of the buffer used by slaveMode in >, at line 1113 of disque/disque-cli.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that slaveMode passes to buf, at line 1113 of disque/disque-cli.c, to overwrite the target buffer.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1126	1126
Object	buf	>

Code Snippet

File Name	disque/disque-cli.c
Method	static void slaveMode(void) {

```
....
1126.           nread = read(fd,buf,(payload > sizeof(buf)) ? sizeof(buf)
: payload);
```

Heuristic 2nd Order Buffer Overflow read\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=74
Status	New

The size of the buffer used by slaveMode in payload, at line 1113 of disque/disque-cli.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that slaveMode passes to buf, at line 1113 of disque/disque-cli.c, to overwrite the target buffer.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1126	1126
Object	buf	payload

Code Snippet	
File Name	disque/disque-cli.c
Method	static void slaveMode(void) {

```
....
1126.           nread = read(fd,buf,(payload > sizeof(buf)) ? sizeof(buf)
: payload);
```

Heuristic 2nd Order Buffer Overflow read\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=75
Status	New

The size of the buffer used by getRDB in >, at line 1147 of disque/disque-cli.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that getRDB passes to buf, at line 1147 of disque/disque-cli.c, to overwrite the target buffer.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1171	1171
Object	buf	>

Code Snippet	
File Name	disque/disque-cli.c

Method	static void getRDB(void) {

```
....
1171.           nread = read(s,buf,(payload > sizeof(buf)) ? sizeof(buf)
: payload);
```

Heuristic 2nd Order Buffer Overflow read\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=76
Status	New

The size of the buffer used by getRDB in payload, at line 1147 of disque/disque-cli.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that getRDB passes to buf, at line 1147 of disque/disque-cli.c, to overwrite the target buffer.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1171	1171
Object	buf	payload

Code Snippet	
File Name	disque/disque-cli.c
Method	static void getRDB(void) {

```
....
1171.           nread = read(s,buf,(payload > sizeof(buf)) ? sizeof(buf)
: payload);
```

Heuristic 2nd Order Buffer Overflow read\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=77
Status	New

The size of the buffer used by slaveMode in payload, at line 1113 of disque/disque-cli.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that slaveMode passes to buf, at line 1113 of disque/disque-cli.c, to overwrite the target buffer.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1126	1126
Object	buf	payload

Code Snippet	
File Name	disque/disque-cli.c
Method	static void slaveMode(void) {

```
....
1126.           nread = read(fd,buf,(payload > sizeof(buf)) ? sizeof(buf)
: payload);
```

Heuristic 2nd Order Buffer Overflow read\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=78
Status	New

The size of the buffer used by getRDB in payload, at line 1147 of disque/disque-cli.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that getRDB passes to buf, at line 1147 of disque/disque-cli.c, to overwrite the target buffer.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1171	1171
Object	buf	payload

Code Snippet
File Name disque/disque-cli.c
Method static void getRDB(void) {

```
....
1171.           nread = read(s,buf,(payload > sizeof(buf)) ? sizeof(buf)
: payload);
```

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=1010004&projectid=10005&pathid=1
Status	New

Method pipeMode at line 1194 of disque/disque-cli.c uses a weak method rand to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

Source	Destination

File	disque/disque-cli.c	disque/disque-cli.c
Line	1306	1306
Object	rand	rand

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1306.                               magic[j] = rand() & 0xff;
```

Use of Insufficiently Random Values\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=2
Status	New

Method randomTimeError at line 281 of disque/server.c uses a weak method rand to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	disque/server.c	disque/server.c
Line	282	282
Object	rand	rand

Code Snippet
File Name disque/server.c
Method mstime_t randomTimeError(mstime_t milliseconds) {

```
....
282.      return rand()%milliseconds - milliseconds/2;
```

Use of Insufficiently Random Values\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=3
Status	New

Method pipeMode at line 1194 of disque/disque-cli.c uses a weak method srand 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	disque/disque-cli.c	disque/disque-cli.c

Line	1206	1206
Object	srand	srand

Code Snippet
File Name disque/disque-cli.c
Method static void pipeMode(void) {

```
....
1206.       srand(time(NULL));
```

Use of Insufficiently Random Values\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=4
Status	New

Method main at line 2507 of disque/server.c uses a weak method srand 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	disque/server.c	disque/server.c
Line	2518	2518
Object	srand	srand

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

```
....
2518.       srand(time(NULL)^getpid());
```

Heuristic Buffer Overflow malloc

Query Path:
CPP\Cx\CPP Heuristic\Heuristic Buffer Overflow malloc Version:0

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
NIST SP 800-53: SI-10 Information Input Validation (P1)
OWASP Top 10 2017: A1-Injection

Description

Heuristic Buffer Overflow malloc\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=79
Status	New

The size of the buffer used by cliSendCommand in argc, at line 593 of disque/disque-cli.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 1876 of disque/disque-cli.c, to overwrite the target buffer.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1876	628
Object	argc	argc

Code Snippet
File Name disque/disque-cli.c
Method int main(int argc, char **argv) {

```
....
1876.  int main(int argc, char **argv) {
```

File Name disque/disque-cli.c
Method static int cliSendCommand(int argc, char **argv, int repeat) {

```
....
628.      argvlen = malloc(argc*sizeof(size_t));
```

Heuristic Buffer Overflow malloc\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=80
Status	New

The size of the buffer used by cliSendCommand in BinaryExpr, at line 593 of disque/disque-cli.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 1876 of disque/disque-cli.c, to overwrite the target buffer.

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	1876	628
Object	argc	BinaryExpr

Code Snippet
File Name disque/disque-cli.c
Method int main(int argc, char **argv) {

```
....
1876.  int main(int argc, char **argv) {
```

File Name	disque/disque-cli.c
Method	static int cliSendCommand(int argc, char **argv, int repeat) {

```
....
628.      argvlen = malloc(argc*sizeof(size_t));
```

Sizeof Pointer Argument

Query Path:
CPP\Cx\CPP Low Visibility\Sizeof Pointer Argument Version:0

Description

Sizeof Pointer Argument\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1010004&projectid=10005&pathid=92
Status	New

	Source	Destination
File	disque/disque-cli.c	disque/disque-cli.c
Line	238	238
Object	commandGroups	sizeof

Code Snippet

File Name	disque/disque-cli.c
Method	static void cliOutputHelp(int argc, char **argv) {

```
....
238.          len = sizeof(commandGroups)/sizeof(char*);
```

Improper Null Termination

Weakness ID: 170 *(Weakness Base)* **Status:** Incomplete

Description

Description Summary

The software does not terminate or incorrectly terminates a string or array with a null character or equivalent terminator.

Extended Description

Null termination errors frequently occur in two different ways. An off-by-one error could cause a null to be written out of bounds, leading to an overflow. Or, a program could use a strncpy() function call incorrectly, which prevents a null terminator from being added at all. Other scenarios are possible.

Time of Introduction

- Implementation

Applicable Platforms

Languages

C

C++

Platform Notes

Conceptually, this does not just apply to the C language; any language or representation that involves a terminator could have this type of problem.

Common Consequences

Scope	Effect
Confidentiality Integrity	The case of an omitted null character is the most dangerous of the possible issues. This will almost certainly result in information disclosure, and possibly a buffer overflow condition, which may be exploited to execute arbitrary code.
Confidentiality Integrity Availability	If a null character is omitted from a string, then most string-copying functions will read data until they locate a null character, even outside of the intended boundaries of the string. This could: • cause a crash due to a segmentation fault • cause sensitive adjacent memory to be copied and sent to an outsider • trigger a buffer overflow when the copy is bering written to a fixed-size buffer
Integrity Availability	Misplaced null characters may result in any number of security problems. The biggest issue is a subset of buffer overflow, and write-what-where conditions, where data corruption occurs from the writing of a null character over valid data, or even instructions. A randomly placed null character may put the system into an undefined state, and therefore make it prone to crashing. A misplaced null character may corrupt other data in memory
Access Control	Should the null character corrupt the process flow, or affect a flag controlling access, it may lead to logical errors which allow for the execution of arbitrary code.

Likelihood of Exploit

Medium

Demonstrative Examples

Example 1

The following code reads from cfgfile and copies the input into inputbuf using strcpy(). The code mistakenly assumes that inputbuf will always contain a NULL terminator.

(Bad Code)

Example Language: **C**

```
#define MAXLEN 1024
...
char *pathbuf[MAXLEN];
...
read(cfgfile,inputbuf,MAXLEN); //does not null terminate
strcpy(pathbuf,input_buf); //requires null terminated input
...
```

The code above will behave correctly if the data read from cfgfile is null terminated on disk as expected. But if an attacker is able to modify this input so that it does not contain the expected NULL character, the call to strcpy() will continue copying from memory until it encounters an arbitrary NULL character. This will likely overflow the destination buffer and, if the attacker can control the contents of memory immediately following inputbuf, can leave the application susceptible to a buffer overflow attack.

Example 2

In the following code, readlink() expands the name of a symbolic link stored in the buffer path so that the buffer filename contains the absolute path of the file referenced by the symbolic link. The length of the resulting value is then calculated using strlen().

(Bad Code)

Example Language: **C**

```
char buf[MAXPATH];
...
readlink(path, buf, MAXPATH);
int length = strlen(filename);
...
```

The code above will not behave correctly because the value read into buf by readlink() will not be null terminated. In testing, vulnerabilities like this one might not be caught because the unused contents of buf and the memory immediately following it may be NULL, thereby causing strlen() to appear as if it is behaving correctly. However, in the wild strlen() will continue traversing memory until it encounters an arbitrary NULL character on the stack, which results in a value of length that is much larger than the size of buf and may cause a buffer overflow in subsequent uses of this value. Buffer overflows aside, whenever a single call to readlink() returns the same value that has been passed to its third argument, it is impossible to know whether the name is precisely that many bytes long, or whether readlink() has truncated the name to avoid overrunning the buffer. Traditionally, strings are represented as a region of memory containing data terminated with a NULL character. Older string-handling methods frequently rely on this NULL character to determine the length of the string. If a buffer that does not contain a NULL terminator is passed to one of these functions, the function will read past the end of the buffer. Malicious users typically exploit this type of vulnerability by injecting data with unexpected size or content into the application. They may provide the malicious input either directly as input to the program or indirectly by modifying application resources, such as configuration files. In the event that an attacker causes the application to read beyond the bounds of a buffer, the attacker may be able use a resulting buffer overflow to inject and execute arbitrary code on the system.

Example 3

While the following example is not exploitable, it provides a good example of how nulls can be omitted or misplaced, even when "safe" functions are used:

(Bad Code)

Example Language: **C**

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

int main() {

char longString[] = "String signifying nothing";
char shortString[16];

strncpy(shortString, longString, 16);
printf("The last character in shortString is: %c %1$x\n", shortString[15]);
return (0);
}
```

The above code gives the following output: The last character in shortString is: l 6c So, the shortString array does not end in a NULL character, even though the "safe" string function strncpy() was used.

Observed Examples

Reference	Description
CVE-2000-0312	Attacker does not null-terminate argv[] when invoking another program.
CVE-2003-0777	Interrupted step causes resultant lack of null termination.
CVE-2004-1072	Fault causes resultant lack of null termination, leading to buffer expansion.

CVE-2001-1389	Multiple vulnerabilities related to improper null termination.
CVE-2003-0143	Product does not null terminate a message buffer after snprintf-like call, leading to overflow.

Potential Mitigations

Phase: Requirements

Use a language that is not susceptible to these issues. However, be careful of null byte interaction errors (CWE-626) with lower-level constructs that may be written in a language that is susceptible.

Phase: Implementation

Ensure that all string functions used are understood fully as to how they append null characters. Also, be wary of off-by-one errors when appending nulls to the end of strings.

Phase: Implementation

If performance constraints permit, special code can be added that validates null-termination of string buffers, this is a rather naive and error-prone solution.

Phase: Implementation

Switch to bounded string manipulation functions. Inspect buffer lengths involved in the buffer overrun trace reported with the defect.

Phase: Implementation

Add code that fills buffers with nulls (however, the length of buffers still needs to be inspected, to ensure that the non null-terminated string is not written at the physical end of the buffer).

Weakness Ordinalities

Ordinality	Description
Resultant	*(where the weakness is typically related to the presence of some other weaknesses)*

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	20	Improper Input Validation	**Seven Pernicious Kingdoms (primary)700**
ChildOf	Category	169	Technology-Specific Special Elements	**Development Concepts (primary)699**
ChildOf	Weakness Class	707	Improper Enforcement of Message or Data Structure	**Research Concepts (primary)1000**
ChildOf	Category	730	OWASP Top Ten 2004 Category A9 - Denial of Service	**Weaknesses in OWASP Top Ten (2004) (primary)711**
ChildOf	Category	741	CERT C Secure Coding Section 07 - Characters and Strings (STR)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	748	CERT C Secure Coding Section 50 - POSIX (POS)	Weaknesses Addressed by the CERT C Secure Coding Standard734
CanPrecede	Weakness Base	120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')	Research Concepts1000
CanPrecede	Weakness Variant	126	Buffer Over-read	Research Concepts1000
PeerOf	Weakness Base	463	Deletion of Data Structure Sentinel	Research Concepts1000
PeerOf	Weakness Base	464	Addition of Data Structure Sentinel	Research Concepts1000
CanAlsoBe	Weakness Variant	147	Improper Neutralization of Input Terminators	Research Concepts1000
MemberOf	View	630	Weaknesses Examined by SAMATE	**Weaknesses Examined by SAMATE (primary)630**
CanFollow	Weakness Base	193	Off-by-one Error	Research Concepts1000
CanFollow	Weakness Class	682	Incorrect Calculation	Research Concepts1000

Relationship Notes

Factors: this is usually resultant from other weaknesses such as off-by-one errors, but it can be primary to boundary condition violations such as buffer overflows. In buffer overflows, it can act as an expander for assumed-immutable data.

Overlaps missing input terminator.

f Causal Nature

Explicit

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
PLOVER			Improper Null Termination
7 Pernicious Kingdoms			String Termination Error
CLASP			Miscalculated null termination
OWASP Top Ten 2004	A9	CWE More Specific	Denial of Service
CERT C Secure Coding	POS30-C		Use the readlink() function properly
CERT C Secure Coding	STR03-C		Do not inadvertently truncate a null-terminated byte string
CERT C Secure Coding	STR32-C		Null-terminate byte strings as required

White Box Definitions

A weakness where the code path has:

1. end statement that passes a data item to a null-terminated string function
2. start statement that produces the improper null-terminated data item

Where "produces" is defined through the following scenarios:

1. data item never ended with null-terminator
2. null-terminator is re-written

Maintenance Notes

As currently described, this entry is more like a category than a weakness.

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		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Causal Nature, Common Consequences, Description, Likelihood of Exploit, Maintenance Notes, Relationships, Other Notes, Relationship 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 Common Consequences		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-07-17	KDM Analytics		External
	Improved the White Box Definition		
2009-07-27	CWE Content Team	MITRE	Internal
	updated Common Consequences, Other Notes, Potential Mitigations, White Box Definitions		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Description		

BACK TO TOP

Buffer Overflow IndexFromInput

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Divide By Zero

Risk

What might happen

When a program divides a number by zero, an exception will be raised. If this exception is not handled by the application, unexpected results may occur, including crashing the application. This can be considered a DoS (Denial of Service) attack, if an external user has control of the value of the denominator or can cause this error to occur.

Cause

How does it happen

The program receives an unexpected value, and uses it for division without filtering, validation, or verifying that the value is not zero. The application does not explicitly handle this error or prevent division by zero from occuring.

General Recommendations

How to avoid it

- Before dividing by an unknown value, validate the number and explicitly ensure it does not evaluate to zero.
- Validate all untrusted input from all sources, in particular verifying that it is not zero before dividing with it.
- Verify output of methods, calculations, dictionary lookups, and so on, and ensure it is not zero before dividing with the result.
- Ensure divide-by-zero errors are caught and handled appropriately.

Source Code Examples

Java

Divide by Zero

```
public float getAverage(HttpServletRequest req) {
      int total = Integer.parseInt(req.getParameter("total"));
      int count = Integer.parseInt(req.getParameter("count"));

      return total / count;
}
```

Checked Division

```
public float getAverage(HttpServletRequest req) {
      int total = Integer.parseInt(req.getParameter("total"));
      int count = Integer.parseInt(req.getParameter("count"));

      if (count > 0)
            return total / count;
      else
```

```
            return 0;
}
```

Buffer Overflow AddressOfLocalVarReturned

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

---

Source Code Examples

CPP

Use of Variable after It was Freed

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

Use of Pointer to Local Variable That Was Freed On Return

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

void func2()

{

	int j;
	j = 5;
```

```
}

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

Buffer Overflow boundcpy WrongSizeParam

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

CPP

Overflowing Buffers

```
const int BUFFER_SIZE = 10;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
{
    strcpy(buffer, inputString);
}
```

Checked Buffers

```
const int BUFFER_SIZE = 10;
const int MAX_INPUT_SIZE = 256;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
```

```
{
    if (strnlen(inputString, MAX_INPUT_SIZE) < sizeof(buffer))
    {
        strncpy(buffer, inputString, sizeof(buffer));
    }
}
```

MemoryFree on StackVariable

Risk

What might happen

Undefined Behavior may result with a crash. Crashes may give an attacker valuable information about the system and the program internals. Furthermore, it may leave unprotected files (e.g memory) that may be exploited.

Cause

How does it happen

Calling free() on a variable that was not dynamically allocated (e.g. malloc) will result with an Undefined Behavior.

General Recommendations

How to avoid it

Use free() only on dynamically allocated variables in order to prevent unexpected behavior from the compiler.

Source Code Examples

CPP

Bad - Calling free() on a static variable

```
void clean_up(){
  char temp[256];
  do_something();
  free(tmp);
  return;
}
```

Good - Calling free() only on variables that were dynamically allocated

```
void clean_up(){
  char *buff;
  buff = (char*) malloc(1024);
  free(buff);
  return;
}
```

Char 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;
}
```

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

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;
}
```

Double Free

Weakness ID: 415 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The product calls free() twice on the same memory address, potentially leading to modification of unexpected memory locations.

Extended Description

When a program calls free() twice with the same argument, the program's memory management data structures become corrupted. This corruption can cause the program to crash or, in some circumstances, cause two later calls to malloc() to return the same pointer. If malloc() returns the same value twice and the program later gives the attacker control over the data that is written into this doubly-allocated memory, the program becomes vulnerable to a buffer overflow attack.

Alternate Terms

Double-free

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C

C++

Common Consequences

Scope	Effect
Access Control	Doubly freeing memory may result in a write-what-where condition, allowing an attacker to execute arbitrary code.

Likelihood of Exploit

Low to Medium

Demonstrative Examples

Example 1

The following code shows a simple example of a double free vulnerability.

(Bad Code)

Example Language: **C**

```
char* ptr = (char*)malloc (SIZE);
...
if (abrt) {
free(ptr);
}
...
free(ptr);
```

Double free vulnerabilities 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

Although some double free vulnerabilities are not much more complicated than the previous example, most are spread out across hundreds of lines of code or even different files. Programmers seem particularly susceptible to freeing global variables

more than once.

Example 2

While contrived, this code should be exploitable on Linux distributions which do not ship with heap-chunk check summing turned on.

(Bad Code)

Example Language: **C**

```
#include <stdio.h>
#include <unistd.h>
#define BUFSIZE1 512
#define BUFSIZE2 ((BUFSIZE1/2) - 8)

int main(int argc, char **argv) {
char *buf1R1;
char *buf2R1;
char *buf1R2;
buf1R1 = (char *) malloc(BUFSIZE2);
buf2R1 = (char *) malloc(BUFSIZE2);
free(buf1R1);
free(buf2R1);
buf1R2 = (char *) malloc(BUFSIZE1);
strncpy(buf1R2, argv[1], BUFSIZE1-1);
free(buf2R1);
free(buf1R2);
}
```

Observed Examples

Reference	Description
CVE-2004-0642	Double free resultant from certain error conditions.
CVE-2004-0772	Double free resultant from certain error conditions.
CVE-2005-1689	Double free resultant from certain error conditions.
CVE-2003-0545	Double free from invalid ASN.1 encoding.
CVE-2003-1048	Double free from malformed GIF.
CVE-2005-0891	Double free from malformed GIF.
CVE-2002-0059	Double free from malformed compressed data.

Potential Mitigations

Phase: Architecture and Design

Choose a language that provides automatic memory management.

Phase: Implementation

Ensure that each allocation is freed only once. After freeing a chunk, set the pointer to NULL to ensure the pointer cannot be freed again. In complicated error conditions, be sure that clean-up routines respect the state of allocation properly. If the language is object oriented, ensure that object destructors delete each chunk of memory only once.

Phase: Implementation

Use a static analysis tool to find double free instances.

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	Weakness Base	666	Operation on Resource in Wrong Phase of	**Research Concepts (primary)1000**

			Lifetime	
ChildOf	Weakness Class	675	Duplicate Operations on Resource	Research Concepts1000
ChildOf	Category	742	CERT C Secure Coding Section 08 - Memory Management (MEM)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
PeerOf	Weakness Base	123	Write-what-where Condition	Research Concepts1000
PeerOf	Weakness Base	416	Use After Free	Development Concepts699 Research Concepts1000
MemberOf	View	630	Weaknesses Examined by SAMATE	**Weaknesses Examined by SAMATE (primary)630**
PeerOf	Weakness Base	364	Signal Handler Race Condition	Research Concepts1000

Relationship Notes

This is usually resultant from another weakness, such as an unhandled error or race condition between threads. It could also be primary to weaknesses such as buffer overflows.

Affected Resources

- Memory

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
PLOVER			DFREE - Double-Free Vulnerability
7 Pernicious Kingdoms			Double Free
CLASP			Doubly freeing memory
CERT C Secure Coding	MEM00-C		Allocate and free memory in the same module, at the same level of abstraction
CERT C Secure Coding	MEM01-C		Store a new value in pointers immediately after free()
CERT C Secure Coding	MEM31-C		Free dynamically allocated memory exactly once

White Box Definitions

A weakness where code path has:

1. start statement that relinquishes a dynamically allocated memory resource
2. end statement that relinquishes the dynamically allocated memory resource

Maintenance Notes

It could be argued that Double Free would be most appropriately located as a child of "Use after Free", but "Use" and "Release" are considered to be distinct operations within vulnerability theory, therefore this is more accurately "Release of a Resource after Expiration or Release", which doesn't exist yet.

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 Potential Mitigations, 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, Description, Maintenance Notes, Relationships, Other Notes, Relationship Notes, Taxonomy Mappings		
2008-11-24	CWE Content Team	MITRE	Internal

	updated Relationships, Taxonomy Mappings		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Other Notes		

BACK TO TOP

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

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

Description

Description Summary

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

Extended Description

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

Terminology Notes

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

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C

C++

Modes of Introduction

Memory leaks have two common and sometimes overlapping causes:

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

Common Consequences

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

Likelihood of Exploit

Medium

Demonstrative Examples

Example 1

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

(Bad Code)

Example Language: **C**

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

return NULL;
}
```

```
return buf;
}
```

Example 2

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

(Bad Code)

Example Language: **C**

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

free(foo);
}
int main() {

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

Observed Examples

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

Potential Mitigations

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

Phase: Architecture and Design

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

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

Relationships

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

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

Relationship Notes

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

Affected Resources

- Memory

Functional Areas

- Memory management

Taxonomy Mappings

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

White Box Definitions

A weakness where the code path has:

1. start statement that allocates dynamically allocated memory resource

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

Where "loses" is defined through the following scenarios:

1. identity of the dynamic allocated memory resource never obtained

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

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

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

References

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

Content History

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

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

Previous Entry Names

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

BACK TO TOP

Use of Zero Initialized Pointer

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

Source Code Examples

CPP

Explicit NULL Dereference

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

Implicit NULL Dereference

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

Java

Explicit Null Dereference

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

Use of Insufficiently Random Values

Risk

What might happen

Random values are often used as a mechanism to prevent malicious users from guessing a value, such as a password, encryption key, or session identifier. Depending on what this random value is used for, an attacker would be able to predict the next numbers generated, or previously generated values. This could enable the attacker to hijack another user's session, impersonate another user, or crack an encryption key (depending on what the pseudo-random value was used for).

Cause

How does it happen

The application uses a weak method of generating pseudo-random values, such that other numbers could be determined from a relatively small sample size. Since the pseudo-random number generator used is designed for statistically uniform distribution of values, it is approximately deterministic. Thus, after collecting a few generated values (e.g. by creating a few individual sessions, and collecting the sessionids), it would be possible for an attacker to calculate another sessionid.
Specifically, if this pseudo-random value is used in any security context, such as passwords, keys, or secret identifiers, an attacker would be able to predict the next numbers generated, or previously generated values.

General Recommendations

How to avoid it

Generic Guidance:

- Whenever unpredicatable numbers are required in a security context, use a cryptographically strong random number generator, instead of a statistical pseudo-random generator.
- Use the cryptorandom generator that is built-in to your language or platform, and ensure it is securely seeded. Do not seed the generator with a weak, non-random seed. (In most cases, the default is securely random).
- Ensure you use a long enough random value, to make brute-force attacks unfeasible.

Specific Recommendations:

- Do not use the statistical pseudo-random number generator, use the cryptorandom generator instead. In Java, this is the SecureRandom class.

Source Code Examples

Java

Use of a weak pseudo-random number generator

```
Random random = new Random();

long sessNum = random.nextLong();

String sessionId = sessNum.toString();
```

```
```

Cryptographically secure random number generator

```
SecureRandom random = new SecureRandom();

byte sessBytes[] = new byte[32];

random.nextBytes(sessBytes);

String sessionId = new String(sessBytes);
```

Objc

Use of a weak pseudo-random number generator

```
long sessNum = rand();
NSString* sessionId = [NSString stringWithFormat:@"%ld", sessNum];
```

Cryptographically secure random number generator

```
UInt32 sessBytes;
SecRandomCopyBytes(kSecRandomDefault, sizeof(sessBytes), (uint8_t*)&sessBytes);

NSString* sessionId = [NSString stringWithFormat:@"%llu", sessBytes];
```

Swift

Use of a weak pseudo-random number generator

```
let sessNum = rand();
let sessionId = String(format:"%ld", sessNum)
```

Cryptographically secure random number generator

```
var sessBytes: UInt32 = 0
withUnsafeMutablePointer(&sessBytes, { (sessBytesPointer) -> Void in
    let castedPointer = unsafeBitCast(sessBytesPointer, UnsafeMutablePointer<UInt8>.self)
    SecRandomCopyBytes(kSecRandomDefault, sizeof(UInt32), castedPointer)
})

let sessionId = String(format:"%llu", sessBytes)
```

Unchecked Return Value

Risk

What might happen

A program that does not check function return values could cause the application to enter an undefined state. This could lead to unexpected behavior and unintended consequences, including inconsistent data, system crashes or other error-based exploits.

Cause

How does it happen

The application calls a system function, but does not receive or check the result of this funciton. These functions often return error codes in the result, or share other status codes with it's caller. The application simply ignores this result value, losing this vital information.

General Recommendations

How to avoid it

- Always check the result of any called function that returns a value, and verify the result is an expected value.

- Ensure the calling function responds to all possible return values.

- Expect runtime errors and handle them gracefully. Explicitly define a mechanism for handling unexpected errors.

Source Code Examples

CPP

Unchecked Memory Allocation

```
buff = (char*) malloc(size);
strncpy(buff, source, size);
```

Safer Memory Allocation

```
buff = (char*) malloc(size+1);
if (buff==NULL) exit(1);

strncpy(buff, source, size);
buff[size] = '\0';
```

Use of sizeof() on a Pointer Type

Weakness ID: 467 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The code calls sizeof() on a malloced pointer type, which always returns the wordsize/8. This can produce an unexpected result if the programmer intended to determine how much memory has been allocated.

Time of Introduction

- Implementation

Applicable Platforms

Languages

C

C++

Common Consequences

Scope	Effect
Integrity	This error can often cause one to allocate a buffer that is much smaller than what is needed, leading to resultant weaknesses such as buffer overflows.

Likelihood of Exploit

High

Demonstrative Examples

Example 1

Care should be taken to ensure sizeof returns the size of the data structure itself, and not the size of the pointer to the data structure.

In this example, sizeof(foo) returns the size of the pointer.

(Bad Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(foo));
```

In this example, sizeof(*foo) returns the size of the data structure and not the size of the pointer.

(Good Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(*foo));
```

Example 2

This example defines a fixed username and password. The AuthenticateUser() function is intended to accept a username and a password from an untrusted user, and check to ensure that it matches the username and password. If the username and password match, AuthenticateUser() is intended to indicate that authentication succeeded.

(Bad Code)

```
/* Ignore CWE-259 (hard-coded password) and CWE-309 (use of password system for authentication) for this example. */

char *username = "admin";
char *pass = "password";

int AuthenticateUser(char *inUser, char *inPass) {
```

```
printf("Sizeof username = %d\n", sizeof(username));
printf("Sizeof pass = %d\n", sizeof(pass));

if (strncmp(username, inUser, sizeof(username))) {
printf("Auth failure of username using sizeof\n");
return(AUTH_FAIL);
}
/* Because of CWE-467, the sizeof returns 4 on many platforms and architectures. */
if (! strncmp(pass, inPass, sizeof(pass))) {
printf("Auth success of password using sizeof\n");
return(AUTH_SUCCESS);
}
else {
printf("Auth fail of password using sizeof\n");
return(AUTH_FAIL);
}
}

int main (int argc, char **argv)
{
int authResult;

if (argc < 3) {
ExitError("Usage: Provide a username and password");
}
authResult = AuthenticateUser(argv[1], argv[2]);
if (authResult != AUTH_SUCCESS) {
ExitError("Authentication failed");
}
else {
DoAuthenticatedTask(argv[1]);
}
}
```

In AuthenticateUser(), because sizeof() is applied to a parameter with an array type, the sizeof() call might return 4 on many modern architectures. As a result, the strncmp() call only checks the first four characters of the input password, resulting in a partial comparison (CWE-187), leading to improper authentication (CWE-287).

Because of the partial comparison, any of these passwords would still cause authentication to succeed for the "admin" user:

(Attack)

```
pass5
passABCDEFGH
passWORD
```

Because only 4 characters are checked, this significantly reduces the search space for an attacker, making brute force attacks more feasible.

The same problem also applies to the username, so values such as "adminXYZ" and "administrator" will succeed for the username.

Potential Mitigations

Phase: Implementation

Use expressions such as "sizeof(*pointer)" instead of "sizeof(pointer)", unless you intend to run sizeof() on a pointer type to gain some platform independence or if you are allocating a variable on the stack.

Other Notes

The use of sizeof() on a pointer can sometimes generate useful information. An obvious case is to find out the wordsize on a platform. More often than not, the appearance of sizeof(pointer) indicates a bug.

Weakness Ordinalities

Ordinality	Description
Primary	*(where the weakness exists independent of other weaknesses)*

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	465	Pointer Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	682	Incorrect Calculation	**Research Concepts (primary)1000**
ChildOf	Category	737	CERT C Secure Coding Section 03 - Expressions (EXP)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
CanPrecede	Weakness Base	131	Incorrect Calculation of Buffer Size	Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Use of sizeof() on a pointer type
CERT C Secure Coding	ARR01-C		Do not apply the sizeof operator to a pointer when taking the size of an array
CERT C Secure Coding	EXP01-C		Do not take the size of a pointer to determine the size of the pointed-to type

White Box Definitions

A weakness where code path has:

1. end statement that passes an identity of a dynamically allocated memory resource to a sizeof operator
2. start statement that allocates the dynamically allocated memory resource

References

Robert Seacord. "EXP01-A. Do not take the sizeof a pointer to determine the size of a type". <https://www.securecoding.cert.org/confluence/display/seccode/EXP01-A.+Do+not+take+the+sizeof+a+pointer+to+determine+the+size+of+a+type>.

Content History

Submissions

Submission Date	Submitter	Organization	Source
	CLASP		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-01		KDM Analytics	External
	added/updated white box definitions		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

BACK TO TOP

NULL Pointer Dereference

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

Source Code Examples

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

Heuristic Buffer Overflow malloc

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

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

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