magma Scan Report

Project Name	magma
Scan Start	Friday, June 21, 2024 11:09:56 PM
Preset	Checkmarx Default
Scan Time	00h:02m:05s
Lines Of Code Scanned	4903
Files Scanned	6
Report Creation Time	Friday, June 21, 2024 11:13:44 PM
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057
Team	CxServer
Checkmarx Version	8.7.0
Scan Type	Full
Source Origin	LocalPath
Density	5/100 (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

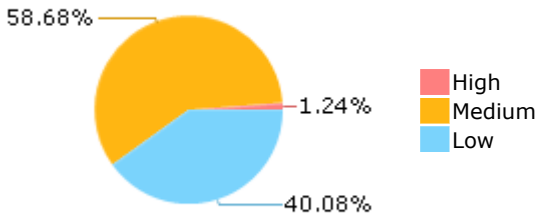
Results Limit

Results limit per query was set to 50

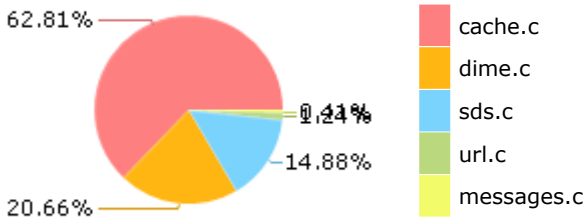
Selected Queries

Selected queries are listed in Result Summary

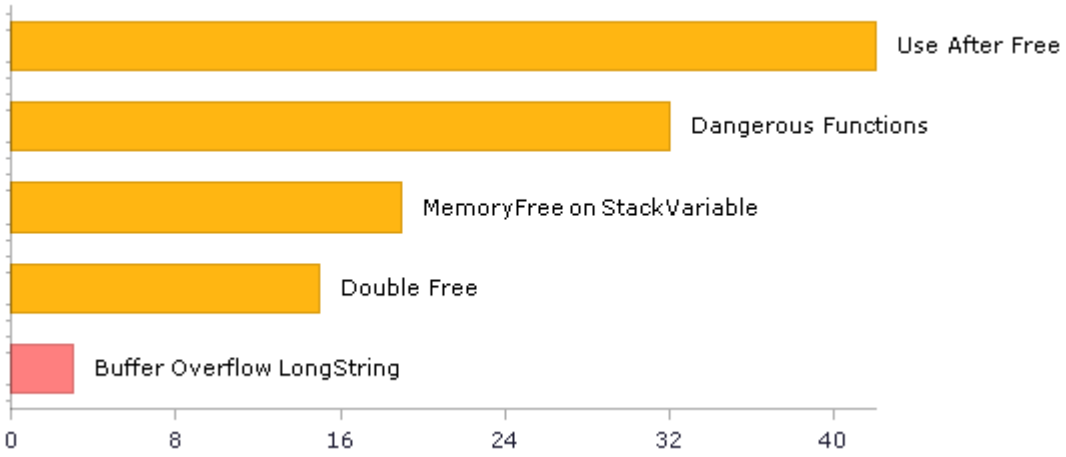
Result Summary



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	64	22
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	70	70
A3-Sensitive Data Exposure	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	0	0
A4-XML External Entities (XXE)	App. Specific	AVERAGE	COMMON	EASY	SEVERE	App. Specific	0	0
A5-Broken Access Control*	App. Specific	AVERAGE	COMMON	AVERAGE	SEVERE	App. Specific	3	1
A6-Security Misconfiguration	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A7-Cross-Site Scripting (XSS)	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A8-Insecure Deserialization	App. Specific	DIFFICULT	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A9-Using Components with Known Vulnerabilities*	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	MODERATE	App. Specific	32	32
A10-Insufficient Logging & Monitoring	App. Specific	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	App. Specific	0	0

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

Scan Summary - OWASP Top 10 2013

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

Category	Threat Agent	Attack Vectors	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	AVERAGE	SEVERE	ALL DATA	0	0
A2-Broken Authentication and Session Management	EXTERNAL, INTERNAL USERS	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	AFFECTED DATA AND FUNCTIONS	0	0
A3-Cross-Site Scripting (XSS)	EXTERNAL, INTERNAL, ADMIN USERS	AVERAGE	VERY WIDESPREAD	EASY	MODERATE	AFFECTED DATA AND SYSTEM	0	0
A4-Insecure Direct Object References	SYSTEM USERS	EASY	COMMON	EASY	MODERATE	EXPOSED DATA	3	1
A5-Security Misconfiguration	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	EASY	MODERATE	ALL DATA AND SYSTEM	0	0
A6-Sensitive Data Exposure	EXTERNAL, INTERNAL, ADMIN USERS, USERS BROWSERS	DIFFICULT	UNCOMMON	AVERAGE	SEVERE	EXPOSED DATA	0	0
A7-Missing Function Level Access Control*	EXTERNAL, INTERNAL USERS	EASY	COMMON	AVERAGE	MODERATE	EXPOSED DATA AND FUNCTIONS	0	0
A8-Cross-Site Request Forgery (CSRF)	USERS BROWSERS	AVERAGE	COMMON	EASY	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0
A9-Using Components with Known Vulnerabilities*	EXTERNAL USERS, AUTOMATED TOOLS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	32	32
A10-Unvalidated Redirects and Forwards	USERS BROWSERS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0

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

Scan Summary - PCI DSS v3.2

Category	Issues Found	Best Fix Locations
PCI DSS (3.2) - 6.5.1 - Injection flaws - particularly SQL injection	0	0
PCI DSS (3.2) - 6.5.2 - Buffer overflows	25	23
PCI DSS (3.2) - 6.5.3 - Insecure cryptographic storage	0	0
PCI DSS (3.2) - 6.5.4 - Insecure communications	0	0
PCI DSS (3.2) - 6.5.5 - Improper error handling*	0	0
PCI DSS (3.2) - 6.5.7 - Cross-site scripting (XSS)	0	0
PCI DSS (3.2) - 6.5.8 - Improper access control	0	0
PCI DSS (3.2) - 6.5.9 - Cross-site request forgery	0	0
PCI DSS (3.2) - 6.5.10 - Broken authentication and session management	0	0

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

Scan Summary - FISMA 2014

Category	Description	Issues Found	Best Fix Locations
Access Control	Organizations must limit information system access to authorized users, processes acting on behalf of authorized users, or devices (including other information systems) and to the types of transactions and functions that authorized users are permitted to exercise.	1	1
Audit And Accountability*	Organizations must: (i) create, protect, and retain information system audit records to the extent needed to enable the monitoring, analysis, investigation, and reporting of unlawful, unauthorized, or inappropriate information system activity; and (ii) ensure that the actions of individual information system users can be uniquely traced to those users so they can be held accountable for their actions.	0	0
Configuration Management	Organizations must: (i) establish and maintain baseline configurations and inventories of organizational information systems (including hardware, software, firmware, and documentation) throughout the respective system development life cycles; and (ii) establish and enforce security configuration settings for information technology products employed in organizational information systems.	5	5
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.	69	69
Media Protection	Organizations must: (i) protect information system media, both paper and digital; (ii) limit access to information on information system media to authorized users; and (iii) sanitize or destroy information system media before disposal or release for reuse.	0	0
System And Communications Protection	Organizations must: (i) monitor, control, and protect organizational communications (i.e., information transmitted or received by organizational information systems) at the external boundaries and key internal boundaries of the information systems; and (ii) employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational information systems.	0	0
System And Information Integrity	Organizations must: (i) identify, report, and correct information and information system flaws in a timely manner; (ii) provide protection from malicious code at appropriate locations within organizational information systems; and (iii) monitor information system security alerts and advisories and take appropriate actions in response.	8	8

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

Scan Summary - NIST SP 800-53

Category	Issues Found	Best Fix Locations
AC-12 Session Termination (P2)	0	0
AC-3 Access Enforcement (P1)	75	75
AC-4 Information Flow Enforcement (P1)	0	0
AC-6 Least Privilege (P1)	0	0
AU-9 Protection of Audit Information (P1)	0	0
CM-6 Configuration Settings (P2)	0	0
IA-5 Authenticator Management (P1)	0	0
IA-6 Authenticator Feedback (P2)	0	0
IA-8 Identification and Authentication (Non-Organizational Users) (P1)	0	0
SC-12 Cryptographic Key Establishment and Management (P1)	0	0
SC-13 Cryptographic Protection (P1)	0	0
SC-17 Public Key Infrastructure Certificates (P1)	0	0
SC-18 Mobile Code (P2)	0	0
SC-23 Session Authenticity (P1)*	0	0
SC-28 Protection of Information at Rest (P1)	0	0
SC-4 Information in Shared Resources (P1)	0	0
SC-5 Denial of Service Protection (P1)*	52	13
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*	19	14
SI-11 Error Handling (P2)*	9	9
SI-15 Information Output Filtering (P0)	0	0
SI-16 Memory Protection (P1)	15	6

***** 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

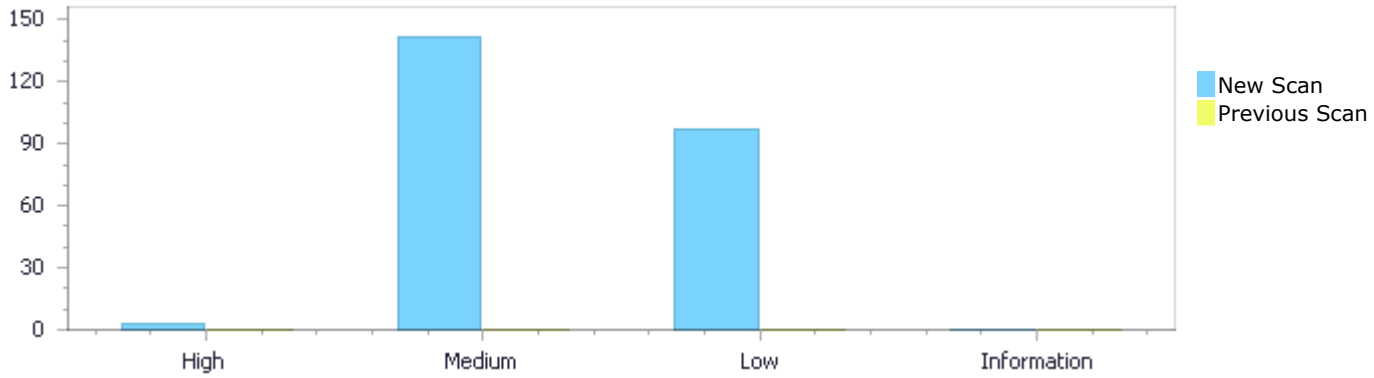
Scan Summary - Custom

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	3	142	97	0	242
Recurrent Issues	0	0	0	0	0
Total	3	142	97	0	242
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	3	142	97	0	242
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	3	142	97	0	242

Result Summary

Vulnerability Type	Occurrences	Severity
Buffer Overflow LongString	3	High
Use After Free	42	Medium
Dangerous Functions	32	Medium
MemoryFree on StackVariable	19	Medium
Double Free	15	Medium

Buffer Overflow boundcpy WrongSizeParam	13	Medium
Integer Overflow	8	Medium
Use of Zero Initialized Pointer	7	Medium
Stored Buffer Overflow boundcpy	4	Medium
Memory Leak	2	Medium
Improper Resource Access Authorization	69	Low
Unchecked Return Value	9	Low
Exposure of System Data to Unauthorized Control Sphere	5	Low
Potential Path Traversal	3	Low
Unchecked Array Index	3	Low
TOCTOU	2	Low
Use of Sizeof On a Pointer Type	2	Low
Heuristic 2nd Order Buffer Overflow read	1	Low
Inconsistent Implementations	1	Low
Incorrect Permission Assignment For Critical Resources	1	Low
NULL Pointer Dereference	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
magma/cache.c	104
magma/sds.c	31
magma/dime.c	6
magma/url.c	3
magma/messages.c	1

Scan Results Details

Buffer Overflow LongString

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

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
NIST SP 800-53: SI-10 Information Input Validation (P1)
OWASP Top 10 2017: A1-Injection

Description

Buffer Overflow LongString\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=1
Status	New

The size of the buffer used by url_encode in Address, at line 65 of magma/url.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that url_encode passes to "%%%02X", at line 65 of magma/url.c, to overwrite the target buffer.

	Source	Destination
File	magma/url.c	magma/url.c
Line	109	109
Object	"%%%02X"	Address

Code Snippet
File Name magma/url.c
Method stringer_t * url_encode(stringer_t *s) {

```
....
109.              else if (snprintf(hex, 4, "%%%02X", *p) == 3 && (r =
st_append(output, PLACER(&hex[0], 3)))) {
```

Buffer Overflow LongString\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=2
Status	New

The size of the buffer used by url_encode in hex, at line 65 of magma/url.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that url_encode passes to "%%%02X", at line 65 of magma/url.c, to overwrite the target buffer.

	Source	Destination
File	magma/url.c	magma/url.c

Line	109	109
Object	"%%%02X"	hex

Code Snippet
File Name magma/url.c
Method stringer_t * url_encode(stringer_t *s) {

```
....
109.              else if (snprintf(hex, 4, "%%%02X", *p) == 3 && (r =
st_append(output, PLACER(&hex[0], 3)))) {
```

Buffer Overflow LongString\Path 3:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=3
Status	New

The size of the buffer used by url_encode in hex, at line 65 of magma/url.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that url_encode passes to "%%%02X", at line 65 of magma/url.c, to overwrite the target buffer.

	Source	Destination
File	magma/url.c	magma/url.c
Line	109	109
Object	"%%%02X"	hex

Code Snippet
File Name magma/url.c
Method stringer_t * url_encode(stringer_t *s) {

```
....
109.              else if (snprintf(hex, 4, "%%%02X", *p) == 3 && (r =
st_append(output, PLACER(&hex[0], 3)))) {
```

Use After Free

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

Categories

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

Description

Use After Free\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=113

Status	New

The pointer cdir at magma/cache.c in line 61 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	121	137
Object	cdir	cdir

Code Snippet
File Name magma/cache.c
Method char *_get_dime_dir_location(const char *suffix) {

```
....
121.                    free(cdir);
....
137.        _dime_dir = cdir;
```

Use After Free\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=114
Status	New

The pointer cdir at magma/cache.c in line 61 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	127	137
Object	cdir	cdir

Code Snippet
File Name magma/cache.c
Method char *_get_dime_dir_location(const char *suffix) {

```
....
127.                free(cdir);
....
137.        _dime_dir = cdir;
```

Use After Free\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=115
Status	New

The pointer cdir at magma/cache.c in line 61 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	133	137
Object	cdir	cdir

Code Snippet
File Name magma/cache.c
Method char *_get_dime_dir_location(const char *suffix) {

```
....
133.          free(cdir);
....
137.      _dime_dir = cdir;
```

Use After Free\Path 4:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=116
Status New

The pointer dataholder at magma/cache.c in line 1585 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1611	1608
Object	dataholder	dataholder

Code Snippet
File Name magma/cache.c
Method size_t _mem_append_serialized_array(unsigned char **buf, size_t *blen, const unsigned char **array, size_t itemsize) {

```
....
1611.                  free(dataholder);
....
1608.          if (!_mem_append(&dataholder, &holdlen, *arptr,
itemsize)) {
```

Use After Free\Path 5:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=117
Status New

The pointer strholder at magma/cache.c in line 1640 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1667	1664
Object	strholder	strholder

Code Snippet
File Name magma/cache.c
Method size_t _mem_append_serialized_str_array(unsigned char **buf, size_t *blen, const char **array) {

```
....
1667.                   free(strholder);
....
1664.           if (!_mem_append(&strholder, &holdlen, (unsigned char
*)*arptr, strlen(*arptr) + 1)) {
```

Use After Free\Path 6:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=118
Status New

The pointer dataholder at magma/cache.c in line 1698 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1724	1721
Object	dataholder	dataholder

Code Snippet
File Name magma/cache.c
Method size_t _mem_append_serialized_array_cb(unsigned char **buf, size_t *blen, const char **array, custom_serializer_t sfn) {

```
....
1724.                   free(dataholder);
....
1721.           if (!sfn(&dataholder, &holdlen, *arptr)) {
```

Use After Free\Path 7:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=119
Status New

The pointer next at magma/cache.c in line 741 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	779
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method cached_object_t *_add_cached_object_cmp(const char *id, const void *key, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed, cached_store_comparator_t cmpfn) {

```
....
779.           ptr = ptr->next;
```

Use After Free\Path 8:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=120
Status New

The pointer data at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2182
Object	object	data

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2182.               store->destructor(object->data);
```

Use After Free\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=121
Status	New

The pointer dtype at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2150
Object	object	dtype

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2150.       if (!(store = _get_cached_store_by_type(object->dtype))) {
```

Use After Free\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=122
Status	New

The pointer next at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2165
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2165.           next = object->next;
```

Use After Free\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=123
Status	New

The pointer next at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2163
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2163.       if (object->next) {
```

Use After Free\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=124
Status	New

The pointer next at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2160
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2160.           object->prev->next = object->next;
```

Use After Free\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=125
Status	New

The pointer next at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2160
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2160.           object->prev->next = object->next;
```

Use After Free\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=126
Status	New

The pointer prev at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2164
Object	object	prev

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2164.           object->next->prev = object->prev;
```

Use After Free\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=127
Status	New

The pointer prev at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2164
Object	object	prev

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2164.           object->next->prev = object->prev;
```

Use After Free\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=128
Status	New

The pointer next at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2156
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2156.           store->head = object->next;
```

Use After Free\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=129
Status	New

The pointer head at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2156
Object	object	head

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2156.           store->head = object->next;
```

Use After Free\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=130
Status	New

The pointer prev at magma/cache.c in line 2140 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2159
Object	object	prev

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
....
2159.       if (object->prev) {
```

Use After Free\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=131
Status	New

The pointer timestamp at magma/cache.c in line 2093 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2117
Object	object	timestamp

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method int _is_object_expired(cached_object_t *obj, int *refresh) {

```
....
2117.       if (obj->ttl && (time_t)(obj->timestamp + obj->ttl) <= now) {
```

Use After Free\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=132
Status	New

The pointer timestamp at magma/cache.c in line 2093 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2098

Object	object	timestamp

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c
Method int _is_object_expired(cached_object_t *obj, int *refresh) {

```
....
2098.       if (!obj || !obj->timestamp) {
```

Use After Free\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=133
Status	New

The pointer ttl at magma/cache.c in line 2093 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2117
Object	object	ttl

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c
Method int _is_object_expired(cached_object_t *obj, int *refresh) {

```
....
2117.       if (obj->ttl && (time_t)(obj->timestamp + obj->ttl) <= now) {
```

Use After Free\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=134
Status	New

The pointer ttl at magma/cache.c in line 2093 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2117
Object	object	ttl

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method int _is_object_expired(cached_object_t *obj, int *refresh) {

```
....
2117.       if (obj->ttl && (time_t)(obj->timestamp + obj->ttl) <= now) {
```

Use After Free\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=135
Status	New

The pointer expiration at magma/cache.c in line 2093 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2119
Object	object	expiration

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.          free(object);
```

File Name magma/cache.c

Method int _is_object_expired(cached_object_t *obj, int *refresh) {

```
....
2119.          if (!obj->relaxed || !obj->expiration) {
```

Use After Free\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=136
Status	New

The pointer expiration at magma/cache.c in line 2093 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2111
Object	object	expiration

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.          free(object);
```

File Name magma/cache.c

Method int _is_object_expired(cached_object_t *obj, int *refresh) {

```
....
2111.      if (obj->expiration && obj->expiration < now) {
```

Use After Free\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=137
Status	New

The pointer expiration at magma/cache.c in line 2093 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2111
Object	object	expiration

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c
Method int _is_object_expired(cached_object_t *obj, int *refresh) {

```
....
2111.       if (obj->expiration && obj->expiration < now) {
```

Use After Free\Path 26:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=138
Status New

The pointer relaxed at magma/cache.c in line 2093 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	2119
Object	object	relaxed

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method	int _is_object_expired(cached_object_t *obj, int *refresh) {

```
....
2119.           if (!obj->relaxed || !obj->expiration) {
```

Use After Free\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=139
Status	New

The pointer data at magma/cache.c in line 741 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	774
Object	object	data

Code Snippet	
File Name	magma/cache.c
Method	cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale) {

```
....
2192.           free(object);
```

File Name	magma/cache.c
Method	cached_object_t *_add_cached_object_cmp(const char *id, const void *key, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed, cached_store_comparator_t cmpfn) {

```
....
774.          } else if (!cmpfn(ptr->data, key)) {
```

Use After Free\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=140
Status	New

The pointer id at magma/cache.c in line 741 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c

Line	2192	771
Object	object	id

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method cached_object_t *_add_cached_object_cmp(const char *id, const void *key, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed, cached_store_comparator_t cmpfn) {

```
....
771.          if (!memcmp(ptr->id, hashid, SHA_256_SIZE)) {
```

Use After Free\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=141
Status	New

The pointer next at magma/cache.c in line 903 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	937
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method int _remove_cached_object(const char *oid, cached_store_t *store) {

```
....
937.          ptr = ptr->next;
```

Use After Free\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=142
Status	New

The pointer id at magma/cache.c in line 903 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	931
Object	object	id

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.          free(object);
```

File Name magma/cache.c

Method int _remove_cached_object(const char *oid, cached_store_t *store) {

```
....
931.          if (!memcmp(ptr->id, hashid, SHA_256_SIZE)) {
```

Use After Free\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=143
Status	New

The pointer next at magma/cache.c in line 953 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	982
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method int _remove_cached_object_cmp(const void *key, cached_store_t *store, cached_store_comparator_t cmpfn) {

```
....
982.          ptr = ptr->next;
```

Use After Free\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=144
Status	New

The pointer data at magma/cache.c in line 953 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	976
Object	object	data

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method int _remove_cached_object_cmp(const void *key, cached_store_t *store, cached_store_comparator_t cmpfn) {

```
....
976.          if (!cmpfn(ptr->data, key)) {
```

Use After Free\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=145
Status	New

The pointer next at magma/cache.c in line 506 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	534
Object	object	next

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method int _cached_object_exists_cmp(const void *key, cached_store_t *store, cached_store_comparator_t cmpfn) {

```
....
534.           ptr = ptr->next;
```

Use After Free\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=146
Status	New

The pointer data at magma/cache.c in line 506 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	529
Object	object	data

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.          free(object);
```

File Name magma/cache.c

Method int _cached_object_exists_cmp(const void *key, cached_store_t *store, cached_store_comparator_t cmpfn) {

```
....
529.          if (!cmpfn(ptr->data, key)) {
```

Use After Free\Path 35:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=147
Status	New

The pointer next at magma/cache.c in line 355 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	394
Object	object	next

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale) {

```
....
2192.          free(object);
```

File Name magma/cache.c

Method cached_object_t *_find_cached_object(const char *oid, cached_store_t *store) {

```
....
394.          ptr = ptr->next;
```

Use After Free\Path 36:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=148
Status	New

The pointer id at magma/cache.c in line 355 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	383
Object	object	id

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c
Method cached_object_t *_find_cached_object(const char *oid, cached_store_t *store) {

```
....
383.          if (!memcmp(ptr->id, hashid, SHA_256_SIZE)) {
```

Use After Free\Path 37:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=149
Status	New

The pointer next at magma/cache.c in line 412 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	445
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method	cached_object_t *_find_cached_object_cmp(const void *key, cached_store_t *store, cached_store_comparator_t cmpfn) {

```
....
445.          ptr = ptr->next;
```

Use After Free\Path 38:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=150
Status	New

The pointer data at magma/cache.c in line 412 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	434
Object	object	data

Code Snippet

File Name	magma/cache.c
Method	cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale) {

```
....
2192.          free(object);
```

File Name	magma/cache.c
Method	cached_object_t *_find_cached_object_cmp(const void *key, cached_store_t *store, cached_store_comparator_t cmpfn) {

```
....
434.          if (!cmpfn(ptr->data, key)) {
```

Use After Free\Path 39:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=151
Status	New

The pointer next at magma/cache.c in line 460 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c

Line	2192	488
Object	object	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c
Method int _cached_object_exists(const unsigned char *hashid, cached_store_t *store) {

```
....
488.          ptr = ptr->next;
```

Use After Free\Path 40:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=152
Status	New

The pointer id at magma/cache.c in line 460 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	483
Object	object	id

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c
Method int _cached_object_exists(const unsigned char *hashid, cached_store_t *store) {

```
....
483.          if (!memcmp(ptr->id, hashid, SHA_256_SIZE)) {
```

Use After Free\Path 41:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=153
Status	New

The pointer next at magma/cache.c in line 556 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	590
Object	object	next

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2192.           free(object);
```

File Name magma/cache.c

Method cached_object_t *_add_cached_object(const char *id, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed) {

```
....
590.          ptr = ptr->next;
```

Use After Free\Path 42:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=154
Status	New

The pointer id at magma/cache.c in line 556 is being used after it has been freed.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2192	585
Object	object	id

Code Snippet

File Name magma/cache.c

Method	cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale) {

```
....
2192.           free(object);
```

File Name	magma/cache.c
Method	cached_object_t *_add_cached_object(const char *id, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed) {

```
....
585.          if (!memcmp(ptr->id, hashid, SHA_256_SIZE)) {
```

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=1050067&projectid=50057&pathid=64
Status	New

The dangerous function, memcpy, was found in use at line 556 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	610	610
Object	memcpy	memcpy

Code Snippet

File Name	magma/cache.c
Method	cached_object_t *_add_cached_object(const char *id, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed) {

```
....
610.      memcpy(entry->id, hashid, SHA_256_SIZE);
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=65
Status	New

The dangerous function, memcpy, was found in use at line 741 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	788	788
Object	memcpy	memcpy

Code Snippet
File Name magma/cache.c
Method cached_object_t *_add_cached_object_cmp(const char *id, const void *key, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed, cached_store_comparator_t cmpfn) {

```
....
788.      memcpy(entry->id, hashid, SHA_256_SIZE);
```

Dangerous Functions\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=66
Status	New

The dangerous function, memcpy, was found in use at line 994 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1025	1025
Object	memcpy	memcpy

Code Snippet
File Name magma/cache.c
Method cached_object_t *_clone_cached_object(const cached_object_t *obj) {

```
....
1025.      memcpy(result->id, obj->id, sizeof(result->id));
```

Dangerous Functions\Path 4:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=67
Status	New

The dangerous function, memcpy, was found in use at line 1196 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1322	1322
Object	memcpy	memcpy

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1322.          memcpy(obj, cdata, chdr_size);
```

Dangerous Functions\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=68
Status	New

The dangerous function, memcpy, was found in use at line 1755 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1768	1768
Object	memcpy	memcpy

Code Snippet
File Name magma/cache.c
Method unsigned int _deserialize_data(unsigned char *dst, unsigned char **bufptr, const unsigned char *bufend, size_t len) {

```
....
1768.      memcpy(dst, *bufptr, len);
```

Dangerous Functions\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=69

Status	New

The dangerous function, memcpy, was found in use at line 1826 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1858	1858
Object	memcpy	memcpy

Code Snippet
File Name magma/cache.c
Method unsigned int _deserialize_string(char **dst, unsigned char **bufptr, const unsigned char *bufend) {

```
....
1858.       memcpy(*dst, *bufptr, rsize);
```

Dangerous Functions\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=70
Status	New

The dangerous function, memcpy, was found in use at line 2208 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2242	2242
Object	memcpy	memcpy

Code Snippet
File Name magma/cache.c
Method cached_object_t *_replace_object(cached_object_t *oobj, cached_object_t *nobj, int shadow) {

```
....
2242.       memcpy(nobj->id, oobj->id, SHA_256_SIZE);
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=71
Status	New

The dangerous function, memcpy, was found in use at line 2343 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2364	2364
Object	memcpy	memcpy

Code Snippet
File Name magma/cache.c
Method void *_serialize_signet_cb(void *record, size_t *outlen) {

```
....
2364.       memcpy(result, serial, ssize);
```

Dangerous Functions\Path 9:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=72
Status New

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

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

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

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

Dangerous Functions\Path 10:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=73
Status New

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

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

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

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

Dangerous Functions\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=74
Status	New

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

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

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

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

Dangerous Functions\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=75
Status	New

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

	Source	Destination
File	magma/sds.c	magma/sds.c

Line	381	381
Object	memcpy	memcpy

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

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

Dangerous Functions\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=76
Status	New

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

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

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

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

Dangerous Functions\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=77
Status	New

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

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

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

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

Dangerous Functions\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=78
Status	New

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

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

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

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

Dangerous Functions\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=79
Status	New

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

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

Code Snippet	
File Name	magma/sds.c

Method	sds sdscatfmt(sds s, char const *fmt, ...) {

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

Dangerous Functions\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=80
Status	New

The dangerous function, strlen, was found in use at line 355 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	369	369
Object	strlen	strlen

Code Snippet	
File Name	magma/cache.c
Method	cached_object_t *_find_cached_object(const char *oid, cached_store_t *store) {

```
....
369.      if (_compute_sha_hash(256, (unsigned char *)oid, strlen(oid),
hashid) < 0) {
```

Dangerous Functions\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=81
Status	New

The dangerous function, strlen, was found in use at line 556 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	570	570
Object	strlen	strlen

Code Snippet	
File Name	magma/cache.c
Method	cached_object_t *_add_cached_object(const char *id, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed) {

```
....
570.      if (_compute_sha_hash(256, (unsigned char *)id, strlen(id),
hashid) < 0) {
```

Dangerous Functions\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=82
Status	New

The dangerous function, strlen, was found in use at line 741 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	755	755
Object	strlen	strlen

Code Snippet

File Name magma/cache.c

Method cached_object_t *_add_cached_object_cmp(const char *id, const void *key, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed, cached_store_comparator_t cmpfn) {

```
....
755.      if (_compute_sha_hash(256, (unsigned char *)id, strlen(id),
hashid) < 0) {
```

Dangerous Functions\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=83
Status	New

The dangerous function, strlen, was found in use at line 903 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	916	916
Object	strlen	strlen

Code Snippet

File Name magma/cache.c

Method	int _remove_cached_object(const char *oid, cached_store_t *store) {

```
....
916.      if (_compute_sha_hash(256, (unsigned char *)oid, strlen(oid),
hashid) < 0) {
```

Dangerous Functions\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=84
Status	New

The dangerous function, strlen, was found in use at line 1552 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1564	1564
Object	strlen	strlen

Code Snippet	
File Name	magma/cache.c
Method	size_t _mem_append_serialized_string(unsigned char **buf, size_t *blen, const char *string) {

```
....
1564.          res = _mem_append(buf, blen, (unsigned char *)string,
strlen(string) + 1);
```

Dangerous Functions\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=85
Status	New

The dangerous function, strlen, was found in use at line 1640 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1664	1664
Object	strlen	strlen

Code Snippet	
File Name	magma/cache.c

Method	size_t _mem_append_serialized_str_array(unsigned char **buf, size_t *blen, const char **array) {

```
....
1664.          if (!_mem_append(&strholder, &holdlen, (unsigned char
*)*arptr, strlen(*arptr) + 1)) {
```

Dangerous Functions\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=86
Status	New

The dangerous function, strlen, was found in use at line 63 in magma/dime.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	107	107
Object	strlen	strlen

Code Snippet	
File Name	magma/dime.c
Method	static void show_coc(const char *cocstr) {

```
....
107.              dptr = token + strlen(token) - 1;
```

Dangerous Functions\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=87
Status	New

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

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

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

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

Dangerous Functions\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=88
Status	New

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

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

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

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

Dangerous Functions\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=89
Status	New

The dangerous function, strlen, was found in use at line 391 in magma/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	392	392
Object	strlen	strlen

Code Snippet
File Name magma/sds.c
Method sds sdscat(sds s, const char *t) {

```
....
392.      return sdscatlen(s, t, strlen(t));
```

Dangerous Functions\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=90
Status	New

The dangerous function, strlen, was found in use at line 418 in magma/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	419	419
Object	strlen	strlen

Code Snippet
File Name magma/sds.c
Method sds sdscpy(sds s, const char *t) {

```
....
419.      return sdscpylen(s, t, strlen(t));
```

Dangerous Functions\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=91
Status	New

The dangerous function, strlen, was found in use at line 501 in magma/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	504	504
Object	strlen	strlen

Code Snippet
File Name magma/sds.c
Method sds sdscatvprintf(sds s, const char *fmt, va_list ap) {

```
....
504.      size_t buflen = strlen(fmt)*2;
```

Dangerous Functions\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=92
Status	New

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

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	607	607
Object	strlen	strlen

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

```
....
607.                 l = (next == 's') ? strlen(str) : sdslen(str);
```

Dangerous Functions\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=93
Status	New

The dangerous function, vsnprintf, was found in use at line 501 in magma/sds.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	520	520
Object	vsnprintf	vsnprintf

Code Snippet
File Name magma/sds.c
Method sds sdscatvprintf(sds s, const char *fmt, va_list ap) {

```
....
520.          vsnprintf(buf, buflen, fmt, cpy);
```

Dangerous Functions\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=94
Status	New

The dangerous function, realloc, was found in use at line 1196 in magma/cache.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1268	1268
Object	realloc	realloc

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1268.              if (!(reall_res = realloc(cdata, clen))) {
```

Dangerous Functions\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=95
Status	New

The dangerous function, atoi, was found in use at line 138 in magma/dime.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	184	184
Object	atoi	atoi

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
184.                    if (!(port = atoi(optarg))) {
```

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=1050067&projectid=50057&pathid=32
Status	New

Calling free() (line 61) on a variable that was not dynamically allocated (line 61) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	121	121
Object	cdir	cdir

Code Snippet
File Name magma/cache.c
Method char *_get_dime_dir_location(const char *suffix) {

```
....
121.                    free(cdir);
```

MemoryFree on StackVariable\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=33
Status	New

Calling free() (line 61) on a variable that was not dynamically allocated (line 61) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	127	127
Object	cdir	cdir

Code Snippet
File Name magma/cache.c

Method	char *_get_dime_dir_location(const char *suffix) {

```
....
127.              free(cdir);
```

MemoryFree on StackVariable\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=34
Status	New

Calling free() (line 61) on a variable that was not dynamically allocated (line 61) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	133	133
Object	cdir	cdir

Code Snippet	
File Name	magma/cache.c
Method	char *_get_dime_dir_location(const char *suffix) {

```
....
133.          free(cdir);
```

MemoryFree on StackVariable\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=35
Status	New

Calling free() (line 154) on a variable that was not dynamically allocated (line 154) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	183	183
Object	cfile	cfile

Code Snippet	
File Name	magma/cache.c
Method	char *_get_cache_location(void) {

```
....
183.      free(cfile);
```

MemoryFree on StackVariable\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=36
Status	New

Calling free() (line 1068) on a variable that was not dynamically allocated (line 1068) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1136	1136
Object	tstr	tstr

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1136.              free(tstr);
```

MemoryFree on StackVariable\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=37
Status	New

Calling free() (line 1196) on a variable that was not dynamically allocated (line 1196) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1374	1374
Object	tstr	tstr

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1374.               free(tstr);
```

MemoryFree on StackVariable\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=38
Status	New

Calling free() (line 1414) on a variable that was not dynamically allocated (line 1414) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1476	1476
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _save_cache_contents(void) {

```
....
1476.                           free(cdata);
```

MemoryFree on StackVariable\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=39
Status	New

Calling free() (line 1414) on a variable that was not dynamically allocated (line 1414) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1481	1481
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _save_cache_contents(void) {

```
....
1481.                    free(cdata);
```

MemoryFree on StackVariable\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=40
Status	New

Calling free() (line 1585) on a variable that was not dynamically allocated (line 1585) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1611	1611
Object	dataholder	dataholder

Code Snippet
File Name magma/cache.c
Method size_t _mem_append_serialized_array(unsigned char **buf, size_t *blen, const unsigned char **array, size_t itemsize) {

```
....
1611.                free(dataholder);
```

MemoryFree on StackVariable\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=41
Status	New

Calling free() (line 1585) on a variable that was not dynamically allocated (line 1585) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1621	1621
Object	dataholder	dataholder

Code Snippet
File Name magma/cache.c
Method size_t _mem_append_serialized_array(unsigned char **buf, size_t *blen, const unsigned char **array, size_t itemsize) {

```
....
1621.       free(dataholder);
```

MemoryFree on StackVariable\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=42
Status	New

Calling free() (line 1640) on a variable that was not dynamically allocated (line 1640) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1667	1667
Object	strholder	strholder

Code Snippet

File Name magma/cache.c

Method size_t _mem_append_serialized_str_array(unsigned char **buf, size_t *blen, const char **array) {

```
....
1667.                   free(strholder);
```

MemoryFree on StackVariable\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=43
Status	New

Calling free() (line 1640) on a variable that was not dynamically allocated (line 1640) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1677	1677
Object	strholder	strholder

Code Snippet

File Name magma/cache.c

Method size_t _mem_append_serialized_str_array(unsigned char **buf, size_t *blen, const char **array) {

```
....
1677.       free(strholder);
```

MemoryFree on StackVariable\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=44
Status	New

Calling free() (line 1698) on a variable that was not dynamically allocated (line 1698) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1724	1724
Object	dataholder	dataholder

Code Snippet
File Name magma/cache.c
Method size_t _mem_append_serialized_array_cb(unsigned char **buf, size_t *blen, const char **array, custom_serializer_t sfn) {

```
....
1724.                   free(dataholder);
```

MemoryFree on StackVariable\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=45
Status	New

Calling free() (line 1698) on a variable that was not dynamically allocated (line 1698) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1734	1734
Object	dataholder	dataholder

Code Snippet
File Name magma/cache.c
Method size_t _mem_append_serialized_array_cb(unsigned char **buf, size_t *blen, const char **array, custom_serializer_t sfn) {

```
....
1734.       free(dataholder);
```

MemoryFree on StackVariable\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=46
Status	New

Calling free() (line 2343) on a variable that was not dynamically allocated (line 2343) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2359	2359
Object	serial	serial

Code Snippet
File Name magma/cache.c
Method void *_serialize_signet_cb(void *record, size_t *outlen) {

```
....
2359.           free(serial);
```

MemoryFree on StackVariable\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=47
Status	New

Calling free() (line 2343) on a variable that was not dynamically allocated (line 2343) in file magma/cache.c may result with a crash.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2365	2365
Object	serial	serial

Code Snippet
File Name magma/cache.c
Method void *_serialize_signet_cb(void *record, size_t *outlen) {

```
....
2365.       free(serial);
```

MemoryFree on StackVariable\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=48
Status	New

Calling free() (line 138) on a variable that was not dynamically allocated (line 138) in file magma/dime.c may result with a crash.

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	435	435
Object	line	line

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
435.                            free(line);
```

MemoryFree on StackVariable\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=49
Status	New

Calling free() (line 138) on a variable that was not dynamically allocated (line 138) in file magma/dime.c may result with a crash.

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	441	441
Object	line	line

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
441.                    free(line);
```

MemoryFree on StackVariable\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=50
Status	New

Calling free() (line 138) on a variable that was not dynamically allocated (line 138) in file magma/dime.c may result with a crash.

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	470	470
Object	line	line

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
470.        free(line);
```

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=1050067&projectid=50057&pathid=96
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	103	108
Object	pwbuf	pwbuf

Code Snippet

File Name	magma/cache.c
Method	char *_get_dime_dir_location(const char *suffix) {

```
....
103.            free(pwbuf);
....
108.            free(pwbuf);
```

Double Free\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=97
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	100	108
Object	pwbuf	pwbuf

Code Snippet

File Name	magma/cache.c
Method	char *_get_dime_dir_location(const char *suffix) {

```
....
100.            free(pwbuf);
....
108.            free(pwbuf);
```

Double Free\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=98
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	108	112
Object	pwbuf	pwbuf

Code Snippet

File Name	magma/cache.c
Method	char *_get_dime_dir_location(const char *suffix) {

```
....
108.            free(pwbuf);
....
112.        free(pwbuf);
```

Double Free\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=99
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	103	112
Object	pwbuf	pwbuf

Code Snippet
File Name magma/cache.c
Method char *_get_dime_dir_location(const char *suffix) {

```
....
103.            free(pwbuf);
....
112.        free(pwbuf);
```

Double Free\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=100
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	100	112
Object	pwbuf	pwbuf

Code Snippet
File Name magma/cache.c
Method char *_get_dime_dir_location(const char *suffix) {

```
....
100.            free(pwbuf);
....
112.        free(pwbuf);
```

Double Free\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=101
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1226	1231
Object	cfile	cfile

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1226.               free(cfile);
....
1231.           free(cfile);
```

Double Free\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=102
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1286	1243
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1286.               free(cdata);
....
1243.                   free(cdata);
```

Double Free\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=500

	57&pathid=103
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1316	1243
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1316.               free(cdata);
....
1243.                    free(cdata);
```

Double Free\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=104
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1316	1272
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1316.               free(cdata);
....
1272.                         free(cdata);
```

Double Free\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=105
Status	New

	Source	Destination

File	magma/cache.c	magma/cache.c
Line	1286	1272
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1286.               free(cdata);
....
1272.                       free(cdata);
```

Double Free\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=106
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1286	1402
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1286.               free(cdata);
....
1402.          free(cdata);
```

Double Free\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=107
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1316	1402
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1316.               free(cdata);
....
1402.           free(cdata);
```

Double Free\Path 13:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=108
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1476	1481
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _save_cache_contents(void) {

```
....
1476.                           free(cdata);
....
1481.                       free(cdata);
```

Double Free\Path 14:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=109
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2359	2365
Object	serial	serial

Code Snippet
File Name magma/cache.c
Method void *_serialize_signet_cb(void *record, size_t *outlen) {

```
....
2359.           free(serial);
....
2365.       free(serial);
```

Double Free\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=110
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	435	441
Object	line	line

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
435.                            free(line);
....
441.                        free(line);
```

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=1050067&projectid=50057&pathid=19
Status	New

The size of the buffer used by *_clone_cached_object in ->, at line 994 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *_clone_cached_object passes to ->, at line 994 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c

Line	1025	1025
Object	->	->

Code Snippet
File Name magma/cache.c
Method cached_object_t *_clone_cached_object(const cached_object_t *obj) {

```
....
1025.       memcpy(result->id, obj->id, sizeof(result->id));
```

Buffer Overflow boundcpy WrongSizeParam\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=20
Status	New

The size of the buffer used by *_create_cached_object in cached_object_t, at line 244 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *_create_cached_object passes to cached_object_t, at line 244 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	260	260
Object	cached_object_t	cached_object_t

Code Snippet
File Name magma/cache.c
Method cached_object_t *_create_cached_object(cached_data_type_t dtype, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed) {

```
....
260.       memset(result, 0, sizeof(cached_object_t));
```

Buffer Overflow boundcpy WrongSizeParam\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=21
Status	New

The size of the buffer used by _destroy_cache_entry in cached_object_t, at line 316 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _destroy_cache_entry passes to cached_object_t, at line 316 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c

Line	343	343
Object	cached_object_t	cached_object_t

Code Snippet
File Name magma/cache.c
Method void _destroy_cache_entry(cached_object_t *entry) {

```
....
343.      memset(entry, 0, sizeof(cached_object_t));
```

Buffer Overflow boundcpy WrongSizeParam\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=22
Status	New

The size of the buffer used by *_add_cached_object in cached_object_t, at line 556 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *_add_cached_object passes to cached_object_t, at line 556 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	601	601
Object	cached_object_t	cached_object_t

Code Snippet
File Name magma/cache.c
Method cached_object_t *_add_cached_object(const char *id, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed) {

```
....
601.      memset(entry, 0, sizeof(cached_object_t));
```

Buffer Overflow boundcpy WrongSizeParam\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=23
Status	New

The size of the buffer used by *_clone_cached_object in cached_object_t, at line 994 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *_clone_cached_object passes to cached_object_t, at line 994 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c

Line	1021	1021
Object	cached_object_t	cached_object_t

Code Snippet
File Name magma/cache.c
Method cached_object_t *_clone_cached_object(const cached_object_t *obj) {

```
....
1021.       memset(result, 0, sizeof(cached_object_t));
```

Buffer Overflow boundcpy WrongSizeParam\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=24
Status	New

The size of the buffer used by _load_cache_contents in cached_object_t, at line 1196 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _load_cache_contents passes to cached_object_t, at line 1196 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1321	1321
Object	cached_object_t	cached_object_t

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1321.           memset(obj, 0, sizeof(cached_object_t));
```

Buffer Overflow boundcpy WrongSizeParam\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=25
Status	New

The size of the buffer used by _deserialize_string in rsize, at line 1826 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _deserialize_string passes to rsize, at line 1826 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1858	1858

Object	rsize	rsize

Code Snippet
File Name magma/cache.c
Method unsigned int _deserialize_string(char **dst, unsigned char **bufptr, const unsigned char *bufend) {

```
....
1858.       memcpy(*dst, *bufptr, rsize);
```

Buffer Overflow boundcpy WrongSizeParam\Path 8:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=26
Status New

The size of the buffer used by sdscatlen in len, at line 376 of magma/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 magma/sds.c, to overwrite the target buffer.

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

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 9:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=27
Status New

The size of the buffer used by sdscpylen in len, at line 405 of magma/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 magma/sds.c, to overwrite the target buffer.

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

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=28
Status	New

The size of the buffer used by sdscatfmt in l, at line 579 of magma/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 magma/sds.c, to overwrite the target buffer.

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

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=29
Status	New

The size of the buffer used by sdscatfmt in l, at line 579 of magma/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 magma/sds.c, to overwrite the target buffer.

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

Code Snippet	
File Name	magma/sds.c

Method	sds sdscatfmt(sds s, char const *fmt, ...) {

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

Buffer Overflow boundcpy WrongSizeParam\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=30
Status	New

The size of the buffer used by sdscatfmt in l, at line 579 of magma/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 magma/sds.c, to overwrite the target buffer.

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

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=31
Status	New

The size of the buffer used by sdscmp in minlen, at line 767 of magma/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 magma/sds.c, to overwrite the target buffer.

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

Code Snippet	
File Name	magma/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=1050067&projectid=50057&pathid=53
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 1503 of magma/cache.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1509	1509
Object	AssignExpr	AssignExpr

Code Snippet
File Name magma/cache.c
Method int _set_cache_permissions(unsigned long flags) {

```
....
1509.       _cache_flags = flags;
```

Integer Overflow\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=54
Status	New

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

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	587	587

Object	AssignExpr	AssignExpr

Code Snippet
File Name magma/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 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=55
Status	New

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

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

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

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

Integer Overflow\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=56
Status	New

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

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

Code Snippet

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

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

Integer Overflow\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=57
Status	New

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

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

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

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

Integer Overflow\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=58
Status	New

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

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

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

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

Integer Overflow\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=59
Status	New

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

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

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

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

Integer Overflow\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=60
Status	New

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

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

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

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

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=1050067&projectid=50057&pathid=155
Status	New

The variable declared in next at magma/cache.c in line 2140 is not initialized when it is used by shadow at magma/cache.c in line 2208.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2143	2247
Object	next	shadow

Code Snippet

File Name magma/cache.c

Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2143.       cached_object_t *next = NULL;
```

File Name magma/cache.c

Method cached_object_t *_replace_object(cached_object_t *oobj, cached_object_t *nobj, int shadow) {

```
....
2247.           nobj->shadow = oobj;
```

Use of Zero Initialized Pointer\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=156
Status	New

The variable declared in next at magma/cache.c in line 2140 is not initialized when it is used by next at magma/cache.c in line 2208.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2143	2231
Object	next	next

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2143.       cached_object_t *next = NULL;
```

File Name magma/cache.c

Method cached_object_t *_replace_object(cached_object_t *oobj, cached_object_t *nobj, int shadow) {

```
....
2231.       nobj->next = oobj->next;
```

Use of Zero Initialized Pointer\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=157
Status	New

The variable declared in data at magma/messages.c in line 73 is not initialized when it is used by data at magma/messages.c in line 73.

	Source	Destination
File	magma/messages.c	magma/messages.c
Line	81	99
Object	data	data

Code Snippet
File Name magma/messages.c
Method stringer_t * naked_message_get(stringer_t *message, prime_org_signet_t *org, prime_user_key_t *user) {

```
....
81.    uchr_t *data = NULL, *position = NULL;
....
99.    data += 6;
```

Use of Zero Initialized Pointer\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=158
Status	New

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

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

Code Snippet
File Name magma/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=1050067&projectid=50057&pathid=159
Status	New

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

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

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

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

Use of Zero Initialized Pointer\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=160
Status	New

The variable declared in reall_res at magma/cache.c in line 1196 is not initialized when it is used by cdata at magma/cache.c in line 1196.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1280	1279
Object	reall_res	cdata

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1280.               reall_res = NULL;
....
1279.               cdata = reall_res;
```

Use of Zero Initialized Pointer\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=161
Status	New

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

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

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

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

Stored Buffer Overflow boundcpy

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

Categories

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

Description

Stored Buffer Overflow boundcpy\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=162
Status	New

The size of the buffer used by _load_cache_contents in objlen, at line 1196 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _load_cache_contents passes to Address, at line 1196 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1239	1283
Object	Address	objlen

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1239.           if ((nread = read(cfd, &objlen, sizeof(objlen))) !=
sizeof(objlen)) {
....
1283.           memset(cdata, 0, objlen);
```

Stored Buffer Overflow boundcpy\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=163
Status	New

The size of the buffer used by _load_cache_contents in objlen, at line 1196 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _load_cache_contents passes to cdata, at line 1196 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1285	1283
Object	cdata	objlen

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1285.           if (read(cfd, cdata, objlen) != objlen) {
....
1283.           memset(cdata, 0, objlen);
```

Stored Buffer Overflow boundcpy\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=164
Status	New

The size of the buffer used by _load_cache_contents in chdr_size, at line 1196 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _load_cache_contents passes to Address, at line 1196 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1239	1322
Object	Address	chdr_size

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1239.           if ((nread = read(cfd, &objlen, sizeof(objlen))) !=
sizeof(objlen)) {
....
1322.           memcpy(obj, cdata, chdr_size);
```

Stored Buffer Overflow boundcpy\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=165
Status	New

The size of the buffer used by _load_cache_contents in chdr_size, at line 1196 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _load_cache_contents passes to cdata, at line 1196 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c

Line	1285	1322
Object	cdata	chdr_size

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1285.          if (read(cfd, cdata, objlen) != objlen) {
....
1322.          memcpy(obj, cdata, chdr_size);
```

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=1050067&projectid=50057&pathid=111
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2357	2357
Object	result	result

Code Snippet
File Name magma/cache.c
Method void *_serialize_signet_cb(void *record, size_t *outlen) {

```
....
2357.      if(!(result = malloc(ssize))) {
```

Memory Leak\Path 2:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=112
Status New

	Source	Destination

File	magma/cache.c	magma/cache.c
Line	1853	1853
Object	dst	dst

Code Snippet
File Name magma/cache.c
Method unsigned int _deserialize_string(char **dst, unsigned char **bufptr, const unsigned char *bufend) {

```
....
1853.       if (!(*dst = malloc(rsize))) {
```

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=1050067&projectid=50057&pathid=166
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1239	1239
Object	Address	Address

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1239.           if ((nread = read(cfd, &objlen, sizeof(objlen))) !=
sizeof(objlen)) {
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=167
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1285	1285
Object	cdata	cdata

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1285.           if (read(cfd, cdata, objlen) != objlen) {
```

Improper Resource Access Authorization\Path 3:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=168
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	325	325
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _destroy_cache_entry(cached_object_t *entry) {

```
....
325.           fprintf(stderr, "Error destroying cached object: could not
find associated cached store.\n");
```

Improper Resource Access Authorization\Path 4:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=169
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	332	332
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _destroy_cache_entry(cached_object_t *entry) {

```
....
332.                    fprintf(stderr, "Cache destructor generated
error(s):\n");
```

Improper Resource Access Authorization\Path 5:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=170
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	683	683
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method cached_object_t *_add_cached_object_forced(const char *id, cached_store_t *store, unsigned long ttl, time_t expiration, void *data, int persists, int relaxed) {

```
....
683.                fprintf(stderr, "\n");
```

Improper Resource Access Authorization\Path 6:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=171
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1081	1081
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1081.          fprintf(stderr, "Dumping data cached store of type: %s
...\n", cached_stores[i].description);
```

Improper Resource Access Authorization\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=172
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1086	1086
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1086.              fprintf(stderr, "- Skipped empty store.\n");
```

Improper Resource Access Authorization\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=173
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1105	1105
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1105.                  fprintf(stderr, "Error: Could not get current
time for TTL calculation.\n");
```

Improper Resource Access Authorization\Path 9:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=174
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1119	1119
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1119.               fprintf(stderr, "+++ [%u: ", (unsigned int)(j + 1));
```

Improper Resource Access Authorization\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=175
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1123	1123
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1123.               fprintf(stderr, "] ttl = %s, expiration = %s, data =
%s, timestamp = %s, persist = %s",
```

Improper Resource Access Authorization\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=176
Status	New

	Source	Destination

File	magma/cache.c	magma/cache.c
Line	1127	1127
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1127.                   fprintf(stderr, " [RELAXED]");
```

Improper Resource Access Authorization\Path 12:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=177
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1131	1131
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1131.                   fprintf(stderr, " [SHADOWED]\n");
```

Improper Resource Access Authorization\Path 13:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=178
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1133	1133
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1133.                   fprintf(stderr, "\n");
```

Improper Resource Access Authorization\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=179
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1148	1148
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1148.               fprintf(stderr, "    Skipped a total of %zu non-
persistent cache entries.\n", (total - j));
```

Improper Resource Access Authorization\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=180
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1172	1172
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache_data(FILE *fp, const cached_object_t *obj, int brief) {

```
....
1172.           fprintf(stderr, "Error dumping object cache data: could
not find associated cached store.\n");
```

Improper Resource Access Authorization\Path 16:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=181
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1184	1184
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache_data(FILE *fp, const cached_object_t *obj, int brief) {

```
....
1184.           fprintf(stderr, "Errors were encountered while dumping
cache store:\n");
```

Improper Resource Access Authorization\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=182
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1295	1295
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1295.               fprintf(stderr, "Error: read cached data of
unrecognized type. Continuing...\n");
```

Improper Resource Access Authorization\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=183
Status	New

	Source	Destination

File	magma/cache.c	magma/cache.c
Line	1298	1298
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1298.               fprintf(stderr, "Error: cached object did not have a
deserialization handler. Continuing...\n");
```

Improper Resource Access Authorization\Path 19:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=184
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1309	1309
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1309.               fprintf(stderr, "Error reading in cached object data;
unexpected small entry size.\n");
```

Improper Resource Access Authorization\Path 20:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=185
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1333	1333
Object	fprintf	fprintf

Code Snippet

File Name	magma/cache.c
Method	int _load_cache_contents(void) {

```
....
1333.                   fprintf(stderr, "Error: cached object could not
be deserialized (continuing)...\n");
```

Improper Resource Access Authorization\Path 21:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=186
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1349	1349
Object	fprintf	fprintf

Code Snippet

File Name	magma/cache.c
Method	int _load_cache_contents(void) {

```
....
1349.               fprintf(stderr, "Error: Could not get current time
for TTL calculation.\n");
```

Improper Resource Access Authorization\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=187
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1363	1363
Object	fprintf	fprintf

Code Snippet

File Name	magma/cache.c
Method	int _load_cache_contents(void) {

```
....
1363.               fprintf(stderr, "+++ type = %u, ttl = %s, expiration
= %s, data = %s, timestamp = %s\n",
```

Improper Resource Access Authorization\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=188
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1379	1379
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1379.               fprintf(stderr, "Error: deserialized cached object
was a duplicate:\n");
```

Improper Resource Access Authorization\Path 24:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=189
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1384	1384
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1384.               fprintf(stderr, "Error: could not look up object in
cache.\n");
```

Improper Resource Access Authorization\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=190

Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1465	1465
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method int _save_cache_contents(void) {

```
....
1465.                       fprintf(stderr, "Error serializing cached
data for storage:\n");
```

Improper Resource Access Authorization\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=191
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2185	2185
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method cached_object_t *_unlink_object(cached_object_t *object, int destroy, int stale)
{

```
....
2185.                   fprintf(stderr, "Cache destructor generated
error(s):\n");
```

Improper Resource Access Authorization\Path 27:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=192
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c

Line	2299	2299
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method unsigned int _evict_if_stale(cached_object_t **objptr) {

```
....
2299.              fprintf(stderr, "Error: could not unlink expired item
from cache:");
```

Improper Resource Access Authorization\Path 28:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=193
Status New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2386	2386
Object	fprintf	fprintf

Code Snippet
File Name magma/cache.c
Method void _dump_signet_cb(FILE *fp, void *record, int brief) {

```
....
2386.          fprintf(fp, "*** hashed ***");
```

Improper Resource Access Authorization\Path 29:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=194
Status New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	185	185
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
185.                            fprintf(stderr, "Error: specified invalid
port number.\n");
```

Improper Resource Access Authorization\Path 30:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=195
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	190	190
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
190.                            fprintf(stderr, "Error: invalid port
specified; must be %u or %u.\n", DMTP_PORT, DMTP_PORT_DUAL);
```

Improper Resource Access Authorization\Path 31:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=196
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	207	207
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
207.                fprintf(stderr, "Error: unable to bind the program to
the required dynamic symbols.\n");
```

Improper Resource Access Authorization\Path 32:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=197
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	227	227
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
227.                fprintf(stderr, "Error: no signet name was
specified!\n");
```

Improper Resource Access Authorization\Path 33:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=198
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	232	232
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
232.                fprintf(stderr, "Error: -h or -c cannot be specified
together. Please choose one.\n");
```

Improper Resource Access Authorization\Path 34:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=199
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	237	237
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
237.              fprintf(stderr, "Error: the -h option requires a
signet fingerprint to be supplied with -f.\n");
```

Improper Resource Access Authorization\Path 35:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=200
Status New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	240	240
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
240.              fprintf(stderr, "Error: the -c option requires a
signet fingerprint to be supplied with -f.\n");
```

Improper Resource Access Authorization\Path 36:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=201
Status New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	245	245
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
245.              fprintf(stderr, "Error: the -e option can only be
specified together with -h.\n");
```

Improper Resource Access Authorization\Path 37:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=202
Status New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	250	250
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
250.              fprintf(stderr, "Error: the -0 option can only be used
in conjunction with -i.\n");
```

Improper Resource Access Authorization\Path 38:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=203
Status New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	265	265
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
265.              fprintf(stderr, "Error: chain of custody query is not
allowed for org signets.\n");
```

Improper Resource Access Authorization\Path 39:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=204
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	273	273
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
273.              fprintf(stderr, "Error: unable to load cache contents
from disk.\n");
```

Improper Resource Access Authorization\Path 40:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=205
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	298	298
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
298.                    fprintf(stderr, "Failed to retrieve DIME
management record from file.\n");
```

Improper Resource Access Authorization\Path 41:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=206
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	309	309
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
309.                    fprintf(stderr, "Failed to query DIME management
record.\n");
```

Improper Resource Access Authorization\Path 42:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=207
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	314	314
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
314.                            fprintf(stderr, "Error: unable to save
cache contents to disk.\n");
```

Improper Resource Access Authorization\Path 43:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=208
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	324	324
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
324.              fprintf(stderr, "Error: unable to save cache contents
to disk.\n");
```

Improper Resource Access Authorization\Path 44:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=209
Status New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	346	346
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
346.              fprintf(stderr, "Connecting to DX at %s:%d ...\n",
dxname, port);
```

Improper Resource Access Authorization\Path 45:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=210
Status New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	356	356
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
356.                fprintf(stderr, "Establishing connection to DX
server...\n");
```

Improper Resource Access Authorization\Path 46:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=211
Status New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	361	361
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
361.                fprintf(stderr, "Error: could not connect to DX
server.\n");
```

Improper Resource Access Authorization\Path 47:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=212
Status New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	369	369
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
369.              fprintf(stderr, "Error: an error was encountered
during the DX certificate verification process.\n");
```

Improper Resource Access Authorization\Path 48:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=213
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	373	373
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
373.              fprintf(stderr, "Error: DX certificate verification
failed.\n");
```

Improper Resource Access Authorization\Path 49:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=214
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	395	395
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
395.                    fprintf(stderr, "Error: signet verification
failed.\n");
```

Improper Resource Access Authorization\Path 50:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=215
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	414	414
Object	fprintf	fprintf

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
414.                    fprintf(stderr, "Signet history command
failed.\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=1050067&projectid=50057&pathid=8
Status	New

The _dump_cache method calls the snprintf function, at line 1068 of magma/cache.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	magma/cache.c	magma/cache.c
Line	1106	1106
Object	snprintf	snprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1106.                   snprintf(ttlstr, sizeof(ttlstr), "error [original
= %lu]", (unsigned long)ptr->ttl);
```

Unchecked Return Value\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=9
Status	New

The _dump_cache method calls the snprintf function, at line 1068 of magma/cache.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	magma/cache.c	magma/cache.c
Line	1110	1110
Object	snprintf	snprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1110.                       snprintf(ttlstr, sizeof(ttlstr), "[no ttl]");
```

Unchecked Return Value\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=10
Status	New

The _dump_cache method calls the snprintf function, at line 1068 of magma/cache.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	magma/cache.c	magma/cache.c
Line	1112	1112
Object	snprintf	snprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1112.                       snprintf(ttlstr, sizeof(ttlstr), "%lu
seconds", (unsigned long)(ptr->timestamp + ptr->ttl - now));
```

Unchecked Return Value\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=11
Status	New

The _dump_cache method calls the snprintf function, at line 1068 of magma/cache.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	magma/cache.c	magma/cache.c
Line	1114	1114
Object	snprintf	snprintf

Code Snippet
File Name magma/cache.c
Method void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1114.                       snprintf(ttlstr, sizeof(ttlstr), "expired %lu
seconds ago", (unsigned long)(now - (ptr->timestamp + ptr->ttl)));
```

Unchecked Return Value\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=12
Status	New

The _load_cache_contents method calls the snprintf function, at line 1196 of magma/cache.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	magma/cache.c	magma/cache.c
Line	1350	1350
Object	snprintf	snprintf

Code Snippet
File Name magma/cache.c

Method	int _load_cache_contents(void) {

```
....
1350.                snprintf(ttlstr, sizeof(ttlstr), "error [original =
%lu]", (unsigned long)obj->ttl);
```

Unchecked Return Value\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=13
Status	New

The _load_cache_contents method calls the snprintf function, at line 1196 of magma/cache.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	magma/cache.c	magma/cache.c
Line	1354	1354
Object	snprintf	snprintf

Code Snippet	
File Name	magma/cache.c
Method	int _load_cache_contents(void) {

```
....
1354.                    snprintf(ttlstr, sizeof(ttlstr), "%lu seconds",
(unsigned long)(obj->timestamp + obj->ttl - now));
```

Unchecked Return Value\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=14
Status	New

The _load_cache_contents method calls the snprintf function, at line 1196 of magma/cache.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	magma/cache.c	magma/cache.c
Line	1356	1356
Object	snprintf	snprintf

Code Snippet

File Name	magma/cache.c
Method	int _load_cache_contents(void) {

```
....
1356.                   snprintf(ttlstr, sizeof(ttlstr), "expired %lu
seconds ago", (unsigned long)(now - (obj->timestamp + obj->ttl)));
```

Unchecked Return Value\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=15
Status	New

The _dump_cache method calls the expstr function, at line 1068 of magma/cache.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	magma/cache.c	magma/cache.c
Line	1122	1122
Object	expstr	expstr

Code Snippet	
File Name	magma/cache.c
Method	void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1122.              expstr = ptr->expiration ? _get_chr_date(ptr-
>expiration, 1) : strdup("[none]");
```

Unchecked Return Value\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=16
Status	New

The _deserialize_string method calls the Pointer function, at line 1826 of magma/cache.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	magma/cache.c	magma/cache.c
Line	1853	1853
Object	Pointer	Pointer

Code Snippet	
File Name	magma/cache.c
Method	unsigned int _deserialize_string(char **dst, unsigned char **bufptr, const unsigned char *bufend) {

```
....
1853.       if (!(*dst = malloc(rsize))) {
```

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=1050067&projectid=50057&pathid=236
Status	New

The system data read by _dump_cache in the file magma/cache.c at line 1068 is potentially exposed by _dump_cache found in magma/cache.c at line 1068.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1104	1104
Object	perror	perror

Code Snippet	
File Name	magma/cache.c
Method	void _dump_cache(cached_data_type_t dtype, int do_data, int ephemeral) {

```
....
1104.                   perror("time");
```

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=1050067&projectid=50057&pathid=237
Status	New

The system data read by _load_cache_contents in the file magma/cache.c at line 1196 is potentially exposed by _load_cache_contents found in magma/cache.c at line 1196.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1348	1348
Object	perror	perror

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1348.               perror("time");
```

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=1050067&projectid=50057&pathid=238
Status	New

The system data read by _lock_cache_store in the file magma/cache.c at line 2261 is potentially exposed by _lock_cache_store found in magma/cache.c at line 2261.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	2264	2264
Object	perror	perror

Code Snippet
File Name magma/cache.c
Method void _lock_cache_store(cached_store_t *store) {

```
....
2264.           perror("pthread_mutex_lock");
```

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=1050067&projectid=50057&pathid=239
Status	New

The system data read by _unlock_cache_store in the file magma/cache.c at line 2274 is potentially exposed by _unlock_cache_store found in magma/cache.c at line 2274.

	Source	Destination
File	magma/cache.c	magma/cache.c

Line	2277	2277
Object	perror	perror

Code Snippet
File Name magma/cache.c
Method void _unlock_cache_store(cached_store_t *store) {

```
....
2277.          perror("pthread_mutex_unlock");
```

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=1050067&projectid=50057&pathid=240
Status	New

The system data read by show_coc in the file magma/dime.c at line 63 is potentially exposed by show_coc found in magma/dime.c at line 63.

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	75	75
Object	perror	perror

Code Snippet
File Name magma/dime.c
Method static void show_coc(const char *cocstr) {

```
....
75.          perror("strdup");
```

Potential Path Traversal

Query Path:
CPP\Cx\CPP Low Visibility\Potential Path Traversal Version:0

Categories

OWASP Top 10 2013: A4-Insecure Direct Object References
OWASP Top 10 2017: A5-Broken Access Control

Description

Potential Path Traversal\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=5
Status	New

Method *_get_cache_location at line 154 of magma/cache.c gets user input from the getenv element. This element's value then flows through the code and is eventually used in a file path for local disk access in _load_cache_contents at line 1196 of magma/cache.c. This may cause a Path Traversal vulnerability.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	172	1220
Object	getenv	cfile

Code Snippet
File Name magma/cache.c
Method char *_get_cache_location(void) {

```
....
172.      if ((cfile = getenv("DIME_CACHE_FILE"))) {
```

File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1220.      if ((cfd = open(cfile, O_RDONLY)) < 0) {
```

Potential Path Traversal\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=6
Status	New

Method *_get_cache_location at line 154 of magma/cache.c gets user input from the getenv element. This element's value then flows through the code and is eventually used in a file path for local disk access in _load_cache_contents at line 1196 of magma/cache.c. This may cause a Path Traversal vulnerability.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	172	1224
Object	getenv	cfile

Code Snippet
File Name magma/cache.c
Method char *_get_cache_location(void) {

```
....
172.      if ((cfile = getenv("DIME_CACHE_FILE"))) {
```

File Name magma/cache.c

Method	int _load_cache_contents(void) {

```
....
1224.          if ((cfd = creat(cfile, S_IRWXU)) < 0) {
```

Potential Path Traversal\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=7
Status	New

Method *_get_cache_location at line 154 of magma/cache.c gets user input from the getenv element. This element's value then flows through the code and is eventually used in a file path for local disk access in _save_cache_contents at line 1414 of magma/cache.c. This may cause a Path Traversal vulnerability.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	172	1432
Object	getenv	cfile

Code Snippet
File Name magma/cache.c
Method char *_get_cache_location(void) {

```
....
172.      if ((cfile = getenv("DIME_CACHE_FILE"))) {
```

File Name magma/cache.c
Method int _save_cache_contents(void) {

```
....
1432.      if ((cfd = open(cfile, (O_CREAT | O_TRUNC | O_WRONLY),
(S_IRWXU))) < 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=1050067&projectid=50057&pathid=61

Status	New

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	350	350
Object	len	len

Code Snippet
File Name magma/sds.c
Method void sdsIncrLen(sds s, int incr) {

```
....
350.        s[len] = '\0';
```

Unchecked Array Index\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=62
Status	New

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	665	665
Object	i	i

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

```
....
665.        s[i] = '\0';
```

Unchecked Array Index\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=63
Status	New

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	693	693
Object	len	len

Code Snippet
File Name magma/sds.c
Method sds sdstrim(sds s, const char *cset) {

```
....
693.        s[len] = '\0';
```

Use of Sizeof On a Pointer Type

Query Path:
CPP\Cx\CPP Low Visibility\Use of Sizeof On a Pointer Type Version:1

Description

Use of Sizeof On a Pointer Type\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=17
Status	New

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	1023	1023
Object	sizeof	sizeof

Code Snippet
File Name magma/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 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=18
Status	New

	Source	Destination
File	magma/sds.c	magma/sds.c
Line	1029	1029
Object	sizeof	sizeof

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

```
....
1029.              if (vector == NULL) vector = s_malloc(sizeof(void*));
```

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=1050067&projectid=50057&pathid=241
Status	New

The _load_cache_contents method in magma/cache.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	magma/cache.c	magma/cache.c
Line	1220	1220
Object	open	open

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1220.      if ((cfd = open(cfile, O_RDONLY)) < 0) {
```

TOCTOU\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=242
Status	New

The _save_cache_contents method in magma/cache.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	magma/cache.c	magma/cache.c
Line	1432	1432
Object	open	open

Code Snippet
File Name magma/cache.c

Method	int _save_cache_contents(void) {

```
....
1432.       if ((cfd = open(cfile, (O_CREAT | O_TRUNC | O_WRONLY),
(S_IRWXU))) < 0) {
```

Inconsistent Implementations

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

Description

Inconsistent Implementations\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050067&projectid=50057&pathid=4
Status	New

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	149	149
Object	getopt	getopt

Code Snippet	
File Name	magma/dime.c
Method	int main(int argc, char *argv[]) {

```
....
149.        while ((opt = getopt(argc, argv, "046d:e:f:i:hcnp:v")) != -
1) {
```

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=1050067&projectid=50057&pathid=51
Status	New

The variable declared in 0 at magma/dime.c in line 138 is not initialized when it is used by drec at magma/dime.c in line 138.

	Source	Destination
File	magma/dime.c	magma/dime.c
Line	304	304
Object	0	drec

Code Snippet
File Name magma/dime.c
Method int main(int argc, char *argv[]) {

```
....
304.              drec->validated = no_trust ? 0 : 1;
```

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=1050067&projectid=50057&pathid=52
Status	New

The size of the buffer used by _load_cache_contents in objlen, at line 1196 of magma/cache.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that _load_cache_contents passes to Address, at line 1196 of magma/cache.c, to overwrite the target buffer.

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	1239	1285
Object	Address	objlen

Code Snippet
File Name magma/cache.c
Method int _load_cache_contents(void) {

```
....
1239.          if ((nread = read(cfd, &objlen, sizeof(objlen))) !=
sizeof(objlen)) {
....
1285.          if (read(cfd, cdata, objlen) != objlen) {
```

Incorrect Permission Assignment For Critical Resources

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=1050067&projectid=50057&pathid=235
Status	New

	Source	Destination
File	magma/cache.c	magma/cache.c
Line	119	119
Object	mkdir	mkdir

Code Snippet
File Name magma/cache.c
Method char *_get_dime_dir_location(const char *suffix) {

```
....
119.              if (mkdir(cdir, S_IRWXU) < 0) {
```

Buffer Overflow LongString

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Buffer Overflow boundcpy WrongSizeParam

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

MemoryFree on StackVariable

Risk

What might happen

Undefined Behavior may result with a crash. Crashes may give an attacker valuable information about the system and the program internals. Furthermore, it may leave unprotected files (e.g memory) that may be exploited.

Cause

How does it happen

Calling free() on a variable that was not dynamically allocated (e.g. malloc) will result with an Undefined Behavior.

General Recommendations

How to avoid it

Use free() only on dynamically allocated variables in order to prevent unexpected behavior from the compiler.

Source Code Examples

CPP

Bad - Calling free() on a static variable

```
void clean_up(){
  char temp[256];
  do_something();
  free(tmp);
  return;
}
```

Good - Calling free() only on variables that were dynamically allocated

```
void clean_up(){
  char *buff;
  buff = (char*) malloc(1024);
  free(buff);
  return;
}
```

Integer Overflow

Risk

What might happen

Assigning large data types into smaller data types, without proper checks and explicit casting, will lead to undefined behavior and unintentional effects, such as data corruption (e.g. value wraparound, wherein maximum values become minimum values); system crashes; infinite loops; logic errors, such as bypassing of security mechanisms; or even buffer overflows leading to arbitrary code execution.

Cause

How does it happen

This flaw can occur when implicitly casting numerical data types of a larger size, into a variable with a data type of a smaller size. This forces the program to discard some bits of information from the number. Depending on how the numerical data types are stored in memory, this is often the bits with the highest value, causing substantial corruption of the stored number. Alternatively, the sign bit of a signed integer could be lost, completely reversing the intention of the number.

General Recommendations

How to avoid it

- Avoid casting larger data types to smaller types.
- Prefer promoting the target variable to a large enough data type.
- If downcasting is necessary, always check that values are valid and in range of the target type, before casting

---

Source Code Examples

CPP

Unsafe Downsize Casting

```
int unsafe_addition(short op1, int op2) {

    // op2 gets forced from int into a short
    short total = op1 + op2;

    return total;
}
```

Safer Use of Proper Data Types

```
int safe_addition(short op1, int op2) {

    // total variable is of type int, the largest type that is needed
    int total = 0;

    // check if total will overflow available integer size
    if (INT_MAX - abs(op2) > op1)
```

```
    {
        total = op1 + op2;
    }
    else
    {
        // instead of overflow, saturate (but this is not always a good thing)
        total = INT_MAX
    }

    return total;
}
```

Dangerous Functions

Risk

What might happen

Use of dangerous functions may expose varying risks associated with each particular function, with potential impact of improper usage of these functions varying significantly. The presence of such functions indicates a flaw in code maintenance policies and adherence to secure coding practices, in a way that has allowed introducing known dangerous code into the application.

Cause

How does it happen

A dangerous function has been identified within the code. Functions are often deemed dangerous to use for numerous reasons, as there are different sets of vulnerabilities associated with usage of such functions. For example, some string copy and concatenation functions are vulnerable to Buffer Overflow, Memory Disclosure, Denial of Service and more. Use of these functions is not recommended.

General Recommendations

How to avoid it

- Deploy a secure and recommended alternative to any functions that were identified as dangerous.
    - If no secure alternative is found, conduct further researching and testing to identify whether current usage successfully sanitizes and verifies values, and thus successfully avoids the use-cases for whom the function is indeed dangerous
- Conduct a periodical review of methods that are in use, to ensure that all external libraries and built-in functions are up-to-date and whose use has not been excluded from best secure coding practices.

Source Code Examples

CPP

Buffer Overflow in gets()

```
int main()
{
	char buf[10];

	printf("Please enter your name: ");
	gets(buf); // veryveryverylongname
	if (buf == ACCEPTED_NAME)
	{
		// Do something
	}
	return 0;
}
```

Safe reading from user

```
int main()
{
	char buf[10];

	printf("Please enter your name: ");
	fgets(buf, sizeof(buf), stdin); //setting the amount of bytes to read
	if (buf == ACCEPTED_NAME)
	{
		//Do something
	}
	return 0;
}
```

Unsafe function for string copy

```
int main(int argc, char* argv[])
{
	char buf[10];
	strcpy(buf, argv[1]); // overflow occurs when len(argv[1]) > 10 bytes

	return 0;
}
```

Safe string copy

```
int main(int argc, char* argv[])
{
	char buf[10];
	strncpy(buf, argv[1], sizeof(buf));
	buf[9]= '\0'; //strncpy doesn't NULL terminates

	return 0;
}
```

Unsafe format string

```
int main(int argc, char* argv[])
{
	printf(argv[1]); // If argv[1] contains a format token, such as %s,%x or %d, will cause
an access violation
	return 0;
}
```

Safe format string

```
int main(int argc, char* argv[])
{
      printf("%s", argv[1]); // Second parameter is not a formattable string
      return 0;
}
```

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 After Free

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

---

Source Code Examples

CPP

Use of Variable after It was Freed

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

Use of Pointer to Local Variable That Was Freed On Return

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

void func2()

{

	int j;
	j = 5;
```

```
}

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

Use of Zero Initialized Pointer

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

Source Code Examples

CPP

Explicit NULL Dereference

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

Implicit NULL Dereference

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

Java

Explicit Null Dereference

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

Stored Buffer Overflow boundcpy

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

CPP

Overflowing Buffers

```
const int BUFFER_SIZE = 10;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
{
    strcpy(buffer, inputString);
}
```

Checked Buffers

```
const int BUFFER_SIZE = 10;
const int MAX_INPUT_SIZE = 256;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
```

```
{
    if (strnlen(inputString, MAX_INPUT_SIZE) < sizeof(buffer))
    {
        strncpy(buffer, inputString, sizeof(buffer));
    }
}
```

Use of Function with Inconsistent Implementations

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

Description

Description Summary

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

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C: *(Often)*

PHP: *(Often)*

All

Potential Mitigations

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

Other Notes

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

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

Relationships

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

Taxonomy Mappings

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

Content History

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

BACK TO TOP

Potential Path Traversal

Risk

What might happen

An attacker could define any arbitrary file path for the application to use, potentially leading to:

- Stealing sensitive files, such as configuration or system files
- Overwriting files such as program binaries, configuration files, or system files
- Deleting critical files, causing a denial of service (DoS).

Cause

How does it happen

The application uses user input in the file path for accessing files on the application server's local disk. This enables an attacker to arbitrarily determine the file path.

General Recommendations

How to avoid it

1. Ideally, avoid depending on user input for file selection.
2. Validate all input, regardless of source. Validation should be based on a whitelist: accept only data fitting a specified structure, rather than reject bad patterns. Check for:
   - Data type
   - Size
   - Range
   - Format
   - Expected values
3. Accept user input only for the filename, not for the path and folders.
4. Ensure that file path is fully canonicalized.
5. Explicitly limit the application to using a designated folder that separate from the applications binary folder.
6. Restrict the privileges of the application's OS user to necessary files and folders. The application should not be able to write to the application binary folder, and should not read anything outside of the application folder and data folder.

Source Code Examples

CSharp

Using unvalidated user input as the file name may enable the user to access arbitrary files on the server local disk

```
public class PathTraversal
{
        private void foo(TextBox textbox1)
    {
                string fileNum = textbox1.Text;
                string path = "c:\files\file" + fileNum;
                FileStream f = new FileStream(path, FileMode.Open);
                byte[] output = new byte[10];
                f.Read(output,0, 10);
```

```
        }
}
```

Potentially hazardous characters are removed from the user input before use

```
public class PathTraversalFixed
{
        private void foo(TextBox textbox1)
    {
                string fileNum = textbox1.Text.Replace("\", "").Replace("..", "");
         string path = "c:\files\file" + fileNum;
                FileStream f = new FileStream(path, FileMode.Open);
                byte[] output = new byte[10];
                f.Read(output,0, 10);
        }
}
```

Java

Using unvalidated user input as the file name may enable the user to access arbitrary files on the server local disk

```
public class Absolute_Path_Traversal {
    public static void main(String[] args) {
        Scanner userInputScanner = new Scanner(System.in);
        System.out.print("\nEnter file name: ");
        String name = userInputScanner.nextLine();
        String path = "c:\files\file" + name;
        try {
            BufferedReader reader = new BufferedReader(new FileReader(path));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Potentially hazardous characters are removed from the user input before use

```
public class Absolute_Path_Traversal_Fixed {
    public static void main(String[] args) {
        Scanner userInputScanner = new Scanner(System.in);
        System.out.print("\nEnter file name: ");
        String name = userInputScanner.nextLine();
        name = name.replace("/", "").replace("..", "");
        String path = "c:\files\file" + name;
        try {
            BufferedReader reader = new BufferedReader(new FileReader(path));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Unchecked Return Value

Risk

What might happen

A program that does not check function return values could cause the application to enter an undefined state. This could lead to unexpected behavior and unintended consequences, including inconsistent data, system crashes or other error-based exploits.

Cause

How does it happen

The application calls a system function, but does not receive or check the result of this funciton. These functions often return error codes in the result, or share other status codes with it's caller. The application simply ignores this result value, losing this vital information.

General Recommendations

How to avoid it

- Always check the result of any called function that returns a value, and verify the result is an expected value.

- Ensure the calling function responds to all possible return values.

- Expect runtime errors and handle them gracefully. Explicitly define a mechanism for handling unexpected errors.

Source Code Examples

CPP

Unchecked Memory Allocation

```
buff = (char*) malloc(size);
strncpy(buff, source, size);
```

Safer Memory Allocation

```
buff = (char*) malloc(size+1);
if (buff==NULL) exit(1);

strncpy(buff, source, size);
buff[size] = '\0';
```

Use of sizeof() on a Pointer Type

Weakness ID: 467 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The code calls sizeof() on a malloced pointer type, which always returns the wordsize/8. This can produce an unexpected result if the programmer intended to determine how much memory has been allocated.

Time of Introduction

- Implementation

Applicable Platforms

Languages

C

C++

Common Consequences

Scope	Effect
Integrity	This error can often cause one to allocate a buffer that is much smaller than what is needed, leading to resultant weaknesses such as buffer overflows.

Likelihood of Exploit

High

Demonstrative Examples

Example 1

Care should be taken to ensure sizeof returns the size of the data structure itself, and not the size of the pointer to the data structure.

In this example, sizeof(foo) returns the size of the pointer.

(Bad Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(foo));
```

In this example, sizeof(*foo) returns the size of the data structure and not the size of the pointer.

(Good Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(*foo));
```

Example 2

This example defines a fixed username and password. The AuthenticateUser() function is intended to accept a username and a password from an untrusted user, and check to ensure that it matches the username and password. If the username and password match, AuthenticateUser() is intended to indicate that authentication succeeded.

(Bad Code)

```
/* Ignore CWE-259 (hard-coded password) and CWE-309 (use of password system for authentication) for this example. */

char *username = "admin";
char *pass = "password";

int AuthenticateUser(char *inUser, char *inPass) {
```

```
printf("Sizeof username = %d\n", sizeof(username));
printf("Sizeof pass = %d\n", sizeof(pass));

if (strncmp(username, inUser, sizeof(username))) {
printf("Auth failure of username using sizeof\n");
return(AUTH_FAIL);
}
/* Because of CWE-467, the sizeof returns 4 on many platforms and architectures. */
if (! strncmp(pass, inPass, sizeof(pass))) {
printf("Auth success of password using sizeof\n");
return(AUTH_SUCCESS);
}
else {
printf("Auth fail of password using sizeof\n");
return(AUTH_FAIL);
}
}

int main (int argc, char **argv)
{
int authResult;

if (argc < 3) {
ExitError("Usage: Provide a username and password");
}
authResult = AuthenticateUser(argv[1], argv[2]);
if (authResult != AUTH_SUCCESS) {
ExitError("Authentication failed");
}
else {
DoAuthenticatedTask(argv[1]);
}
}
```

In AuthenticateUser(), because sizeof() is applied to a parameter with an array type, the sizeof() call might return 4 on many modern architectures. As a result, the strncmp() call only checks the first four characters of the input password, resulting in a partial comparison (CWE-187), leading to improper authentication (CWE-287).

Because of the partial comparison, any of these passwords would still cause authentication to succeed for the "admin" user:

(Attack)

```
pass5
passABCDEFGH
passWORD
```

Because only 4 characters are checked, this significantly reduces the search space for an attacker, making brute force attacks more feasible.

The same problem also applies to the username, so values such as "adminXYZ" and "administrator" will succeed for the username.

Potential Mitigations

Phase: Implementation

Use expressions such as "sizeof(*pointer)" instead of "sizeof(pointer)", unless you intend to run sizeof() on a pointer type to gain some platform independence or if you are allocating a variable on the stack.

Other Notes

The use of sizeof() on a pointer can sometimes generate useful information. An obvious case is to find out the wordsize on a platform. More often than not, the appearance of sizeof(pointer) indicates a bug.

Weakness Ordinalities

Ordinality	Description
Primary	*(where the weakness exists independent of other weaknesses)*

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	465	Pointer Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	682	Incorrect Calculation	**Research Concepts (primary)1000**
ChildOf	Category	737	CERT C Secure Coding Section 03 - Expressions (EXP)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
CanPrecede	Weakness Base	131	Incorrect Calculation of Buffer Size	Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Use of sizeof() on a pointer type
CERT C Secure Coding	ARR01-C		Do not apply the sizeof operator to a pointer when taking the size of an array
CERT C Secure Coding	EXP01-C		Do not take the size of a pointer to determine the size of the pointed-to type

White Box Definitions

A weakness where code path has:

1. end statement that passes an identity of a dynamically allocated memory resource to a sizeof operator
2. start statement that allocates the dynamically allocated memory resource

References

Robert Seacord. "EXP01-A. Do not take the sizeof a pointer to determine the size of a type". <https://www.securecoding.cert.org/confluence/display/seccode/EXP01-A.+Do+not+take+the+sizeof+a+pointer+to+determine+the+size+of+a+type>.

Content History

Submissions

Submission Date	Submitter	Organization	Source
	CLASP		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-01		KDM Analytics	External
	added/updated white box definitions		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

BACK TO TOP

NULL Pointer Dereference

Risk

What might happen

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

Cause

How does it happen

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

General Recommendations

How to avoid it

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

Source Code Examples

Heuristic 2nd Order Buffer Overflow read

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Improper Validation of Array Index

Weakness ID: 129 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The product uses untrusted input when calculating or using an array index, but the product does not validate or incorrectly validates the index to ensure the index references a valid position within the array.

Alternate Terms

out-of-bounds array index

index-out-of-range

array index underflow

Time of Introduction

- Implementation

Applicable Platforms

Languages

C: *(Often)*

C++: *(Often)*

Language-independent

Common Consequences

Scope	Effect
Integrity Availability	Unchecked array indexing will very likely result in the corruption of relevant memory and perhaps instructions, leading to a crash, if the values are outside of the valid memory area.
Integrity	If the memory corrupted is data, rather than instructions, the system will continue to function with improper values.
Confidentiality Integrity	Unchecked array indexing can also trigger out-of-bounds read or write operations, or operations on the wrong objects; i.e., "buffer overflows" are not always the result. This may result in the exposure or modification of sensitive data.
Integrity	If the memory accessible by the attacker can be effectively controlled, it may be possible to execute arbitrary code, as with a standard buffer overflow and possibly without the use of large inputs if a precise index can be controlled.
Integrity Availability Confidentiality	A single fault could allow either an overflow (CWE-788) or underflow (CWE-786) of the array index. What happens next will depend on the type of operation being performed out of bounds, but can expose sensitive information, cause a system crash, or possibly lead to arbitrary code execution.

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

This weakness can often be detected using automated static analysis tools. Many modern tools use data flow analysis or constraint-based techniques to minimize the number of false positives.

Automated static analysis generally does not account for environmental considerations when reporting out-of-bounds memory operations. This can make it difficult for users to determine which warnings should be investigated first. For example, an analysis tool might report array index errors that originate from command line arguments in a program that is not expected to run with setuid or other special privileges.

Effectiveness: High

This is not a perfect solution, since 100% accuracy and coverage are not feasible.

Automated Dynamic Analysis

This weakness can be detected using dynamic tools and techniques that interact with the software using large test suites with many diverse inputs, such as fuzz testing (fuzzing), robustness testing, and fault injection. The software's operation may slow down, but it should not become unstable, crash, or generate incorrect results.

Black Box

Black box methods might not get the needed code coverage within limited time constraints, and a dynamic test might not produce any noticeable side effects even if it is successful.

Demonstrative Examples

Example 1

The following C/C++ example retrieves the sizes of messages for a pop3 mail server. The message sizes are retrieved from a socket that returns in a buffer the message number and the message size, the message number (num) and size (size) are extracted from the buffer and the message size is placed into an array using the message number for the array index.

(Bad Code)

Example Language: **C**

```
/* capture the sizes of all messages */
int getsizes(int sock, int count, int *sizes) {
...
char buf[BUFFER_SIZE];
int ok;
int num, size;

// read values from socket and added to sizes array
while ((ok = gen_recv(sock, buf, sizeof(buf))) == 0)
{

// continue read from socket until buf only contains '.'
if (DOTLINE(buf))
break;
else if (sscanf(buf, "%d %d", &num, &size) == 2)
sizes[num - 1] = size;
}
...
}
```

In this example the message number retrieved from the buffer could be a value that is outside the allowable range of indices for the array and could possibly be a negative number. Without proper validation of the value to be used for the array index an array overflow could occur and could potentially lead to unauthorized access to memory addresses and system crashes. The value of the array index should be validated to ensure that it is within the allowable range of indices for the array as in the following code.

(Good Code)

Example Language: **C**

```
/* capture the sizes of all messages */
int getsizes(int sock, int count, int *sizes) {
...
char buf[BUFFER_SIZE];
int ok;
int num, size;

// read values from socket and added to sizes array
while ((ok = gen_recv(sock, buf, sizeof(buf))) == 0)
{

// continue read from socket until buf only contains '.'
if (DOTLINE(buf))
```

```
break;
else if (sscanf(buf, "%d %d", &num, &size) == 2) {
if (num > 0 && num <= (unsigned)count)
sizes[num - 1] = size;
else
/* warn about possible attempt to induce buffer overflow */
report(stderr, "Warning: ignoring bogus data for message sizes returned by server.\n");
}
}
...
}
```

Example 2

In the code snippet below, an unchecked integer value is used to reference an object in an array.

(Bad Code)

Example Language: **Java**

```
public String getValue(int index) {
return array[index];
}
```

If index is outside of the range of the array, this may result in an ArrayIndexOutOfBounds Exception being raised.

Example 3

In the following Java example the method displayProductSummary is called from a Web service servlet to retrieve product summary information for display to the user. The servlet obtains the integer value of the product number from the user and passes it to the displayProductSummary method. The displayProductSummary method passes the integer value of the product number to the getProductSummary method which obtains the product summary from the array object containing the project summaries using the integer value of the product number as the array index.

(Bad Code)

Example Language: **Java**

```
// Method called from servlet to obtain product information
public String displayProductSummary(int index) {

String productSummary = new String("");

try {
String productSummary = getProductSummary(index);

} catch (Exception ex) {...}

return productSummary;
}

public String getProductSummary(int index) {
return products[index];
}
```

In this example the integer value used as the array index that is provided by the user may be outside the allowable range of indices for the array which may provide unexpected results or may comes the application to fail. The integer value used for the array index should be validated to ensure that it is within the allowable range of indices for the array as in the following code.

(Good Code)

Example Language: **Java**

```
// Method called from servlet to obtain product information
public String displayProductSummary(int index) {

String productSummary = new String("");
```

```
try {
String productSummary = getProductSummary(index);

} catch (Exception ex) {...}

return productSummary;
}

public String getProductSummary(int index) {
String productSummary = "";

if ((index >= 0) && (index < MAX_PRODUCTS)) {
productSummary = products[index];
}
else {
System.err.println("index is out of bounds");
throw new IndexOutOfBoundsException();
}

return productSummary;
}
```

An alternative in Java would be to use one of the collection objects such as ArrayList that will automatically generate an exception if an attempt is made to access an array index that is out of bounds.

(Good Code)

Example Language: **Java**

```
ArrayList productArray = new ArrayList(MAX_PRODUCTS);
...
try {
productSummary = (String) productArray.get(index);
} catch (IndexOutOfBoundsException ex) {...}
```

Observed Examples

Reference	Description
CVE-2005-0369	large ID in packet used as array index
CVE-2001-1009	negative array index as argument to POP LIST command
CVE-2003-0721	Integer signedness error leads to negative array index
CVE-2004-1189	product does not properly track a count and a maximum number, which can lead to resultant array index overflow.
CVE-2007-5756	chain: device driver for packet-capturing software allows access to an unintended IOCTL with resultant array index error.

Potential Mitigations

Phase: Architecture and Design

Strategies: Input Validation; Libraries or Frameworks

Use an input validation framework such as Struts or the OWASP ESAPI Validation API. If you use Struts, be mindful of weaknesses covered by the CWE-101 category.

Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.

Even though client-side checks provide minimal benefits with respect to server-side security, they are still useful. First, they can support intrusion detection. If the server receives input that should have been rejected by the client, then it may be an indication of an attack. Second, client-side error-checking can provide helpful feedback to the user about the expectations for valid input. Third, there may be a reduction in server-side processing time for accidental input errors, although this is typically a small savings.

Phase: Requirements

Strategy: Language Selection

Use a language with features that can automatically mitigate or eliminate out-of-bounds indexing errors.

For example, Ada allows the programmer to constrain the values of a variable and languages such as Java and Ruby will allow the programmer to handle exceptions when an out-of-bounds index is accessed.

Phase: Implementation

Strategy: Input Validation

Assume all input is malicious. Use an "accept known good" input validation strategy (i.e., use a whitelist). Reject any input that does not strictly conform to specifications, or transform it into something that does. Use a blacklist to reject any unexpected inputs and detect potential attacks.

When accessing a user-controlled array index, use a stringent range of values that are within the target array. Make sure that you do not allow negative values to be used. That is, verify the minimum as well as the maximum of the range of acceptable values.

Phase: Implementation

Be especially careful to validate your input when you invoke code that crosses language boundaries, such as from an interpreted language to native code. This could create an unexpected interaction between the language boundaries. Ensure that you are not violating any of the expectations of the language with which you are interfacing. For example, even though Java may not be susceptible to buffer overflows, providing a large argument in a call to native code might trigger an overflow.

Weakness Ordinalities

Ordinality	Description
Resultant	The most common condition situation leading to unchecked array indexing is the use of loop index variables as buffer indexes. If the end condition for the loop is subject to a flaw, the index can grow or shrink unbounded, therefore causing a buffer overflow or underflow. Another common situation leading to this condition is the use of a function's return value, or the resulting value of a calculation directly as an index in to a buffer.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	20	Improper Input Validation	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	189	Numeric Errors	Development Concepts699
ChildOf	Category	633	Weaknesses that Affect Memory	**Resource-specific Weaknesses (primary)631**
ChildOf	Category	738	CERT C Secure Coding Section 04 - Integers (INT)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
ChildOf	Category	802	2010 Top 25 - Risky Resource Management	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
CanPrecede	Weakness Class	119	Failure to Constrain Operations within the Bounds of a Memory Buffer	Research Concepts1000
CanPrecede	Weakness Variant	789	Uncontrolled Memory Allocation	Research Concepts1000
PeerOf	Weakness Base	124	Buffer Underwrite ('Buffer Underflow')	Research Concepts1000

Theoretical Notes

An improperly validated array index might lead directly to the always-incorrect behavior of "access of array using out-of-bounds index."

Affected Resources

- Memory

f Causal Nature

Explicit

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Unchecked array indexing
PLOVER			INDEX - Array index overflow
CERT C Secure Coding	ARR00-C		Understand how arrays work
CERT C Secure Coding	ARR30-C		Guarantee that array indices are within the valid range
CERT C Secure Coding	ARR38-C		Do not add or subtract an integer to a pointer if the resulting value does not refer to a valid array element
CERT C Secure Coding	INT32-C		Ensure that operations on signed integers do not result in overflow

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
100	Overflow Buffers	

References

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 5, "Array Indexing Errors" Page 144. 2nd Edition. Microsoft. 2002.

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	CLASP		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Sean Eidemiller	Cigital	External
	added/updated demonstrative examples		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Common Consequences		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Description, Name, Relationships		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Observed Examples, Other Notes, Potential Mitigations, Theoretical Notes, Weakness Ordinalities		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Demonstrative Examples, Detection Factors, Likelihood of Exploit, Potential Mitigations, References, Related Attack Patterns, Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Related Attack Patterns		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2009-10-29	Unchecked Array Indexing		

BACK TO TOP

Improper Access Control (Authorization)

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

Description

Description Summary

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

Extended Description

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

Alternate Terms

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

Time of Introduction

- Architecture and Design
- Implementation
- Operation

Applicable Platforms

Languages

Language-independent

Technology Classes

Web-Server: *(Often)*

Database-Server: *(Often)*

Modes of Introduction

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

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

Common Consequences

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

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

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

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

Effectiveness: Limited

Automated Dynamic Analysis

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

Manual Analysis

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

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

Effectiveness: Moderate

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

Demonstrative Examples

Example 1

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

(Bad Code)

Example Language: **Perl**

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

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

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

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

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

Observed Examples

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

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

Potential Mitigations

Phase: Architecture and Design

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

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

Phase: Architecture and Design

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

Phase: Architecture and Design

Strategy: Libraries or Frameworks

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

easier to avoid.

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

Phase: Architecture and Design

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

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

Phases: System Configuration; Installation

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

Relationships

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

Taxonomy Mappings

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

Related Attack Patterns

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

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

References

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

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

Content History

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

BACK TO TOP

Incorrect Permission Assignment for Critical Resource

Weakness ID: 732 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software specifies permissions for a security-critical resource in a way that allows that resource to be read or modified by unintended actors.

Extended Description

When a resource is given a permissions setting that provides access to a wider range of actors than required, it could lead to the disclosure of sensitive information, or the modification of that resource by unintended parties. This is especially dangerous when the resource is related to program configuration, execution or sensitive user data.

Time of Introduction

- Architecture and Design
- Implementation
- Installation
- Operation

Applicable Platforms

Languages

Language-independent

Modes of Introduction

The developer may set loose permissions in order to minimize problems when the user first runs the program, then create documentation stating that permissions should be tightened. Since system administrators and users do not always read the documentation, this can result in insecure permissions being left unchanged.

The developer might make certain assumptions about the environment in which the software runs - e.g., that the software is running on a single-user system, or the software is only accessible to trusted administrators. When the software is running in a different environment, the permissions become a problem.

Common Consequences

Scope	Effect
Confidentiality	An attacker may be able to read sensitive information from the associated resource, such as credentials or configuration information stored in a file.
Integrity	An attacker may be able to modify critical properties of the associated resource to gain privileges, such as replacing a world-writable executable with a Trojan horse.
Availability	An attacker may be able to destroy or corrupt critical data in the associated resource, such as deletion of records from a database.

Likelihood of Exploit

Medium to High

Detection Methods

Automated Static Analysis

Automated static analysis may be effective in detecting permission problems for system resources such as files, directories, shared memory, device interfaces, etc. Automated techniques may be able to detect the use of library functions that modify permissions, then analyze function calls for arguments that contain potentially insecure values.

However, since the software's intended security policy might allow loose permissions for certain operations (such as publishing a file on a web server), automated static analysis may produce some false positives - i.e., warnings that do not have any security consequences or require any code changes.

When custom permissions models are used - such as defining who can read messages in a particular forum in a bulletin board system - these can be difficult to detect using automated static analysis. It may be possible to define custom signatures that

identify any custom functions that implement the permission checks and assignments.

Automated Dynamic Analysis

Automated dynamic analysis may be effective in detecting permission problems for system resources such as files, directories, shared memory, device interfaces, etc.

However, since the software's intended security policy might allow loose permissions for certain operations (such as publishing a file on a web server), automated dynamic analysis may produce some false positives - i.e., warnings that do not have any security consequences or require any code changes.

When custom permissions models are used - such as defining who can read messages in a particular forum in a bulletin board system - these can be difficult to detect using automated dynamic analysis. It may be possible to define custom signatures that identify any custom functions that implement the permission checks and assignments.

Manual Static Analysis

Manual static analysis may be effective in detecting the use of custom permissions models and functions. The code could then be examined to identifying usage of the related functions. Then the human analyst could evaluate permission assignments in the context of the intended security model of the software.

Manual Dynamic Analysis

Manual dynamic analysis may be effective in detecting the use of custom permissions models and functions. The program could then be executed with a focus on exercising code paths that are related to the custom permissions. Then the human analyst could evaluate permission assignments in the context of the intended security model of the software.

Fuzzing

Fuzzing is not effective in detecting this weakness.

Demonstrative Examples

Example 1

The following code sets the umask of the process to 0 before creating a file and writing "Hello world" into the file.

(Bad Code)

Example Language: **C**

```
#define OUTFILE "hello.out"

umask(0);
FILE *out;
/* Ignore CWE-59 (link following) for brevity */
out = fopen(OUTFILE, "w");
if (out) {
fprintf(out, "hello world!\n");
fclose(out);
}
```

After running this program on a UNIX system, running the "ls -l" command might return the following output:

(Result)

```
-rw-rw-rw- 1 username 13 Nov 24 17:58 hello.out
```

The "rw-rw-rw-" string indicates that the owner, group, and world (all users) can read the file and write to it.

Example 2

The following code snippet might be used as a monitor to periodically record whether a web site is alive. To ensure that the file can always be modified, the code uses chmod() to make the file world-writable.

(Bad Code)

Example Language: **Perl**

```
$fileName = "secretFile.out";

if (-e $fileName) {
chmod 0777, $fileName;
}
```

```
my $outFH;
if (! open($outFH, ">>$fileName")) {
ExitError("Couldn't append to $fileName: $!");
}
my $dateString = FormatCurrentTime();
my $status = IsHostAlive("cwe.mitre.org");
print $outFH "$dateString cwe status: $status!\n";
close($outFH);
```

The first time the program runs, it might create a new file that inherits the permissions from its environment. A file listing might look like:

(Result)

```
-rw-r--r-- 1 username 13 Nov 24 17:58 secretFile.out
```

This listing might occur when the user has a default umask of 022, which is a common setting. Depending on the nature of the file, the user might not have intended to make it readable by everyone on the system.

The next time the program runs, however - and all subsequent executions - the chmod will set the file's permissions so that the owner, group, and world (all users) can read the file and write to it:

(Result)

```
-rw-rw-rw- 1 username 13 Nov 24 17:58 secretFile.out
```

Perhaps the programmer tried to do this because a different process uses different permissions that might prevent the file from being updated.

Example 3

The following command recursively sets world-readable permissions for a directory and all of its children:

(Bad Code)

Example Language: **Shell**

```
chmod -R ugo+r DIRNAME
```

If this command is run from a program, the person calling the program might not expect that all the files under the directory will be world-readable. If the directory is expected to contain private data, this could become a security problem.

Observed Examples

Reference	Description
CVE-2009-3482	Anti-virus product sets insecure "Everyone: Full Control" permissions for files under the "Program Files" folder, allowing attackers to replace executables with Trojan horses.
CVE-2009-3897	Product creates directories with 0777 permissions at installation, allowing users to gain privileges and access a socket used for authentication.
CVE-2009-3489	Photo editor installs a service with an insecure security descriptor, allowing users to stop or start the service, or execute commands as SYSTEM.
CVE-2009-3289	Library function copies a file to a new target and uses the source file's permissions for the target, which is incorrect when the source file is a symbolic link, which typically has 0777 permissions.
CVE-2009-0115	Device driver uses world-writable permissions for a socket file, allowing attackers to inject arbitrary commands.
CVE-2009-1073	LDAP server stores a cleartext password in a world-readable file.
CVE-2009-0141	Terminal emulator creates TTY devices with world-writable permissions, allowing an attacker to write to the terminals of other users.

CVE-2008-0662	VPN product stores user credentials in a registry key with "Everyone: Full Control" permissions, allowing attackers to steal the credentials.
CVE-2008-0322	Driver installs its device interface with "Everyone: Write" permissions.
CVE-2009-3939	Driver installs a file with world-writable permissions.
CVE-2009-3611	Product changes permissions to 0777 before deleting a backup; the permissions stay insecure for subsequent backups.
CVE-2007-6033	Product creates a share with "Everyone: Full Control" permissions, allowing arbitrary program execution.
CVE-2007-5544	Product uses "Everyone: Full Control" permissions for memory-mapped files (shared memory) in inter-process communication, allowing attackers to tamper with a session.
CVE-2005-4868	Database product uses read/write permissions for everyone for its shared memory, allowing theft of credentials.
CVE-2004-1714	Security product uses "Everyone: Full Control" permissions for its configuration files.
CVE-2001-0006	"Everyone: Full Control" permissions assigned to a mutex allows users to disable network connectivity.
CVE-2002-0969	Chain: database product contains buffer overflow that is only reachable through a .ini configuration file - which has "Everyone: Full Control" permissions.

Potential Mitigations

Phase: Implementation

When using a critical resource such as a configuration file, check to see if the resource has insecure permissions (such as being modifiable by any regular user), and generate an error or even exit the software if there is a possibility that the resource could have been modified by an unauthorized party.

Phase: Architecture and Design

Divide your application into anonymous, normal, privileged, and administrative areas. Reduce the attack surface by carefully defining distinct user groups, privileges, and/or roles. Map these against data, functionality, and the related resources. Then set the permissions accordingly. This will allow you to maintain more fine-grained control over your resources.

Phases: Implementation; Installation

During program startup, explicitly set the default permissions or umask to the most restrictive setting possible. Also set the appropriate permissions during program installation. This will prevent you from inheriting insecure permissions from any user who installs or runs the program.

Phase: System Configuration

For all configuration files, executables, and libraries, make sure that they are only readable and writable by the software's administrator.

Phase: Documentation

Do not suggest insecure configuration changes in your documentation, especially if those configurations can extend to resources and other software that are outside the scope of your own software.

Phase: Installation

Do not assume that the system administrator will manually change the configuration to the settings that you recommend in the manual.

Phase: Testing

Use tools and techniques that require manual (human) analysis, such as penetration testing, threat modeling, and interactive tools that allow the tester to record and modify an active session. These may be more effective than strictly automated techniques. This is especially the case with weaknesses that are related to design and business rules.

Phase: Testing

Use monitoring tools that examine the software's process as it interacts with the operating system and the network. This technique is useful in cases when source code is unavailable, if the software was not developed by you, or if you want to verify that the build phase did not introduce any new weaknesses. Examples include debuggers that directly attach to the running process; system-call tracing utilities such as truss (Solaris) and strace (Linux); system activity monitors such as FileMon, RegMon, Process Monitor, and other Sysinternals utilities (Windows); and sniffers and protocol analyzers that monitor network traffic.

Attach the monitor to the process and watch for library functions or system calls on OS resources such as files, directories, and shared memory. Examine the arguments to these calls to infer which permissions are being used.

Note that this technique is only useful for permissions issues related to system resources. It is not likely to detect application-level business rules that are related to permissions, such as if a user of a blog system marks a post as "private," but the blog system inadvertently marks it as "public."

Phases: Testing; System Configuration

Ensure that your software runs properly under the Federal Desktop Core Configuration (FDCC) or an equivalent hardening configuration guide, which many organizations use to limit the attack surface and potential risk of deployed software.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	275	Permission Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	668	Exposure of Resource to Wrong Sphere	**Research Concepts (primary)1000**
ChildOf	Category	753	2009 Top 25 - Porous Defenses	**Weaknesses in the 2009 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)750**
ChildOf	Category	803	2010 Top 25 - Porous Defenses	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
RequiredBy	Compound Element: Composite	689	Permission Race Condition During Resource Copy	Research Concepts1000
ParentOf	Weakness Variant	276	Incorrect Default Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	277	Insecure Inherited Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	278	Insecure Preserved Inherited Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	279	Incorrect Execution-Assigned Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Base	281	Improper Preservation of Permissions	**Research Concepts (primary)1000**

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
232	Exploitation of Privilege/Trust	
1	Accessing Functionality Not Properly Constrained by ACLs	
17	Accessing, Modifying or Executing Executable Files	
60	Reusing Session IDs (aka Session Replay)	
61	Session Fixation	
62	Cross Site Request Forgery (aka Session Riding)	
122	Exploitation of Authorization	
180	Exploiting Incorrectly Configured Access Control Security Levels	
234	Hijacking a privileged process	

References

Mark Dowd, John McDonald and Justin Schuh. "The Art of Software Security Assessment". Chapter 9, "File Permissions." Page 495.. 1st Edition. Addison Wesley. 2006.

John Viega and Gary McGraw. "Building Secure Software". Chapter 8, "Access Control." Page 194.. 1st Edition. Addison-Wesley. 2002.

Maintenance Notes

The relationships between privileges, permissions, and actors (e.g. users and groups) need further refinement within the Research view. One complication is that these concepts apply to two different pillars, related to control of resources (CWE-664) and protection mechanism failures (CWE-396).

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
2008-09-08			Internal CWE Team
	new weakness-focused entry for Research view.		
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2009-01-12	CWE Content Team	MITRE	Internal
	updated Description, Likelihood of Exploit, Name, Potential Mitigations, Relationships		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Potential Mitigations, Related Attack Patterns		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Name		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Demonstrative Examples, Detection Factors, Modes of Introduction, Observed Examples, Potential Mitigations, References		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Potential Mitigations, Related Attack Patterns		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2009-01-12	Insecure Permission Assignment for Resource		
2009-05-27	Insecure Permission Assignment for Critical Resource		

BACK TO TOP

Exposure of System Data to Unauthorized Control Sphere

Risk

What might happen

System data can provide attackers with valuable insights on systems and services they are targeting - any type of system data, from service version to operating system fingerprints, can assist attackers to hone their attack, correlate data with known vulnerabilities or focus efforts on developing new attacks against specific technologies.

Cause

How does it happen

System data is read and subsequently exposed where it might be read by untrusted entities.

General Recommendations

How to avoid it

Consider the implications of exposure of the specified input, and expected level of access to the specified output. If not required, consider removing this code, or modifying exposed information to exclude potentially sensitive system data.

Source Code Examples

Java

Leaking Environment Variables in JSP Web-Page

```
String envVarValue = System.getenv(envVar);
if (envVarValue == null) {
    out.println("Environment variable is not defined:");
    out.println(System.getenv());
} else {
    //[..]
};
```

TOCTOU

Risk

What might happen

At best, a Race Condition may cause errors in accuracy, overidden values or unexpected behavior that may result in denial-of-service. At worst, it may allow attackers to retrieve data or bypass security processes by replaying a controllable Race Condition until it plays out in their favor.

Cause

How does it happen

Race Conditions occur when a public, single instance of a resource is used by multiple concurrent logical processes. If the these logical processes attempt to retrieve and update the resource without a timely management system, such as a lock, a Race Condition will occur.

An example for when a Race Condition occurs is a resource that may return a certain value to a process for further editing, and then updated by a second process, resulting in the original process' data no longer being valid. Once the original process edits and updates the incorrect value back into the resource, the second process' update has been overwritten and lost.

General Recommendations

How to avoid it

When sharing resources between concurrent processes across the application ensure that these resources are either thread-safe, or implement a locking mechanism to ensure expected concurrent activity.

Source Code Examples

Java

Different Threads Increment and Decrement The Same Counter Repeatedly, Resulting in a Race Condition

```
	public static int counter = 0;
	public static void start() throws InterruptedException {
		incrementCounter ic;
		decrementCounter dc;
		while(counter == 0) {
			counter = 0;
			ic = new incrementCounter();
			dc = new decrementCounter();
			ic.start();
			dc.start();
			ic.join();
			dc.join();
		}
		System.out.println(counter); //Will stop and return either -1 or 1 due to race
condition over counter
	}

	public static class incrementCounter extends Thread {
	    public void run() {
		counter++;
	    }
```

```
    }

    public static class decrementCounter extends Thread {
        public void run() {
           counter--;
        }
    }
```

Different Threads Increment and Decrement The Same Thread-Safe Counter Repeatedly, Never Resulting in a Race Condition

```
    public static int counter = 0;
    public static Object lock = new Object();

    public static void start() throws InterruptedException {
           incrementCounter ic;
           decrementCounter dc;
           while(counter == 0) { // because of proper locking, this condition is never false
                  counter = 0;
                  ic = new incrementCounter();
                  dc = new decrementCounter();
                  ic.start();
                  dc.start();
                  ic.join();
                  dc.join();
           }
           System.out.println(counter); // Never reached
    }

    public static class incrementCounter extends Thread {
        public void run() {
           synchronized (lock) {
                  counter++;
           }
        }
    }

    public static class decrementCounter extends Thread {
        public void run() {
           synchronized (lock) {
                  counter--;
           }
        }
    }
```

Scanned Languages

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