nmap Scan Report

Project Name	nmap
Scan Start	Friday, June 21, 2024 10:51:05 PM
Preset	Checkmarx Default
Scan Time	00h:10m:04s
Lines Of Code Scanned	24654
Files Scanned	17
Report Creation Time	Friday, June 21, 2024 11:04:09 PM
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051
Team	CxServer
Checkmarx Version	8.7.0
Scan Type	Full
Source Origin	LocalPath
Density	2/100 (Vulnerabilities/LOC)
Visibility	Public

Filter Settings

Severity

Included: High, Medium, Low, Information

Excluded: None

Result State

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

Excluded: None

Assigned to

Included: All

Categories

Included:

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

Excluded:

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

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

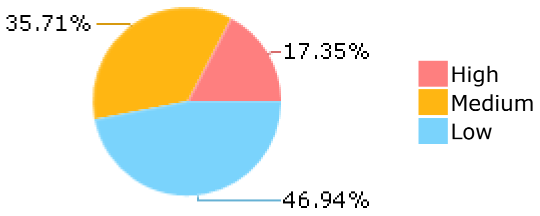
Results Limit

Results limit per query was set to 50

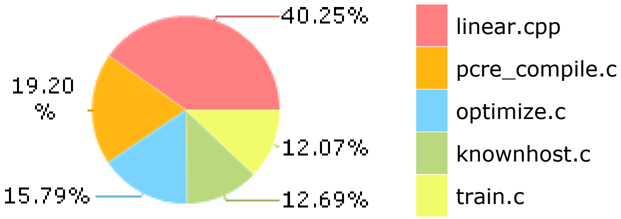
Selected Queries

Selected queries are listed in Result Summary

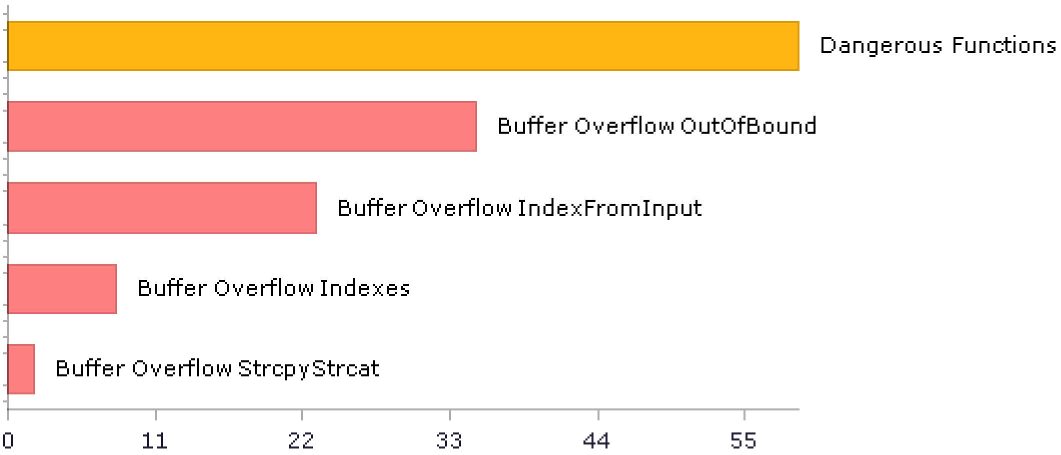
Result Summary



Most Vulnerable Files



Top 5 Vulnerabilities



Scan Summary - OWASP Top 10 2017

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

Category	Threat Agent	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection	App. Specific	EASY	COMMON	EASY	SEVERE	App. Specific	168	65
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	66	66
A3-Sensitive Data Exposure	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	6	6
A4-XML External Entities (XXE)	App. Specific	AVERAGE	COMMON	EASY	SEVERE	App. Specific	0	0
A5-Broken Access Control*	App. Specific	AVERAGE	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A6-Security Misconfiguration	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A7-Cross-Site Scripting (XSS)	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A8-Insecure Deserialization	App. Specific	DIFFICULT	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A9-Using Components with Known Vulnerabilities*	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	MODERATE	App. Specific	59	59
A10-Insufficient Logging & Monitoring	App. Specific	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	App. Specific	0	0

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

Scan Summary - OWASP Top 10 2013

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

Category	Threat Agent	Attack Vectors	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	AVERAGE	SEVERE	ALL DATA	0	0
A2-Broken Authentication and Session Management	EXTERNAL, INTERNAL USERS	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	AFFECTED DATA AND FUNCTIONS	0	0
A3-Cross-Site Scripting (XSS)	EXTERNAL, INTERNAL, ADMIN USERS	AVERAGE	VERY WIDESPREAD	EASY	MODERATE	AFFECTED DATA AND SYSTEM	0	0
A4-Insecure Direct Object References	SYSTEM USERS	EASY	COMMON	EASY	MODERATE	EXPOSED DATA	0	0
A5-Security Misconfiguration	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	EASY	MODERATE	ALL DATA AND SYSTEM	0	0
A6-Sensitive Data Exposure	EXTERNAL, INTERNAL, ADMIN USERS, USERS BROWSERS	DIFFICULT	UNCOMMON	AVERAGE	SEVERE	EXPOSED DATA	0	0
A7-Missing Function Level Access Control*	EXTERNAL, INTERNAL USERS	EASY	COMMON	AVERAGE	MODERATE	EXPOSED DATA AND FUNCTIONS	0	0
A8-Cross-Site Request Forgery (CSRF)	USERS BROWSERS	AVERAGE	COMMON	EASY	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0
A9-Using Components with Known Vulnerabilities*	EXTERNAL USERS, AUTOMATED TOOLS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	59	59
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	3	3
PCI DSS (3.2) - 6.5.2 - Buffer overflows	130	53
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.	6	6
Audit And Accountability*	Organizations must: (i) create, protect, and retain information system audit records to the extent needed to enable the monitoring, analysis, investigation, and reporting of unlawful, unauthorized, or inappropriate information system activity; and (ii) ensure that the actions of individual information system users can be uniquely traced to those users so they can be held accountable for their actions.	1	1
Configuration Management	Organizations must: (i) establish and maintain baseline configurations and inventories of organizational information systems (including hardware, software, firmware, and documentation) throughout the respective system development life cycles; and (ii) establish and enforce security configuration settings for information technology products employed in organizational information systems.	1	1
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.	60	60
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.	6	6
System And Communications Protection	Organizations must: (i) monitor, control, and protect organizational communications (i.e., information transmitted or received by organizational information systems) at the external boundaries and key internal boundaries of the information systems; and (ii) employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational information systems.	0	0
System And Information Integrity	Organizations must: (i) identify, report, and correct information and information system flaws in a timely manner; (ii) provide protection from malicious code at appropriate locations within organizational information systems; and (iii) monitor information system security alerts and advisories and take appropriate actions in response.	3	3

***** 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)	67	67
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)	6	6
SC-4 Information in Shared Resources (P1)	0	0
SC-5 Denial of Service Protection (P1)*****	40	25
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*****	118	36
SI-11 Error Handling (P2)*****	26	26
SI-15 Information Output Filtering (P0)	0	0
SI-16 Memory Protection (P1)	3	3

***** 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

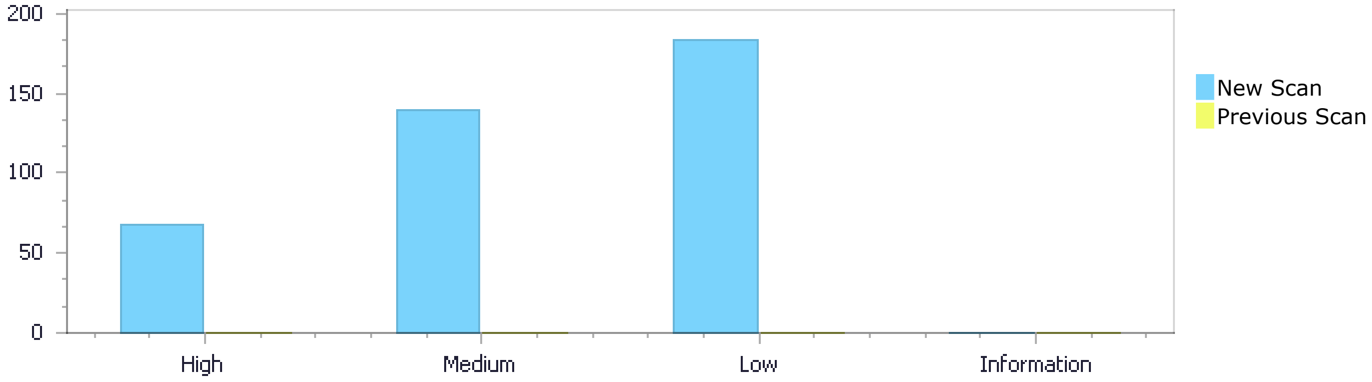
Category	Issues Found	Best Fix Locations
Must audit	0	0
Check	0	0
Optional	0	0

Results Distribution By Status

First scan of the project

	High	Medium	Low	Information	Total
New Issues	68	140	184	0	392
Recurrent Issues	0	0	0	0	0
Total	68	140	184	0	392

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	68	140	184	0	392
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	68	140	184	0	392

Result Summary

Vulnerability Type	Occurrences	Severity
Buffer Overflow OutOfBound	35	High
Buffer Overflow IndexFromInput	23	High
Buffer Overflow Indexes	8	High
Buffer Overflow StrcpyStrcat	2	High
Dangerous Functions	59	Medium

Buffer Overflow boundcpy WrongSizeParam	34	Medium
Use of Zero Initialized Pointer	14	Medium
Memory Leak	10	Medium
Use of Uninitialized Variable	6	Medium
Stored Buffer Overflow fgets	4	Medium
Wrong Size t Allocation	4	Medium
Short Overflow	3	Medium
Stored Buffer Overflow boundcpy	3	Medium
Char Overflow	2	Medium
Divide By Zero	1	Medium
Improper Resource Access Authorization	60	Low
Heuristic Buffer Overflow malloc	36	Low
Unchecked Return Value	26	Low
Unchecked Array Index	14	Low
Heuristic 2nd Order Buffer Overflow malloc	10	Low
NULL Pointer Dereference	9	Low
Incorrect Permission Assignment For Critical Resources	6	Low
TOCTOU	6	Low
Use of Insufficiently Random Values	6	Low
Potential Off by One Error in Loops	3	Low
Sizeof Pointer Argument	2	Low
Use of Sizeof On a Pointer Type	2	Low
Arithmenic Operation On Boolean	1	Low
Exposure of System Data to Unauthorized Control Sphere	1	Low
Inconsistent Implementations	1	Low
Potential Precision Problem	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
nmap/pcre_compile.c	61
nmap/optimize.c	28
nmap/ncat_main.c	27
nmap/train.c	24
nmap/linear.cpp	23
nmap/knownhost.c	21
nmap/lobject.c	16
nmap/puff.c	6
nmap/blast.c	2

Scan Results Details

Buffer Overflow OutOfBound

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

Categories

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

Description

Buffer Overflow OutOfBound\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=358
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2802
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.         uschar pbits[32];
....
2802.             for (c = 0; c < 32; c++) pbits[c] &= ~cbits[c +
taboffset];
```

Buffer Overflow OutOfBound\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=359
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination

File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2800
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.          uschar pbits[32];
....
2800.              for (c = 0; c < 32; c++) pbits[c] |= cbits[c + 
taboffset];
```

Buffer Overflow OutOfBound\Path 3:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=360
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2816
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.          uschar pbits[32];
....
2816.            for (c = 0; c < 32; c++) classbits[c] |= ~pbits[c];
```

Buffer Overflow OutOfBound\Path 4:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=361
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2818
Object	pbits	c

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.           uschar pbits[32];
....
2818.             for (c = 0; c < 32; c++) classbits[c] |= pbits[c];
```

Buffer Overflow OutOfBound\Path 5:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=362
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2816
Object	pbits	c

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.           uschar pbits[32];
....
2816.             for (c = 0; c < 32; c++) classbits[c] |= ~pbits[c];
```

Buffer Overflow OutOfBound\Path 6:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=363
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2818
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.          uschar pbits[32];
....
2818.            for (c = 0; c < 32; c++) classbits[c] |= pbits[c];
```

Buffer Overflow OutOfBound\Path 7:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=364
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2861
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.          uschar pbits[32];
....
2861.              for (c = 0; c < 32; c++) classbits[c] |=
cbits[c+cbit_digit];
```

Buffer Overflow OutOfBound\Path 8:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=365
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2866
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.           uschar pbits[32];
....
2866.               for (c = 0; c < 32; c++) classbits[c] |=
~cbits[c+cbit_digit];
```

Buffer Overflow OutOfBound\Path 9:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=366
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2870
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.           uschar pbits[32];
....
2870.               for (c = 0; c < 32; c++) classbits[c] |=
cbits[c+cbit_word];
```

Buffer Overflow OutOfBound\Path 10:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=500

	51&pathid=367
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2875
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.         uschar pbits[32];
....
2875.             for (c = 0; c < 32; c++) classbits[c] |=
~cbits[c+cbit_word];
```

Buffer Overflow OutOfBound\Path 11:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=368
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2879
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.         uschar pbits[32];
....
2879.             for (c = 0; c < 32; c++) classbits[c] |=
cbits[c+cbit_space];
```

Buffer Overflow OutOfBound\Path 12:

Severity	High
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=369
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2885
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.          uschar pbits[32];
....
2885.              for (c = 0; c < 32; c++) classbits[c] |=
~cbits[c+cbit_space];
```

Buffer Overflow OutOfBound\Path 13:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=370
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	2940
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.          uschar pbits[32];
....
2940.                classbits[c] |= x;
```

Buffer Overflow OutOfBound\Path 14:

Severity	High

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=371
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	3007
Object	pbits	c

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.          uschar pbits[32];
....
3007.                  classbits[c] |= x;
```

Buffer Overflow OutOfBound\Path 15:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=372
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	3429
Object	pbits	c

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.          uschar pbits[32];
....
3429.          for (c = 0; c < 32; c++) code[c] = ~classbits[c];
```

Buffer Overflow OutOfBound\Path 16:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=373
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to pbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2750	3429
Object	pbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2750.          uschar pbits[32];
....
3429.          for (c = 0; c < 32; c++) code[c] = ~classbits[c];
```

Buffer Overflow OutOfBound\Path 17:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=374
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	2818
Object	classbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.  uschar classbits[32];
....
2818.          for (c = 0; c < 32; c++) classbits[c] |= pbits[c];
```

Buffer Overflow OutOfBound\Path 18:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=375
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	2816
Object	classbits	c

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.   uschar classbits[32];
....
2816.             for (c = 0; c < 32; c++) classbits[c] |= ~pbits[c];
```

Buffer Overflow OutOfBound\Path 19:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=376
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	2861
Object	classbits	c

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.   uschar classbits[32];
....
2861.               for (c = 0; c < 32; c++) classbits[c] |=
cbits[c+cbit_digit];
```

Buffer Overflow OutOfBound\Path 20:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=377
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	2866
Object	classbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.  uschar classbits[32];
....
2866.               for (c = 0; c < 32; c++) classbits[c] |=
~cbits[c+cbit_digit];
```

Buffer Overflow OutOfBound\Path 21:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=378
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	2870
Object	classbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.  uschar classbits[32];
....
2870.                for (c = 0; c < 32; c++) classbits[c] |=
cbits[c+cbit_word];
```

Buffer Overflow OutOfBound\Path 22:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=379
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	2875
Object	classbits	c

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.  uschar classbits[32];
....
2875.                for (c = 0; c < 32; c++) classbits[c] |=
~cbits[c+cbit_word];
```

Buffer Overflow OutOfBound\Path 23:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=380
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	2879
Object	classbits	c

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.  uschar classbits[32];
....
2879.              for (c = 0; c < 32; c++) classbits[c] |=
cbits[c+cbit_space];
```

Buffer Overflow OutOfBound\Path 24:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=381
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	2885
Object	classbits	c

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.  uschar classbits[32];
....
2885.              for (c = 0; c < 32; c++) classbits[c] |=
~cbits[c+cbit_space];
```

Buffer Overflow OutOfBound\Path 25:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=382
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	2940
Object	classbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.   uschar classbits[32];
....
2940.                   classbits[c] |= x;
```

Buffer Overflow OutOfBound\Path 26:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=383
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	3007
Object	classbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.   uschar classbits[32];
....
3007.                   classbits[c] |= x;
```

Buffer Overflow OutOfBound\Path 27:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=384
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	3429

Object	classbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.   uschar classbits[32];
....
3429.           for (c = 0; c < 32; c++) code[c] = ~classbits[c];
```

Buffer Overflow OutOfBound\Path 28:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=385
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to classbits, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2376	3429
Object	classbits	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2376.   uschar classbits[32];
....
3429.           for (c = 0; c < 32; c++) code[c] = ~classbits[c];
```

Buffer Overflow OutOfBound\Path 29:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=386
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to mcbuffer, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c

Line	2437	5202
Object	mcbuffer	c

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2437.    uschar mcbuffer[8];
....
5202.      for (c = 0; c < mclength; c++) *code++ = mcbuffer[c];
```

Buffer Overflow OutOfBound\Path 30:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=387
Status	New

The size of the buffer used by compile_branch in mclength, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to mcbuffer, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2437	5192
Object	mcbuffer	mclength

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2437.    uschar mcbuffer[8];
....
5192.          mcbuffer[mclength++] = *(++ptr);
```

Buffer Overflow OutOfBound\Path 31:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=388
Status	New

The size of the buffer used by compile_branch in c, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to mcbuffer, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination

File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2437	3429
Object	mcbuffer	c

Code Snippet
File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2437.    uschar mcbuffer[8];
....
3429.          for (c = 0; c < 32; c++) code[c] = ~classbits[c];
```

Buffer Overflow OutOfBound\Path 32:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=389
Status	New

The size of the buffer used by codes in symbol, at line 436 of nmap/puff.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that codes passes to lens, at line 436 of nmap/puff.c, to overwrite the target buffer.

	Source	Destination
File	nmap/puff.c	nmap/puff.c
Line	443	477
Object	lens	symbol

Code Snippet
File Name	nmap/puff.c
Method	local int codes(struct state *s,

```
....
443.      static const short lens[29] = { /* Size base for length codes
257..285 */
....
477.              len = lens[symbol] + bits(s, lext[symbol]);
```

Buffer Overflow OutOfBound\Path 33:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=390
Status	New

The size of the buffer used by codes in symbol, at line 436 of nmap/puff.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that codes passes to lext, at line 436 of nmap/puff.c, to overwrite the target buffer.

	Source	Destination
File	nmap/puff.c	nmap/puff.c
Line	446	477
Object	lext	symbol

Code Snippet

File Name	nmap/puff.c
Method	local int codes(struct state *s,

```
....
446.      static const short lext[29] = { /* Extra bits for length codes
257..285 */
....
477.              len = lens[symbol] + bits(s, lext[symbol]);
```

Buffer Overflow OutOfBound\Path 34:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=391
Status	New

The size of the buffer used by codes in symbol, at line 436 of nmap/puff.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that codes passes to dists, at line 436 of nmap/puff.c, to overwrite the target buffer.

	Source	Destination
File	nmap/puff.c	nmap/puff.c
Line	449	483
Object	dists	symbol

Code Snippet

File Name	nmap/puff.c
Method	local int codes(struct state *s,

```
....
449.      static const short dists[30] = { /* Offset base for distance
codes 0..29 */
....
483.              dist = dists[symbol] + bits(s, dext[symbol]);
```

Buffer Overflow OutOfBound\Path 35:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=392
Status	New

The size of the buffer used by codes in symbol, at line 436 of nmap/puff.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that codes passes to dext, at line 436 of nmap/puff.c, to overwrite the target buffer.

	Source	Destination
File	nmap/puff.c	nmap/puff.c
Line	453	483
Object	dext	symbol

Code Snippet
File Name nmap/puff.c
Method local int codes(struct state *s,

```
....
453.      static const short dext[30] = { /* Extra bits for distance
codes 0..29 */
....
483.              dist = dists[symbol] + bits(s, dext[symbol]);
```

Buffer Overflow IndexFromInput

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

Categories

OWASP Top 10 2017: A1-Injection

Description

Buffer Overflow IndexFromInput\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=45
Status	New

The size of the buffer used by main in BinaryExpr, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	847
Object	argc	BinaryExpr

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

```
....
218.   int main(int argc, char *argv[])
....
847.            o.portno = parseport(argv[optind + 1], max_port,
"port");
```

Buffer Overflow IndexFromInput\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=46
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	887
Object	argc	optind

Code Snippet

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

```
....
218.   int main(int argc, char *argv[])
....
887.          o.target = argv[optind];
```

Buffer Overflow IndexFromInput\Path 3:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=47
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	882
Object	argc	optind

Code Snippet

File Name	nmap/ncat_main.c

Method int main(int argc, char *argv[])

```
....
218.  int main(int argc, char *argv[])
....
882.                  o.portno = parseport(argv[optind], max_port,
"port");
```

Buffer Overflow IndexFromInput\Path 4:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=48
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	881
Object	argc	optind

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

```
....
218.  int main(int argc, char *argv[])
....
881.              if (argv[optind][rc] == '\0' && rc <= 5) {
```

Buffer Overflow IndexFromInput\Path 5:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=49
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	879
Object	argc	optind

Code Snippet

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

```
....
218.  int main(int argc, char *argv[])
....
879.              rc = strspn(argv[optind], "1234567890");
```

Buffer Overflow IndexFromInput\Path 6:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=50
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	873
Object	argc	optind

Code Snippet

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

```
....
218.  int main(int argc, char *argv[])
....
873.              o.sslservername = o.target = argv[optind];
```

Buffer Overflow IndexFromInput\Path 7:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=51
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	869
Object	argc	optind

Code Snippet

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

```
....
218.  int main(int argc, char *argv[])
....
869.                  bye("Invalid CID \"%s\".", argv[optind]);
```

Buffer Overflow IndexFromInput\Path 8:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=52
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	867
Object	argc	optind

Code Snippet

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

```
....
218.  int main(int argc, char *argv[])
....
867.              long_cid = strtol(argv[optind], NULL, 10);
```

Buffer Overflow IndexFromInput\Path 9:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=53
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	855
Object	argc	optind

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

```
....
218.  int main(int argc, char *argv[])
....
855.              o.sslservername = o.target = argv[optind];
```

Buffer Overflow IndexFromInput\Path 10:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=54
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	853
Object	argc	optind

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

```
....
218.  int main(int argc, char *argv[])
....
853.              NCAT_INIT_SUN(&targetaddrs->addr, argv[optind]);
```

Buffer Overflow IndexFromInput\Path 11:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=55
Status	New

The size of the buffer used by main in BinaryExpr, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	911

Object	argc	BinaryExpr

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

```
....
218.  int main(int argc, char *argv[])
....
911.              o.portno = parseport(argv[optind + 1], max_port,
"port");
```

Buffer Overflow IndexFromInput\Path 12:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=56
Status	New

The size of the buffer used by main in rc, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argv, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	881
Object	argv	rc

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

```
....
218.  int main(int argc, char *argv[])
....
881.              if (argv[optind][rc] == '\0' && rc <= 5) {
```

Buffer Overflow IndexFromInput\Path 13:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=57
Status	New

The size of the buffer used by decomp in PostfixExpr, at line 282 of nmap/blast.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to stdin, at line 446 of nmap/blast.c, to overwrite the target buffer.

	Source	Destination
File	nmap/blast.c	nmap/blast.c

Line	453	371
Object	stdin	PostfixExpr

Code Snippet

File Name	nmap/blast.c
Method	int main(void)

```
....
453.        ret = blast(inf, stdin, outf, stdout, &left, NULL);
```

File Name	nmap/blast.c
Method	local int decomp(struct state *s)

```
....
371.                s->out[s->next++] = symbol;
```

Buffer Overflow IndexFromInput\Path 14:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=58
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2224	2296
Object	fp	BinaryExpr

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2224.               fscanf(fp,"%80s",cmd);
....
2296.                       fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Buffer Overflow IndexFromInput\Path 15:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=59
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2227	2296
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2227.                   fscanf(fp,"%80s",cmd);
....
2296.                   fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Buffer Overflow IndexFromInput\Path 16:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=60
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2296
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2247.                   fscanf(fp,"%d",&nr_class);
....
2296.                   fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Buffer Overflow IndexFromInput\Path 17:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=61

Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2296
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2252.                     fscanf(fp,"%d",&nr_feature);
....
2296.                     fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Buffer Overflow IndexFromInput\Path 18:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=62
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2257	2296
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2257.                     fscanf(fp,"%lf",&bias);
....
2296.                     fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Buffer Overflow IndexFromInput\Path 19:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=500

	51&pathid=63
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2269	2296
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2269.                               fscanf(fp,"%d",&model_->label[i]);
....
2296.                        fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Buffer Overflow IndexFromInput\Path 20:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=64
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2296
Object	Address	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2247.                        fscanf(fp,"%d",&nr_class);
....
2296.                        fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Buffer Overflow IndexFromInput\Path 21:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=65
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2296
Object	Address	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2252.                    fscanf(fp,"%d",&nr_feature);
....
2296.                    fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Buffer Overflow IndexFromInput\Path 22:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=66
Status	New

The size of the buffer used by knownhost_add in keylen, at line 134 of nmap/knownhost.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that libssh2_knownhost_readfile passes to buf, at line 953 of nmap/knownhost.c, to overwrite the target buffer.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	968	204
Object	buf	keylen

Code Snippet
File Name nmap/knownhost.c
Method libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
968.           while(fgets(buf, sizeof(buf), file)) {
```

File Name nmap/knownhost.c
Method knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
204.         entry->key[keylen] = 0; /* force a terminating zero
trailer */
```

Buffer Overflow IndexFromInput\Path 23:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=67
Status	New

The size of the buffer used by knownhost_add in commentlen, at line 134 of nmap/knownhost.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that libssh2_knownhost_readfile passes to buf, at line 953 of nmap/knownhost.c, to overwrite the target buffer.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	968	241
Object	buf	commentlen

Code Snippet

File Name nmap/knownhost.c

Method libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
968.         while(fgets(buf, sizeof(buf), file)) {
```

File Name nmap/knownhost.c

Method knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
241.         entry->comment[commentlen] = 0; /* force a terminating
zero trailer */
```

Buffer Overflow Indexes

Query Path:

CPP\Cx\CPP Buffer Overflow\Buffer Overflow Indexes 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 Indexes\Path 1:

Severity	High
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=1
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	847
Object	argc	optind

Code Snippet

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

```
....
218.  int main(int argc, char *argv[])
....
847.            o.portno = parseport(argv[optind + 1], max_port, 
"port");
```

Buffer Overflow Indexes\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=2
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	887
Object	argc	optind

Code Snippet

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

```
....
218.  int main(int argc, char *argv[])
....
887.          o.target = argv[optind];
```

Buffer Overflow Indexes\Path 3:

Severity	High

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=3
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	873
Object	argc	optind

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

```
....
218.  int main(int argc, char *argv[])
....
873.              o.sslservername = o.target = argv[optind];
```

Buffer Overflow Indexes\Path 4:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=4
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	869
Object	argc	optind

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

```
....
218.  int main(int argc, char *argv[])
....
869.                  bye("Invalid CID \"%s\".", argv[optind]);
```

Buffer Overflow Indexes\Path 5:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=5
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	867
Object	argc	optind

Code Snippet

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

```
....
218.   int main(int argc, char *argv[])
....
867.                  long_cid = strtol(argv[optind], NULL, 10);
```

Buffer Overflow Indexes\Path 6:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=6
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	855
Object	argc	optind

Code Snippet

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

```
....
218.   int main(int argc, char *argv[])
....
855.                  o.sslservername = o.target = argv[optind];
```

Buffer Overflow Indexes\Path 7:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=7
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	853
Object	argc	optind

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

```
....
218.  int main(int argc, char *argv[])
....
853.              NCAT_INIT_SUN(&targetaddrs->addr, argv[optind]);
```

Buffer Overflow Indexes\Path 8:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=8
Status	New

The size of the buffer used by main in optind, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to argc, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	218	911
Object	argc	optind

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

```
....
218.  int main(int argc, char *argv[])
....
911.              o.portno = parseport(argv[optind + 1], max_port, 
"port");
```

Buffer Overflow StrcpyStrcat

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

Categories

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

Description

Buffer Overflow StrcpyStrcat\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=43
Status	New

The size of the buffer used by parse_command_line in argv, at line 140 of nmap/train.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that parse_command_line passes to argv, at line 140 of nmap/train.c, to overwrite the target buffer.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	140	215
Object	argv	argv

Code Snippet

File Name	nmap/train.c
Method	void parse_command_line(int argc, char **argv, char *input_file_name, char *model_file_name)

```
....
140.  void parse_command_line(int argc, char **argv, char
*input_file_name, char *model_file_name)
....
215.        strcpy(input_file_name, argv[i]);
```

Buffer Overflow StrcpyStrcat\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=44
Status	New

The size of the buffer used by parse_command_line in input_file_name, at line 140 of nmap/train.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that parse_command_line passes to input_file_name, at line 140 of nmap/train.c, to overwrite the target buffer.

	Source	Destination
File	nmap/train.c	nmap/train.c

Line	140	215
Object	input_file_name	input_file_name

Code Snippet

File Name	nmap/train.c
Method	void parse_command_line(int argc, char **argv, char *input_file_name, char *model_file_name)

```
....
140.  void parse_command_line(int argc, char **argv, char
*input_file_name, char *model_file_name)
....
215.        strcpy(input_file_name, argv[i]);
```

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=1050061&projectid=50051&pathid=189
Status	New

The dangerous function, memcpy, was found in use at line 134 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	169	169
Object	memcpy	memcpy

Code Snippet

File Name	nmap/knownhost.c
Method	knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
169.          memcpy(entry->name, host, hostlen + 1);
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=500

	51&pathid=190
Status	New

The dangerous function, memcpy, was found in use at line 134 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	203	203
Object	memcpy	memcpy

Code Snippet
File Name nmap/knownhost.c
Method knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
203.         memcpy(entry->key, key, keylen + 1);
```

Dangerous Functions\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=191
Status	New

The dangerous function, memcpy, was found in use at line 134 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	228	228
Object	memcpy	memcpy

Code Snippet
File Name nmap/knownhost.c
Method knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
228.         memcpy(entry->key_type_name, key_type_name, key_type_len);
```

Dangerous Functions\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=192
Status	New

The dangerous function, memcpy, was found in use at line 134 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	240	240
Object	memcpy	memcpy

Code Snippet

File Name nmap/knownhost.c

Method knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
240.            memcpy(entry->comment, comment, commentlen + 1);
```

Dangerous Functions\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=193
Status	New

The dangerous function, memcpy, was found in use at line 615 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	649	649
Object	memcpy	memcpy

Code Snippet

File Name nmap/knownhost.c

Method static int oldstyle_hostline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
649.                memcpy(hostbuf, name, namelen);
```

Dangerous Functions\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=194
Status	New

The dangerous function, memcpy, was found in use at line 672 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	698	698
Object	memcpy	memcpy

Code Snippet
File Name nmap/knownhost.c
Method static int hashed_hostline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
698.            memcpy(saltbuf, salt, saltlen);
```

Dangerous Functions\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=195
Status	New

The dangerous function, memcpy, was found in use at line 672 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	715	715
Object	memcpy	memcpy

Code Snippet
File Name nmap/knownhost.c
Method static int hashed_hostline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
715.            memcpy(hostbuf, host, hostlen);
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=196
Status	New

The dangerous function, memcpy, was found in use at line 16 in nmap/linear.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp

Line	19	19
Object	memcpy	memcpy

Code Snippet

File Name nmap/linear.cpp

Method template <class S, class T> static inline void clone(T*& dst, S* src, int n)

```
....
19.    memcpy((void *)dst,(void *)src,sizeof(T)*n);
```

Dangerous Functions\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=197
Status	New

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

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

Code Snippet

File Name nmap/lobject.c

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

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

Dangerous Functions\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=198
Status	New

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

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

Code Snippet

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

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

Dangerous Functions\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=199
Status	New

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

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

Code Snippet

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

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

Dangerous Functions\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=200
Status	New

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

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

Code Snippet

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

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

Dangerous Functions\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=201
Status	New

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

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

Code Snippet

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

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

Dangerous Functions\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=202
Status	New

The dangerous function, memcpy, was found in use at line 1479 in nmap/optimize.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	1511	1511
Object	memcpy	memcpy

Code Snippet

File Name	nmap/optimize.c
Method	opt_blk(opt_state_t *opt_state, struct block *b, int do_stmts)

```
....
1511.           memcpy((char *)b->val, (char *)p->pred->val, sizeof(b-
>val));
```

Dangerous Functions\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=203
Status	New

The dangerous function, memcpy, was found in use at line 2945 in nmap/optimize.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2971	2971
Object	memcpy	memcpy

Code Snippet
File Name nmap/optimize.c
Method install_bpf_program(pcap_t *p, struct bpf_program *fp)

```
....
2971.       memcpy(p->fcode.bf_insns, fp->bf_insns, prog_size);
```

Dangerous Functions\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=204
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2789	2789
Object	memcpy	memcpy

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2789.          memcpy(pbits, cbits + posix_class_maps[posix_class],
```

Dangerous Functions\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=205
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	3408	3408
Object	memcpy	memcpy

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
3408.          memcpy(code, classbits, 32);
```

Dangerous Functions\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=206
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	3433	3433
Object	memcpy	memcpy

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
3433.        memcpy(code, classbits, 32);
```

Dangerous Functions\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=207
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	3526	3526
Object	memcpy	memcpy

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
3526.          memcpy(utf8_char, lastchar, c); /* Save the char */
```

Dangerous Functions\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=208
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	3675	3675
Object	memcpy	memcpy

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
3675.                memcpy(code, utf8_char, c & 7);
```

Dangerous Functions\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=209
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	3700	3700
Object	memcpy	memcpy

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
3700.                memcpy(code, utf8_char, c & 7);
```

Dangerous Functions\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=210
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	3730	3730
Object	memcpy	memcpy

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
3730.         memcpy(code, utf8_char, c & 7);
```

Dangerous Functions\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=211
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	3917	3917
Object	memcpy	memcpy

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
3917.               memcpy(code, previous, len);
```

Dangerous Functions\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=212
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	3983	3983
Object	memcpy	memcpy

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
3983.           memcpy(code, previous, len);
```

Dangerous Functions\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=213
Status	New

The dangerous function, memcpy, was found in use at line 2350 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	4552	4552
Object	memcpy	memcpy

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
4552.             memcpy(slot + 2, name, namelen);
```

Dangerous Functions\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=214
Status	New

The dangerous function, sprintf, was found in use at line 140 in nmap/train.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	226	226
Object	sprintf	sprintf

Code Snippet

File Name	nmap/train.c
Method	void parse_command_line(int argc, char **argv, char *input_file_name, char *model_file_name)

```
....
226.                sprintf(model_file_name,"%s.model",p);
```

Dangerous Functions\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=215
Status	New

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

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

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

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

Dangerous Functions\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=216
Status	New

The dangerous function, strcpy, was found in use at line 140 in nmap/train.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	215	215
Object	strcpy	strcpy

Code Snippet
File Name nmap/train.c
Method void parse_command_line(int argc, char **argv, char *input_file_name, char *model_file_name)

```
....
215.        strcpy(input_file_name, argv[i]);
```

Dangerous Functions\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=217
Status	New

The dangerous function, strcpy, was found in use at line 140 in nmap/train.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	218	218
Object	strcpy	strcpy

Code Snippet

File Name	nmap/train.c
Method	void parse_command_line(int argc, char **argv, char *input_file_name, char *model_file_name)

```
....
218.                strcpy(model_file_name,argv[i+1]);
```

Dangerous Functions\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=218
Status	New

The dangerous function, strlen, was found in use at line 134 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	142	142
Object	strlen	strlen

Code Snippet

File Name	nmap/knownhost.c
Method	knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
142.        size_t hostlen = strlen(host);
```

Dangerous Functions\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=219
Status	New

The dangerous function, strlen, was found in use at line 134 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	181	181
Object	strlen	strlen

Code Snippet
File Name nmap/knownhost.c
Method knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
181.                                      salt, strlen(salt));
```

Dangerous Functions\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=220
Status	New

The dangerous function, strlen, was found in use at line 134 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	196	196
Object	strlen	strlen

Code Snippet
File Name nmap/knownhost.c
Method knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
196.                keylen = strlen(key);
```

Dangerous Functions\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=221
Status	New

The dangerous function, strlen, was found in use at line 350 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	433	433
Object	strlen	strlen

Code Snippet
File Name nmap/knownhost.c
Method knownhost_check(LIBSSH2_KNOWNHOSTS *hosts,

```
....
433.                                              strlen(host));
```

Dangerous Functions\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=222
Status	New

The dangerous function, strlen, was found in use at line 953 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	969	969
Object	strlen	strlen

Code Snippet
File Name nmap/knownhost.c
Method libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
969.            if(libssh2_knownhost_readline(hosts, buf, strlen(buf),
type)) {
```

Dangerous Functions\Path 35:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=223
Status	New

The dangerous function, strlen, was found in use at line 997 in nmap/knownhost.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	1078	1078
Object	strlen	strlen

Code Snippet
File Name nmap/knownhost.c
Method knownhost_writeline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1078.      required_size = strlen(node->key);
```

Dangerous Functions\Path 36:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=224
Status	New

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

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

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

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

Dangerous Functions\Path 37:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=225
Status	New

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

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

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

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

Dangerous Functions\Path 38:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=226
Status	New

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

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

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

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

Dangerous Functions\Path 39:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=227
Status	New

The dangerous function, strlen, was found in use at line 5801 in nmap/pcre_compile.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	5995	5995
Object	strlen	strlen

Code Snippet

File Name	nmap/pcre_compile.c
Method	pcre_compile2(const char *pattern, int options, int *errorcodeptr,

```
....
5995.  cd->end_pattern = (const uschar *)(pattern + strlen(pattern));
```

Dangerous Functions\Path 40:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=228
Status	New

The dangerous function, strlen, was found in use at line 55 in nmap/train.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	66	66
Object	strlen	strlen

Code Snippet

File Name	nmap/train.c
Method	static char* readline(FILE *input)

```
....
66.         len = (int) strlen(line);
```

Dangerous Functions\Path 41:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=229
Status	New

The dangerous function, strtok, was found in use at line 186 in nmap/ncat_main.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	204	204
Object	strtok	strtok

Code Snippet
File Name   nmap/ncat_main.c
Method   static void host_list_to_set(struct addrset *set, struct host_list_node *list)

```
....
204.                while ((spec = strtok(commalist, ",")) != NULL) {
```

Dangerous Functions\Path 42:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=230
Status	New

The dangerous function, strtok, was found in use at line 218 in nmap/ncat_main.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	356	356
Object	strtok	strtok

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

```
....
356.                while (o.numsrcrtes < 8 && (a = strtok(from, ",")))
```

Dangerous Functions\Path 43:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=231
Status	New

The dangerous function, strtok, was found in use at line 218 in nmap/ncat_main.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	370	370
Object	strtok	strtok

Code Snippet

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

```
....
370.                if (strtok(from, ","))
```

Dangerous Functions\Path 44:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=232
Status	New

The dangerous function, strtok, was found in use at line 241 in nmap/train.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	261	261
Object	strtok	strtok

Code Snippet

File Name	nmap/train.c
Method	void read_problem(const char *filename)

```
....
261.                char *p = strtok(line," \t"); // label
```

Dangerous Functions\Path 45:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=233
Status	New

The dangerous function, strtok, was found in use at line 241 in nmap/train.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	266	266
Object	strtok	strtok

Code Snippet
File Name nmap/train.c
Method void read_problem(const char *filename)

```
....
266.                      p = strtok(NULL," \t");
```

Dangerous Functions\Path 46:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=234
Status	New

The dangerous function, strtok, was found in use at line 241 in nmap/train.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	289	289
Object	strtok	strtok

Code Snippet
File Name nmap/train.c
Method void read_problem(const char *filename)

```
....
289.                label = strtok(line," \t\n");
```

Dangerous Functions\Path 47:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=235
Status	New

The dangerous function, strtok, was found in use at line 241 in nmap/train.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	299	299
Object	strtok	strtok

Code Snippet
File Name nmap/train.c
Method void read_problem(const char *filename)

```
....
299.                        idx = strtok(NULL,":");
```

Dangerous Functions\Path 48:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=236
Status	New

The dangerous function, strtok, was found in use at line 241 in nmap/train.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	300	300
Object	strtok	strtok

Code Snippet
File Name nmap/train.c
Method void read_problem(const char *filename)

```
....
300.                    val = strtok(NULL," \t");
```

Dangerous Functions\Path 49:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=237
Status	New

The dangerous function, vsprintf, was found in use at line 33 in nmap/linear.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	38	38
Object	vsprintf	vsprintf

Code Snippet

File Name	nmap/linear.cpp
Method	static void info(const char *fmt,...)

```
....
38.    vsprintf(buf,fmt,ap);
```

Dangerous Functions\Path 50:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=238
Status	New

The dangerous function, realloc, was found in use at line 1768 in nmap/linear.cpp file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1796	1796
Object	realloc	realloc

Code Snippet

File Name	nmap/linear.cpp
Method	static void group_classes(const problem *prob, int *nr_class_ret, int **label_ret, int **start_ret, int **count_ret, int *perm)

```
....
1796.                          label = (int
*)realloc(label,max_nr_class*sizeof(int));
```

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=1050061&projectid=50051&pathid=9
Status	New

The size of the buffer used by opt_blk in ->, at line 1479 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that opt_blk passes to ->, at line 1479 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	1511	1511
Object	->	->

Code Snippet
File Name nmap/optimize.c
Method opt_blk(opt_state_t *opt_state, struct block *b, int do_stmts)

```
....
1511.             memcpy((char *)b->val, (char *)p->pred->val, sizeof(b-
>val));
```

Buffer Overflow boundcpy WrongSizeParam\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=10
Status	New

The size of the buffer used by libssh2_knownhost_del in libssh2_knownhost, at line 564 of nmap/knownhost.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that libssh2_knownhost_del passes to libssh2_knownhost, at line 564 of nmap/knownhost.c, to overwrite the target buffer.

	Source	Destination

File	nmap/knownhost.c	nmap/knownhost.c
Line	582	582
Object	libssh2_knownhost	libssh2_knownhost

Code Snippet
File Name nmap/knownhost.c
Method libssh2_knownhost_del(LIBSSH2_KNOWNHOSTS *hosts,

```
....
582.      memset(entry, 0, sizeof(struct libssh2_knownhost));
```

Buffer Overflow boundcpy WrongSizeParam\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=11
Status	New

The size of the buffer used by main in Namespace1788516817, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to Namespace1788516817, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	715	715
Object	Namespace1788516817	Namespace1788516817

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

```
....
715.      memset(&srcaddr.storage, 0, sizeof(srcaddr.storage));
```

Buffer Overflow boundcpy WrongSizeParam\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=12
Status	New

The size of the buffer used by main in sockaddr_vm, at line 218 of nmap/ncat_main.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that main passes to sockaddr_vm, at line 218 of nmap/ncat_main.c, to overwrite the target buffer.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c

Line	863	863
Object	sockaddr_vm	sockaddr_vm

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

```
....
863.          memset(&targetaddrs->addr.storage, 0, sizeof(struct
sockaddr_vm));
```

Buffer Overflow boundcpy WrongSizeParam\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=13
Status	New

The size of the buffer used by init_val in opt_state, at line 709 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that init_val passes to opt_state, at line 709 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	714	714
Object	opt_state	opt_state

Code Snippet
File Name nmap/optimize.c
Method init_val(opt_state_t *opt_state)

```
....
714.        memset((char *)opt_state->hashtbl, 0, sizeof opt_state-
>hashtbl);
```

Buffer Overflow boundcpy WrongSizeParam\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=14
Status	New

The size of the buffer used by opt_deadstores in last, at line 1455 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that opt_deadstores passes to last, at line 1455 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c

Line	1461	1461
Object	last	last

Code Snippet
File Name nmap/optimize.c
Method opt_deadstores(opt_state_t *opt_state, register struct block *b)

```
....
1461.        memset((char *)last, 0, sizeof last);
```

Buffer Overflow boundcpy WrongSizeParam\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=15
Status	New

The size of the buffer used by opt_blk in ->, at line 1479 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that opt_blk passes to ->, at line 1479 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	1503	1503
Object	->	->

Code Snippet
File Name nmap/optimize.c
Method opt_blk(opt_state_t *opt_state, struct block *b, int do_stmts)

```
....
1503.            memset((char *)b->val, 0, sizeof(b->val));
```

Buffer Overflow boundcpy WrongSizeParam\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=16
Status	New

The size of the buffer used by clone in n, at line 16 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that clone passes to n, at line 16 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	19	19

Object	n	n

Code Snippet
File Name nmap/linear.cpp
Method template <class S, class T> static inline void clone(T*& dst, S* src, int n)

```
....
19.    memcpy((void *)dst,(void *)src,sizeof(T)*n);
```

Buffer Overflow boundcpy WrongSizeParam\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=17
Status	New

The size of the buffer used by clone in T, at line 16 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that clone passes to T, at line 16 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	19	19
Object	T	T

Code Snippet
File Name nmap/linear.cpp
Method template <class S, class T> static inline void clone(T*& dst, S* src, int n)

```
....
19.    memcpy((void *)dst,(void *)src,sizeof(T)*n);
```

Buffer Overflow boundcpy WrongSizeParam\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=18
Status	New

The size of the buffer used by luaO_chunkid in srclen, at line 557 of nmap/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to srclen, at line 557 of nmap/lobject.c, to overwrite the target buffer.

	Source	Destination
File	nmap/lobject.c	nmap/lobject.c
Line	561	561
Object	srclen	srclen

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=19
Status	New

The size of the buffer used by luaO_chunkid in char, at line 557 of nmap/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to char, at line 557 of nmap/lobject.c, to overwrite the target buffer.

	Source	Destination
File	nmap/lobject.c	nmap/lobject.c
Line	561	561
Object	char	char

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=20
Status	New

The size of the buffer used by luaO_chunkid in srclen, at line 557 of nmap/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to srclen, at line 557 of nmap/lobject.c, to overwrite the target buffer.

	Source	Destination
File	nmap/lobject.c	nmap/lobject.c
Line	569	569
Object	srclen	srclen

Code Snippet	
File Name	nmap/lobject.c

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=21
Status	New

The size of the buffer used by luaO_chunkid in char, at line 557 of nmap/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to char, at line 557 of nmap/lobject.c, to overwrite the target buffer.

	Source	Destination
File	nmap/lobject.c	nmap/lobject.c
Line	569	569
Object	char	char

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=22
Status	New

The size of the buffer used by luaO_chunkid in bufflen, at line 557 of nmap/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to bufflen, at line 557 of nmap/lobject.c, to overwrite the target buffer.

	Source	Destination
File	nmap/lobject.c	nmap/lobject.c
Line	573	573
Object	bufflen	bufflen

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=23
Status	New

The size of the buffer used by luaO_chunkid in char, at line 557 of nmap/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to char, at line 557 of nmap/lobject.c, to overwrite the target buffer.

	Source	Destination
File	nmap/lobject.c	nmap/lobject.c
Line	573	573
Object	char	char

Code Snippet

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=24
Status	New

The size of the buffer used by luaO_chunkid in char, at line 557 of nmap/lobject.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that luaO_chunkid passes to char, at line 557 of nmap/lobject.c, to overwrite the target buffer.

	Source	Destination
File	nmap/lobject.c	nmap/lobject.c
Line	589	589
Object	char	char

Code Snippet

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

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

Buffer Overflow boundcpy WrongSizeParam\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=25
Status	New

The size of the buffer used by compile_branch in uschar, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to uschar, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2790	2790
Object	uschar	uschar

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2790.              32 * sizeof(uschar));
```

Buffer Overflow boundcpy WrongSizeParam\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=26
Status	New

The size of the buffer used by find_levels in opt_state, at line 407 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that find_levels passes to opt_state, at line 407 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	409	409
Object	opt_state	opt_state

Code Snippet
File Name nmap/optimize.c
Method find_levels(opt_state_t *opt_state, struct icode *ic)

```
....
409.        memset((char *)opt_state->levels, 0, opt_state->n_blocks * 
sizeof(*opt_state->levels));
```

Buffer Overflow boundcpy WrongSizeParam\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=27
Status	New

The size of the buffer used by find_levels in opt_state, at line 407 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that find_levels passes to opt_state, at line 407 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	409	409
Object	opt_state	opt_state

Code Snippet
File Name nmap/optimize.c
Method find_levels(opt_state_t *opt_state, struct icode *ic)

```
....
409.        memset((char *)opt_state->levels, 0, opt_state->n_blocks * 
sizeof(*opt_state->levels));
```

Buffer Overflow boundcpy WrongSizeParam\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=28
Status	New

The size of the buffer used by find_edom in opt_state, at line 471 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that find_edom passes to opt_state, at line 471 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	488	488
Object	opt_state	opt_state

Code Snippet
File Name nmap/optimize.c
Method find_edom(opt_state_t *opt_state, struct block *root)

```
....
488.         memset(root->et.edom, 0, opt_state->edgewords *
sizeof(*(uset)0));
```

Buffer Overflow boundcpy WrongSizeParam\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=29
Status	New

The size of the buffer used by find_edom in opt_state, at line 471 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that find_edom passes to opt_state, at line 471 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	489	489
Object	opt_state	opt_state

Code Snippet

File Name nmap/optimize.c

Method find_edom(opt_state_t *opt_state, struct block *root)

```
....
489.         memset(root->ef.edom, 0, opt_state->edgewords *
sizeof(*(uset)0));
```

Buffer Overflow boundcpy WrongSizeParam\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=30
Status	New

The size of the buffer used by find_closure in opt_state, at line 506 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that find_closure passes to opt_state, at line 506 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	515	515
Object	opt_state	opt_state

Code Snippet

File Name nmap/optimize.c

Method find_closure(opt_state_t *opt_state, struct block *root)

```
....
515.              opt_state->n_blocks * opt_state->nodewords *
sizeof(*opt_state->all_closure_sets));
```

Buffer Overflow boundcpy WrongSizeParam\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=31
Status	New

The size of the buffer used by find_closure in opt_state, at line 506 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that find_closure passes to opt_state, at line 506 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	515	515
Object	opt_state	opt_state

Code Snippet
File Name nmap/optimize.c
Method find_closure(opt_state_t *opt_state, struct block *root)

```
....
515.              opt_state->n_blocks * opt_state->nodewords *
sizeof(*opt_state->all_closure_sets));
```

Buffer Overflow boundcpy WrongSizeParam\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=32
Status	New

The size of the buffer used by find_closure in opt_state, at line 506 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that find_closure passes to opt_state, at line 506 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	515	515
Object	opt_state	opt_state

Code Snippet
File Name nmap/optimize.c
Method find_closure(opt_state_t *opt_state, struct block *root)

```
....
515.                opt_state->n_blocks * opt_state->nodewords *
sizeof(*opt_state->all_closure_sets));
```

Buffer Overflow boundcpy WrongSizeParam\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=33
Status	New

The size of the buffer used by init_val in opt_state, at line 709 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that init_val passes to opt_state, at line 709 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	713	713
Object	opt_state	opt_state

Code Snippet
File Name nmap/optimize.c
Method init_val(opt_state_t *opt_state)

```
....
713.         memset((char *)opt_state->vmap, 0, opt_state->maxval *
sizeof(*opt_state->vmap));
```

Buffer Overflow boundcpy WrongSizeParam\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=34
Status	New

The size of the buffer used by init_val in opt_state, at line 709 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that init_val passes to opt_state, at line 709 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	713	713
Object	opt_state	opt_state

Code Snippet
File Name nmap/optimize.c
Method init_val(opt_state_t *opt_state)

```
....
713.        memset((char *)opt_state->vmap, 0, opt_state->maxval *
sizeof(*opt_state->vmap));
```

Buffer Overflow boundcpy WrongSizeParam\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=35
Status	New

The size of the buffer used by icode_to_fcode in n, at line 2876 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that icode_to_fcode passes to n, at line 2876 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2904	2904
Object	n	n

Code Snippet
File Name nmap/optimize.c
Method icode_to_fcode(struct icode *ic, struct block *root, u_int *lenp,

```
....
2904.         memset((char *)fp, 0, sizeof(*fp) * n);
```

Buffer Overflow boundcpy WrongSizeParam\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=36
Status	New

The size of the buffer used by icode_to_fcode in fp, at line 2876 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that icode_to_fcode passes to fp, at line 2876 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2904	2904
Object	fp	fp

Code Snippet
File Name nmap/optimize.c
Method icode_to_fcode(struct icode *ic, struct block *root, u_int *lenp,

```
....
2904.           memset((char *)fp, 0, sizeof(*fp) * n);
```

Buffer Overflow boundcpy WrongSizeParam\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=37
Status	New

The size of the buffer used by compile_branch in uschar, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to uschar, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2689	2689
Object	uschar	uschar

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2689.       memset(classbits, 0, 32 * sizeof(uschar));
```

Buffer Overflow boundcpy WrongSizeParam\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=38
Status	New

The size of the buffer used by knownhost_add in key_type_len, at line 134 of nmap/knownhost.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that knownhost_add passes to key_type_len, at line 134 of nmap/knownhost.c, to overwrite the target buffer.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	228	228
Object	key_type_len	key_type_len

Code Snippet
File Name nmap/knownhost.c
Method knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
228.           memcpy(entry->key_type_name, key_type_name, key_type_len);
```

Buffer Overflow boundcpy WrongSizeParam\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=39
Status	New

The size of the buffer used by oldstyle_hostline in namelen, at line 615 of nmap/knownhost.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that oldstyle_hostline passes to namelen, at line 615 of nmap/knownhost.c, to overwrite the target buffer.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	649	649
Object	namelen	namelen

Code Snippet
File Name nmap/knownhost.c
Method static int oldstyle_hostline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
649.             memcpy(hostbuf, name, namelen);
```

Buffer Overflow boundcpy WrongSizeParam\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=40
Status	New

The size of the buffer used by install_bpf_program in prog_size, at line 2945 of nmap/optimize.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that install_bpf_program passes to prog_size, at line 2945 of nmap/optimize.c, to overwrite the target buffer.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2971	2971
Object	prog_size	prog_size

Code Snippet
File Name nmap/optimize.c
Method install_bpf_program(pcap_t *p, struct bpf_program *fp)

```
....
2971.       memcpy(p->fcode.bf_insns, fp->bf_insns, prog_size);
```

Buffer Overflow boundcpy WrongSizeParam\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=41
Status	New

The size of the buffer used by compile_branch in namelen, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to namelen, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	4552	4552
Object	namelen	namelen

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
4552.             memcpy(slot + 2, name, namelen);
```

Buffer Overflow boundcpy WrongSizeParam\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=42
Status	New

The size of the buffer used by compile_branch in namelen, at line 2350 of nmap/pcre_compile.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that compile_branch passes to namelen, at line 2350 of nmap/pcre_compile.c, to overwrite the target buffer.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	4529	4529
Object	namelen	namelen

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
4529.                int crc = memcmp(name, slot+2, namelen);
```

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=1050061&projectid=50051&pathid=264
Status	New

The variable declared in label at nmap/linear.cpp in line 1896 is not initialized when it is used by label at nmap/linear.cpp in line 1896.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1912	1923
Object	label	label

Code Snippet
File Name nmap/linear.cpp
Method model* train(const problem *prob, const parameter *param)

```
....
1912.        int *label = NULL;
....
1923.              model_->label[i] = label[i];
```

Use of Zero Initialized Pointer\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=265
Status	New

The variable declared in start at nmap/linear.cpp in line 1896 is not initialized when it is used by start at nmap/linear.cpp in line 1896.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp

Line	1913	1971
Object	start	start

Code Snippet
File Name nmap/linear.cpp
Method model* train(const problem *prob, const parameter *param)

```
....
1913.        int *start = NULL;
....
1971.                    int e0 = start[0]+count[0];
```

Use of Zero Initialized Pointer\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=266
Status	New

The variable declared in start at nmap/linear.cpp in line 1896 is not initialized when it is used by start at nmap/linear.cpp in line 1896.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1913	1986
Object	start	start

Code Snippet
File Name nmap/linear.cpp
Method model* train(const problem *prob, const parameter *param)

```
....
1913.        int *start = NULL;
....
1986.                              int si = start[i];
```

Use of Zero Initialized Pointer\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=267
Status	New

The variable declared in count at nmap/linear.cpp in line 1896 is not initialized when it is used by count at nmap/linear.cpp in line 1896.

	Source	Destination

File	nmap/linear.cpp	nmap/linear.cpp
Line	1914	1971
Object	count	count

Code Snippet
File Name nmap/linear.cpp
Method model* train(const problem *prob, const parameter *param)

```
....
1914.        int *count = NULL;
....
1971.                        int e0 = start[0]+count[0];
```

Use of Zero Initialized Pointer\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=268
Status	New

The variable declared in count at nmap/linear.cpp in line 1896 is not initialized when it is used by count at nmap/linear.cpp in line 1896.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1914	1987
Object	count	count

Code Snippet
File Name nmap/linear.cpp
Method model* train(const problem *prob, const parameter *param)

```
....
1914.        int *count = NULL;
....
1987.                            int ei = si+count[i];
```

Use of Zero Initialized Pointer\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=269
Status	New

The variable declared in save_hwm at nmap/pcre_compile.c in line 2350 is not initialized when it is used by save_hwm at nmap/pcre_compile.c in line 2350.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2375	3989
Object	save_hwm	save_hwm

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2375.   uschar *save_hwm = NULL;
....
3989.             save_hwm = this_hwm;
```

Use of Zero Initialized Pointer\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=270
Status	New

The variable declared in save_hwm at nmap/pcre_compile.c in line 2350 is not initialized when it is used by previous at nmap/pcre_compile.c in line 2350.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2375	3415
Object	save_hwm	previous

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2375.   uschar *save_hwm = NULL;
....
3415.         PUT(previous, 1, code - previous);
```

Use of Zero Initialized Pointer\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=271
Status	New

The variable declared in save_hwm at nmap/pcre_compile.c in line 2350 is not initialized when it is used by bralink at nmap/pcre_compile.c in line 2350.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2375	3979
Object	save_hwm	bralink

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2375.   uschar *save_hwm = NULL;
....
3979.               bralink = code;
```

Use of Zero Initialized Pointer\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=272
Status	New

The variable declared in save_hwm at nmap/pcre_compile.c in line 2350 is not initialized when it is used by save_hwm at nmap/pcre_compile.c in line 2350.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2375	3923
Object	save_hwm	save_hwm

Code Snippet

File Name	nmap/pcre_compile.c
Method	compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2375.   uschar *save_hwm = NULL;
....
3923.                 save_hwm = this_hwm;
```

Use of Zero Initialized Pointer\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=273
Status	New

The variable declared in save_hwm at nmap/pcre_compile.c in line 2350 is not initialized when it is used by previous at nmap/pcre_compile.c in line 2350.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2375	4896
Object	save_hwm	previous

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2375.  uschar *save_hwm = NULL;
....
4896.      previous = (bravalue >= OP_ONCE)? code : NULL;
```

Use of Zero Initialized Pointer\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=274
Status	New

The variable declared in save_hwm at nmap/pcre_compile.c in line 2350 is not initialized when it is used by save_hwm at nmap/pcre_compile.c in line 2350.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	2375	4120
Object	save_hwm	save_hwm

Code Snippet
File Name nmap/pcre_compile.c
Method compile_branch(int *optionsptr, uschar **codeptr, const uschar **ptrptr,

```
....
2375.  uschar *save_hwm = NULL;
....
4120.      save_hwm = cd->hwm;
```

Use of Zero Initialized Pointer\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=275
Status	New

The variable declared in nullpad at nmap/pcre_compile.c in line 5801 is not initialized when it is used by name_table at nmap/pcre_compile.c in line 5801.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	6055	6067
Object	nullpad	name_table

Code Snippet

File Name	nmap/pcre_compile.c
Method	pcre_compile2(const char *pattern, int options, int *errorcodeptr,

```
....
6055.   re->nullpad = NULL;
....
6067.   cd->name_table = (uschar *)re + re->name_table_offset;
```

Use of Zero Initialized Pointer\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=276
Status	New

The variable declared in weight_label at nmap/train.c in line 140 is not initialized when it is used by weight_label at nmap/train.c in line 140.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	150	181
Object	weight_label	weight_label

Code Snippet

File Name	nmap/train.c
Method	void parse_command_line(int argc, char **argv, char *input_file_name, char *model_file_name)

```
....
150.        param.weight_label = NULL;
....
181.                          param.weight_label = (int *)
realloc(param.weight_label,sizeof(int)*param.nr_weight);
```

Use of Zero Initialized Pointer\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=277
Status	New

The variable declared in weight at nmap/train.c in line 140 is not initialized when it is used by weight at nmap/train.c in line 140.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	151	182
Object	weight	weight

Code Snippet
File Name nmap/train.c
Method void parse_command_line(int argc, char **argv, char *input_file_name, char *model_file_name)

```
....
151.        param.weight = NULL;
....
182.                                param.weight = (double *)
realloc(param.weight,sizeof(double)*param.nr_weight);
```

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=1050061&projectid=50051&pathid=248
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	18	18
Object	dst	dst

Code Snippet
File Name nmap/linear.cpp
Method template <class S, class T> static inline void clone(T*& dst, S* src, int n)

```
....
18.    dst = new T[n];
```

Memory Leak\Path 2:

Severity	Medium

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=249
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1805	1805
Object	start	start

Code Snippet

File Name	nmap/linear.cpp
Method	static void group_classes(const problem *prob, int *nr_class_ret, int **label_ret, int **start_ret, int **count_ret, int *perm)

```
....
1805.        int *start = Malloc(int,nr_class);
```

Memory Leak\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=250
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2291	2291
Object	w	w

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2291.        model_->w=Malloc(double, w_size*nr_w);
```

Memory Leak\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=251
Status	New

	Source	Destination

File	nmap/optimize.c	nmap/optimize.c
Line	2533	2533
Object	blocks	blocks

Code Snippet
File Name nmap/optimize.c
Method opt_init(opt_state_t *opt_state, struct icode *ic)

```
....
2533.        opt_state->blocks = (struct block **)calloc(n,
sizeof(*opt_state->blocks));
```

Memory Leak\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=252
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2553	2553
Object	edges	edges

Code Snippet
File Name nmap/optimize.c
Method opt_init(opt_state_t *opt_state, struct icode *ic)

```
....
2553.        opt_state->edges = (struct edge **)calloc(opt_state-
>n_edges, sizeof(*opt_state->edges));
```

Memory Leak\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=253
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2561	2561
Object	levels	levels

Code Snippet

File Name	nmap/optimize.c
Method	opt_init(opt_state_t *opt_state, struct icode *ic)

```
....
2561.       opt_state->levels = (struct block **)calloc(opt_state-
>n_blocks, sizeof(*opt_state->levels));
```

Memory Leak\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=254
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2621	2621
Object	space	space

Code Snippet

File Name	nmap/optimize.c
Method	opt_init(opt_state_t *opt_state, struct icode *ic)

```
....
2621.       opt_state->space = (bpf_u_int32 *)malloc(block_memsize +
edge_memsize);
```

Memory Leak\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=255
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2660	2660
Object	vmap	vmap

Code Snippet

File Name	nmap/optimize.c
Method	opt_init(opt_state_t *opt_state, struct icode *ic)

```
....
2660.       opt_state->vmap = (struct vmapinfo *)calloc(opt_state-
>maxval, sizeof(*opt_state->vmap));
```

Memory Leak\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=256
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2664	2664
Object	vnode_base	vnode_base

Code Snippet
File Name nmap/optimize.c
Method opt_init(opt_state_t *opt_state, struct icode *ic)

```
....
2664.       opt_state->vnode_base = (struct valnode *)calloc(opt_state->maxval, sizeof(*opt_state->vnode_base));
```

Memory Leak\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=257
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2965	2965
Object	bf_insns	bf_insns

Code Snippet
File Name nmap/optimize.c
Method install_bpf_program(pcap_t *p, struct bpf_program *fp)

```
....
2965.       p->fcode.bf_insns = (struct bpf_insn *)malloc(prog_size);
```

Use of Uninitialized Variable

Query Path:
CPP\Cx\CPP Medium Threat\Use of Uninitialized Variable Version:0

Categories

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

Description

Use of Uninitialized Variable\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=258
Status	New

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	5806	5959
Object	newline	newline

Code Snippet

File Name nmap/pcre_compile.c

Method pcre_compile2(const char *pattern, int options, int *errorcodeptr,

```
....
5806.   int firstbyte, reqbyte, newline;
....
5959.       cd->nl[1] = newline & 255;
```

Use of Uninitialized Variable\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=259
Status	New

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	5806	5958
Object	newline	newline

Code Snippet

File Name nmap/pcre_compile.c

Method pcre_compile2(const char *pattern, int options, int *errorcodeptr,

```
....
5806.   int firstbyte, reqbyte, newline;
....
5958.       cd->nl[0] = (newline >> 8) & 255;
```

Use of Uninitialized Variable\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=500

	51&pathid=260
Status	New

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	5806	5964
Object	newline	newline

Code Snippet
File Name nmap/pcre_compile.c
Method pcre_compile2(const char *pattern, int options, int *errorcodeptr,

```
....
5806.  int firstbyte, reqbyte, newline;
....
5964.      cd->nl[0] = newline;
```

Use of Uninitialized Variable\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=261
Status	New

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	5806	5955
Object	newline	newline

Code Snippet
File Name nmap/pcre_compile.c
Method pcre_compile2(const char *pattern, int options, int *errorcodeptr,

```
....
5806.  int firstbyte, reqbyte, newline;
....
5955.    if (newline > 255)
```

Use of Uninitialized Variable\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=262
Status	New

	Source	Destination

File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	5806	5948
Object	newline	newline

Code Snippet
File Name nmap/pcre_compile.c
Method pcre_compile2(const char *pattern, int options, int *errorcodeptr,

```
....
5806.   int firstbyte, reqbyte, newline;
....
5948.   else if (newline < 0)
```

Use of Uninitialized Variable\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=263
Status	New

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	5806	5944
Object	newline	newline

Code Snippet
File Name nmap/pcre_compile.c
Method pcre_compile2(const char *pattern, int options, int *errorcodeptr,

```
....
5806.   int firstbyte, reqbyte, newline;
....
5944.   if (newline == -2)
```

Wrong Size t Allocation

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

Description

Wrong Size t Allocation\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=107
Status	New

The function prog_size in nmap/optimize.c at line 2945 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2965	2965
Object	prog_size	prog_size

Code Snippet
File Name nmap/optimize.c
Method install_bpf_program(pcap_t *p, struct bpf_program *fp)

```
....
2965.        p->fcode.bf_insns = (struct bpf_insn *)malloc(prog_size);
```

Wrong Size t Allocation\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=108
Status	New

The function block_memsize in nmap/optimize.c at line 2520 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2621	2621
Object	block_memsize	block_memsize

Code Snippet
File Name nmap/optimize.c
Method opt_init(opt_state_t *opt_state, struct icode *ic)

```
....
2621.        opt_state->space = (bpf_u_int32 *)malloc(block_memsize + 
edge_memsize);
```

Wrong Size t Allocation\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=109
Status	New

The function edge_memsize in nmap/optimize.c at line 2520 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination

File	nmap/optimize.c	nmap/optimize.c
Line	2621	2621
Object	edge_memsize	edge_memsize

Code Snippet
File Name nmap/optimize.c
Method opt_init(opt_state_t *opt_state, struct icode *ic)

```
....
2621.       opt_state->space = (bpf_u_int32 *)malloc(block_memsize +
edge_memsize);
```

Wrong Size t Allocation\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=110
Status	New

The function elements in nmap/train.c at line 241 assigns an incorrectly calculated size to a buffer, resulting in a mismatch between the value being written and the size of the buffer it is being written into.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	280	280
Object	elements	elements

Code Snippet
File Name nmap/train.c
Method void read_problem(const char *filename)

```
....
280.        x_space = Malloc(struct feature_node,elements+prob.l);
```

Stored Buffer Overflow fgets

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

Categories

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

Description

Stored Buffer Overflow fgets\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=281

Status	New

The size of the buffer used by readline in max_line_len, at line 55 of nmap/train.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that readline passes to BinaryExpr, at line 55 of nmap/train.c, to overwrite the target buffer.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	67	59
Object	BinaryExpr	max_line_len

Code Snippet
File Name nmap/train.c
Method static char* readline(FILE *input)

```
....
67.          if(fgets(line+len,max_line_len-len,input) == NULL)
....
59.    if(fgets(line,max_line_len,input) == NULL)
```

Stored Buffer Overflow fgets\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=282
Status	New

The size of the buffer used by readline in max_line_len, at line 55 of nmap/train.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that readline passes to line, at line 55 of nmap/train.c, to overwrite the target buffer.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	59	59
Object	line	max_line_len

Code Snippet
File Name nmap/train.c
Method static char* readline(FILE *input)

```
....
59.    if(fgets(line,max_line_len,input) == NULL)
```

Stored Buffer Overflow fgets\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=283
Status	New

The size of the buffer used by readline in BinaryExpr, at line 55 of nmap/train.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that readline passes to BinaryExpr, at line 55 of nmap/train.c, to overwrite the target buffer.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	67	67
Object	BinaryExpr	BinaryExpr

Code Snippet
File Name nmap/train.c
Method static char* readline(FILE *input)

```
....
67.          if(fgets(line+len,max_line_len-len,input) == NULL)
```

Stored Buffer Overflow fgets\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=284
Status	New

The size of the buffer used by readline in BinaryExpr, at line 55 of nmap/train.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that readline passes to line, at line 55 of nmap/train.c, to overwrite the target buffer.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	59	67
Object	line	BinaryExpr

Code Snippet
File Name nmap/train.c
Method static char* readline(FILE *input)

```
....
59.    if(fgets(line,max_line_len,input) == NULL)
....
67.          if(fgets(line+len,max_line_len-len,input) == NULL)
```

Short Overflow

Query Path:
CPP\Cx\CPP Integer Overflow\Short Overflow Version:1

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

Short Overflow\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=169
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 191 of nmap/blast.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	nmap/blast.c	nmap/blast.c
Line	206	206
Object	AssignExpr	AssignExpr

Code Snippet
File Name nmap/blast.c
Method local int construct(struct huffman *h, const unsigned char *rep, int n)

```
....
206.              length[symbol++] = len;
```

Short Overflow\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=170
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 665 of nmap/puff.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	nmap/puff.c	nmap/puff.c
Line	711	711
Object	AssignExpr	AssignExpr

Code Snippet
File Name nmap/puff.c
Method local int dynamic(struct state *s)

```
....
711.              lengths[index++] = symbol;
```

Short Overflow\Path 3:

Severity	Medium
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=171
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 665 of nmap/puff.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	nmap/puff.c	nmap/puff.c
Line	727	727
Object	AssignExpr	AssignExpr

Code Snippet
File Name nmap/puff.c
Method local int dynamic(struct state *s)

```
....
727.                    lengths[index++] = len;
```

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=1050061&projectid=50051&pathid=278
Status	New

The size of the buffer used by knownhost_add in key_type_len, at line 134 of nmap/knownhost.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that libssh2_knownhost_readfile passes to buf, at line 953 of nmap/knownhost.c, to overwrite the target buffer.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	968	228
Object	buf	key_type_len

Code Snippet
File Name nmap/knownhost.c
Method libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
968.         while(fgets(buf, sizeof(buf), file)) {
```

▼

File Name nmap/knownhost.c

Method knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
228.         memcpy(entry->key_type_name, key_type_name, key_type_len);
```

Stored Buffer Overflow boundcpy\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=279
Status	New

The size of the buffer used by hashed_hostline in hostlen, at line 672 of nmap/knownhost.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that libssh2_knownhost_readfile passes to buf, at line 953 of nmap/knownhost.c, to overwrite the target buffer.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	968	715
Object	buf	hostlen

Code Snippet

File Name nmap/knownhost.c

Method libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
968.         while(fgets(buf, sizeof(buf), file)) {
```

▼

File Name nmap/knownhost.c

Method static int hashed_hostline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
715.         memcpy(hostbuf, host, hostlen);
```

Stored Buffer Overflow boundcpy\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=280
Status	New

The size of the buffer used by hashed_hostline in saltlen, at line 672 of nmap/knownhost.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that libssh2_knownhost_readfile passes to buf, at line 953 of nmap/knownhost.c, to overwrite the target buffer.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	968	698
Object	buf	saltlen

Code Snippet
File Name nmap/knownhost.c
Method libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
968.          while(fgets(buf, sizeof(buf), file)) {
```

File Name nmap/knownhost.c
Method static int hashed_hostline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
698.          memcpy(saltbuf, salt, saltlen);
```

Char Overflow

Query Path:
CPP\Cx\CPP Integer Overflow\Char Overflow Version:1

Categories

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

Description

Char Overflow\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=167
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 772 of nmap/pcre_compile.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	798	798
Object	AssignExpr	AssignExpr

Code Snippet

File Name	nmap/pcre_compile.c
Method	get_ucp(const uschar **ptrptr, BOOL *negptr, int *dptr, int *errorcodeptr)

```
....
798.        name[i] = c;
```

Char Overflow\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=168
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 772 of nmap/pcre_compile.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	808	808
Object	AssignExpr	AssignExpr

Code Snippet

File Name	nmap/pcre_compile.c
Method	get_ucp(const uschar **ptrptr, BOOL *negptr, int *dptr, int *errorcodeptr)

```
....
808.      name[0] = c;
```

Divide By Zero

Query Path:

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

Description

Divide By Zero\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=103
Status	New

The application performs an illegal operation in predict_probability, in nmap/linear.cpp. In line 2121, the program attempts to divide by sum, which might be evaluate to 0 (zero) at time of division. This value could be a hard-coded zero value, or received from external, untrusted input sum in predict_probability of nmap/linear.cpp, at line 2121.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2146	2146

Object	sum	sum

Code Snippet
File Name nmap/linear.cpp
Method int predict_probability(const struct model *model_, const struct feature_node *x, double* prob_estimates)

```
....
2146.                    prob_estimates[i]=prob_estimates[i]/sum;
```

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=1050061&projectid=50051&pathid=285
Status	New

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	968	968
Object	fgets	fgets

Code Snippet
File Name nmap/knownhost.c
Method libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
968.         while(fgets(buf, sizeof(buf), file)) {
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=286
Status	New

	Source	Destination

File	nmap/train.c	nmap/train.c
Line	59	59
Object	fgets	fgets

Code Snippet
File Name nmap/train.c
Method static char* readline(FILE *input)

```
....
59.    if(fgets(line,max_line_len,input) == NULL)
```

Improper Resource Access Authorization\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=287
Status	New

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	67	67
Object	fgets	fgets

Code Snippet
File Name nmap/train.c
Method static char* readline(FILE *input)

```
....
67.         if(fgets(line+len,max_line_len-len,input) == NULL)
```

Improper Resource Access Authorization\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=288
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2224	2224
Object	fscanf	fscanf

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2224.             fscanf(fp,"%80s",cmd);
```

Improper Resource Access Authorization\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=289
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2227	2227
Object	fscanf	fscanf

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2227.                   fscanf(fp,"%80s",cmd);
```

Improper Resource Access Authorization\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=290
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2247
Object	fscanf	fscanf

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2247.                   fscanf(fp,"%d",&nr_class);
```

Improper Resource Access Authorization\Path 7:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=291
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2252
Object	fscanf	fscanf

Code Snippet

File Name nmap/linear.cpp

Method struct model *load_model(const char *model_file_name)

```
....
2252.                    fscanf(fp,"%d",&nr_feature);
```

Improper Resource Access Authorization\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=292
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2257	2257
Object	fscanf	fscanf

Code Snippet

File Name nmap/linear.cpp

Method struct model *load_model(const char *model_file_name)

```
....
2257.                    fscanf(fp,"%lf",&bias);
```

Improper Resource Access Authorization\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=293
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2269	2269

Object	fscanf	fscanf

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2269.                         fscanf(fp,"%d",&model_->label[i]);
```

Improper Resource Access Authorization\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=294
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2296	2296
Object	fscanf	fscanf

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2296.                    fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Improper Resource Access Authorization\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=295
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2297	2297
Object	fscanf	fscanf

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2297.             fscanf(fp, "\n");
```

Improper Resource Access Authorization\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=296
Status	New

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	968	968
Object	buf	buf

Code Snippet
File Name nmap/knownhost.c
Method libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
968.          while(fgets(buf, sizeof(buf), file)) {
```

Improper Resource Access Authorization\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=297
Status	New

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	59	59
Object	line	line

Code Snippet
File Name nmap/train.c
Method static char* readline(FILE *input)

```
....
59.    if(fgets(line,max_line_len,input) == NULL)
```

Improper Resource Access Authorization\Path 14:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=298
Status	New

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	67	67
Object	BinaryExpr	BinaryExpr

Code Snippet

File Name nmap/train.c

Method static char* readline(FILE *input)

```
....
67.          if(fgets(line+len,max_line_len-len,input) == NULL)
```

Improper Resource Access Authorization\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=299
Status	New

	Source	Destination
File	nmap/blast.c	nmap/blast.c
Line	437	437
Object	hold	hold

Code Snippet

File Name nmap/blast.c

Method local unsigned inf(void *how, unsigned char **buf)

```
....
437.       return fread(hold, 1, CHUNK, (FILE *)how);
```

Improper Resource Access Authorization\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=300
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2224	2224

Object	cmd	cmd

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2224.             fscanf(fp,"%80s",cmd);
```

Improper Resource Access Authorization\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=301
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2227	2227
Object	cmd	cmd

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2227.                   fscanf(fp,"%80s",cmd);
```

Improper Resource Access Authorization\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=302
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2247
Object	Address	Address

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2247.                   fscanf(fp,"%d",&nr_class);
```

Improper Resource Access Authorization\Path 19:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=303
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2252
Object	Address	Address

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2252.                   fscanf(fp,"%d",&nr_feature);
```

Improper Resource Access Authorization\Path 20:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=304
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2257	2257
Object	Address	Address

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2257.                   fscanf(fp,"%lf",&bias);
```

Improper Resource Access Authorization\Path 21:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=305
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2269	2269
Object	Address	Address

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2269.                           fscanf(fp,"%d",&model_->label[i]);
```

Improper Resource Access Authorization\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=306
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2296	2296
Object	Address	Address

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2296.                      fscanf(fp, "%lf ", &model_->w[i*nr_w+j]);
```

Improper Resource Access Authorization\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=307
Status	New

	Source	Destination
File	nmap/blast.c	nmap/blast.c
Line	455	455

Object	fprintf	fprintf

Code Snippet
File Name nmap/blast.c
Method int main(void)

```
....
455.            fprintf(stderr, "blast error: %d\n", ret);
```

Improper Resource Access Authorization\Path 24:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=308
Status	New

	Source	Destination
File	nmap/blast.c	nmap/blast.c
Line	461	461
Object	fprintf	fprintf

Code Snippet
File Name nmap/blast.c
Method int main(void)

```
....
461.            fprintf(stderr, "blast warning: %u unused bytes of
input\n", left);
```

Improper Resource Access Authorization\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=309
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1888	1888
Object	fprintf	fprintf

Code Snippet
File Name nmap/linear.cpp
Method static void train_one(const problem *prob, const parameter *param, double *w, double Cp, double Cn)

```
....
1888.                   fprintf(stderr, "Error: unknown solver_type\n");
```

Improper Resource Access Authorization\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=310
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1935	1935
Object	fprintf	fprintf

Code Snippet
File Name nmap/linear.cpp
Method model* train(const problem *prob, const parameter *param)

```
....
1935.                   fprintf(stderr,"WARNING: class label %d
specified in weight is not found\n", param->weight_label[i]);
```

Improper Resource Access Authorization\Path 27:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=311
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2182	2182
Object	fprintf	fprintf

Code Snippet
File Name nmap/linear.cpp
Method int save_model(const char *model_file_name, const struct model *model_)

```
....
2182.         fprintf(fp, "solver_type %s\n",
solver_type_table[param.solver_type]);
```

Improper Resource Access Authorization\Path 28:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=312
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2183	2183
Object	fprintf	fprintf

Code Snippet
File Name nmap/linear.cpp
Method int save_model(const char *model_file_name, const struct model *model_)

```
....
2183.       fprintf(fp, "nr_class %d\n", model_->nr_class);
```

Improper Resource Access Authorization\Path 29:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=313
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2184	2184
Object	fprintf	fprintf

Code Snippet
File Name nmap/linear.cpp
Method int save_model(const char *model_file_name, const struct model *model_)

```
....
2184.       fprintf(fp, "label");
```

Improper Resource Access Authorization\Path 30:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=314
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp

Line	2186	2186
Object	fprintf	fprintf

Code Snippet

File Name	nmap/linear.cpp
Method	int save_model(const char *model_file_name, const struct model *model_)

```
....
2186.                   fprintf(fp, " %d", model_->label[i]);
```

Improper Resource Access Authorization\Path 31:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=315
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2187	2187
Object	fprintf	fprintf

Code Snippet

File Name	nmap/linear.cpp
Method	int save_model(const char *model_file_name, const struct model *model_)

```
....
2187.             fprintf(fp, "\n");
```

Improper Resource Access Authorization\Path 32:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=316
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2189	2189
Object	fprintf	fprintf

Code Snippet

File Name	nmap/linear.cpp
Method	int save_model(const char *model_file_name, const struct model *model_)

```
....
2189.        fprintf(fp, "nr_feature %d\n", nr_feature);
```

Improper Resource Access Authorization\Path 33:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=317
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2191	2191
Object	fprintf	fprintf

Code Snippet

File Name nmap/linear.cpp

Method int save_model(const char *model_file_name, const struct model *model_)

```
....
2191.        fprintf(fp, "bias %.16g\n", model_->bias);
```

Improper Resource Access Authorization\Path 34:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=318
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2193	2193
Object	fprintf	fprintf

Code Snippet

File Name nmap/linear.cpp

Method int save_model(const char *model_file_name, const struct model *model_)

```
....
2193.        fprintf(fp, "w\n");
```

Improper Resource Access Authorization\Path 35:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=319
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2198	2198
Object	fprintf	fprintf

Code Snippet
File Name nmap/linear.cpp
Method int save_model(const char *model_file_name, const struct model *model_)

```
....
2198.                   fprintf(fp, "%.16g ", model_->w[i*nr_w+j]);
```

Improper Resource Access Authorization\Path 36:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=320
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2199	2199
Object	fprintf	fprintf

Code Snippet
File Name nmap/linear.cpp
Method int save_model(const char *model_file_name, const struct model *model_)

```
....
2199.              fprintf(fp, "\n");
```

Improper Resource Access Authorization\Path 37:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=321
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2239	2239

Object	fprintf	fprintf

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2239.                        fprintf(stderr,"unknown solver type.\n");
```

Improper Resource Access Authorization\Path 38:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=322
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2273	2273
Object	fprintf	fprintf

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2273.                 fprintf(stderr,"unknown text in model file: 
[%s]\n",cmd);
```

Improper Resource Access Authorization\Path 39:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=323
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2990	2990
Object	fprintf	fprintf

Code Snippet
File Name nmap/optimize.c
Method dot_dump_node(struct icode *ic, struct block *block, struct bpf_program *prog,

```
....
2990.       fprintf(out, "\tblock%u [shape=ellipse, id=\"block-%u\"
label=\"BLOCK%u\\n", block->id, block->id, block->id);
```

Improper Resource Access Authorization\Path 40:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=324
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2992	2992
Object	fprintf	fprintf

Code Snippet
File Name nmap/optimize.c
Method dot_dump_node(struct icode *ic, struct block *block, struct bpf_program *prog,

```
....
2992.             fprintf(out, "\\n%s", bpf_image(prog->bf_insns + i,
i));
```

Improper Resource Access Authorization\Path 41:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=325
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2994	2994
Object	fprintf	fprintf

Code Snippet
File Name nmap/optimize.c
Method dot_dump_node(struct icode *ic, struct block *block, struct bpf_program *prog,

```
....
2994.       fprintf(out, "\" tooltip=\"");
```

Improper Resource Access Authorization\Path 42:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=326
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2997	2997
Object	fprintf	fprintf

Code Snippet
File Name nmap/optimize.c
Method dot_dump_node(struct icode *ic, struct block *block, struct bpf_program *prog,

```
....
2997.                   fprintf(out, "val[%d]=%d ", i, block->val[i]);
```

Improper Resource Access Authorization\Path 43:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=327
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2998	2998
Object	fprintf	fprintf

Code Snippet
File Name nmap/optimize.c
Method dot_dump_node(struct icode *ic, struct block *block, struct bpf_program *prog,

```
....
2998.         fprintf(out, "val[A]=%d ", block->val[A_ATOM]);
```

Improper Resource Access Authorization\Path 44:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=328
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c

Line	2999	2999
Object	fprintf	fprintf

Code Snippet

File Name	nmap/optimize.c
Method	dot_dump_node(struct icode *ic, struct block *block, struct bpf_program *prog,

```
....
2999.        fprintf(out, "val[X]=%d", block->val[X_ATOM]);
```

Improper Resource Access Authorization\Path 45:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=329
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	3000	3000
Object	fprintf	fprintf

Code Snippet

File Name	nmap/optimize.c
Method	dot_dump_node(struct icode *ic, struct block *block, struct bpf_program *prog,

```
....
3000.        fprintf(out, "\"");
```

Improper Resource Access Authorization\Path 46:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=330
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	3002	3002
Object	fprintf	fprintf

Code Snippet

File Name	nmap/optimize.c
Method	dot_dump_node(struct icode *ic, struct block *block, struct bpf_program *prog,

```
....
3002.               fprintf(out, ", peripheries=2");
```

Improper Resource Access Authorization\Path 47:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=331
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	3003	3003
Object	fprintf	fprintf

Code Snippet
File Name nmap/optimize.c
Method dot_dump_node(struct icode *ic, struct block *block, struct bpf_program *prog,

```
....
3003.        fprintf(out, "];\n");
```

Improper Resource Access Authorization\Path 48:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=332
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	3017	3017
Object	fprintf	fprintf

Code Snippet
File Name nmap/optimize.c
Method dot_dump_edge(struct icode *ic, struct block *block, FILE *out)

```
....
3017.               fprintf(out, "\t\"block%u\":se -> \"block%u\":n
[label=\"T\"]; \n",
```

Improper Resource Access Authorization\Path 49:

Severity	Low
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=333
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	3019	3019
Object	fprintf	fprintf

Code Snippet
File Name nmap/optimize.c
Method dot_dump_edge(struct icode *ic, struct block *block, FILE *out)

```
....
3019.                fprintf(out, "\t\"block%u\":sw -> \"block%u\":n
[label=\"F\"]; \n",
```

Improper Resource Access Authorization\Path 50:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=334
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	3056	3056
Object	fprintf	fprintf

Code Snippet
File Name nmap/optimize.c
Method dot_dump(struct icode *ic, char *errbuf)

```
....
3056.        fprintf(out, "digraph BPF {\n");
```

Heuristic Buffer Overflow malloc

Query Path:
CPP\Cx\CPP Heuristic\Heuristic Buffer Overflow malloc Version:0

Categories

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

Description

Heuristic Buffer Overflow malloc\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=131
Status	New

The size of the buffer used by *load_model in nr_class, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2224	2267
Object	fp	nr_class

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2224.               fscanf(fp,"%80s",cmd);
....
2267.                       model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=132
Status	New

The size of the buffer used by *load_model in nr_class, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2227	2267
Object	fp	nr_class

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2227.                       fscanf(fp,"%80s",cmd);
....
2267.                       model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=133
Status	New

The size of the buffer used by *load_model in nr_class, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2267
Object	fp	nr_class

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2247.                   fscanf(fp,"%d",&nr_class);
....
2267.                   model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=134
Status	New

The size of the buffer used by *load_model in nr_class, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2267
Object	fp	nr_class

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2252.                   fscanf(fp,"%d",&nr_feature);
....
2267.                   model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=135
Status	New

The size of the buffer used by *load_model in nr_class, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2257	2267
Object	fp	nr_class

Code Snippet	
File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2257.                       fscanf(fp,"%lf",&bias);
....
2267.                       model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=136
Status	New

The size of the buffer used by *load_model in nr_class, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2269	2267
Object	fp	nr_class

Code Snippet	
File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2269.                           fscanf(fp,"%d",&model_->label[i]);
....
2267.                     model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=137
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2224	2267
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2224.             fscanf(fp,"%80s",cmd);
....
2267.                     model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=138
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2227	2267
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp

Method	struct model *load_model(const char *model_file_name)

```
....
2227.                   fscanf(fp,"%80s",cmd);
....
2267.                   model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=139
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2267
Object	fp	BinaryExpr

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2247.                   fscanf(fp,"%d",&nr_class);
....
2267.                   model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=140
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2267
Object	fp	BinaryExpr

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2252.                   fscanf(fp,"%d",&nr_feature);
....
2267.                   model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=141
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2257	2267
Object	fp	BinaryExpr

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2257.                   fscanf(fp,"%lf",&bias);
....
2267.                   model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=142
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2269	2267
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2269.                           fscanf(fp,"%d",&model_->label[i]);
....
2267.                     model_->label = Malloc(int,nr_class);
```

Heuristic Buffer Overflow malloc\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=143
Status	New

The size of the buffer used by *load_model in nr_w, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2224	2291
Object	fp	nr_w

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2224.             fscanf(fp,"%80s",cmd);
....
2291.       model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=144
Status	New

The size of the buffer used by *load_model in nr_w, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2227	2291
Object	fp	nr_w

Code Snippet	
File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2227.                       fscanf(fp,"%80s",cmd);
....
2291.            model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=145
Status	New

The size of the buffer used by *load_model in nr_w, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2291
Object	fp	nr_w

Code Snippet	
File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2247.                       fscanf(fp,"%d",&nr_class);
....
2291.            model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=146
Status	New

The size of the buffer used by *load_model in nr_w, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2291

Object	fp	nr_w

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2252.                          fscanf(fp,"%d",&nr_feature);
....
2291.             model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=147
Status	New

The size of the buffer used by *load_model in nr_w, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2257	2291
Object	fp	nr_w

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2257.                          fscanf(fp,"%lf",&bias);
....
2291.             model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=148
Status	New

The size of the buffer used by *load_model in nr_w, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp

Line	2269	2291
Object	fp	nr_w

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2269.                         fscanf(fp,"%d",&model_->label[i]);
....
2291.       model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 19:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=149
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2224	2291
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2224.             fscanf(fp,"%80s",cmd);
....
2291.       model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 20:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=150
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination

File	nmap/linear.cpp	nmap/linear.cpp
Line	2227	2291
Object	fp	BinaryExpr

Code Snippet
File Name   nmap/linear.cpp
Method   struct model *load_model(const char *model_file_name)

```
....
2227.                        fscanf(fp,"%80s",cmd);
....
2291.            model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 21:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=151
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2291
Object	fp	BinaryExpr

Code Snippet
File Name   nmap/linear.cpp
Method   struct model *load_model(const char *model_file_name)

```
....
2247.                        fscanf(fp,"%d",&nr_class);
....
2291.            model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=152
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2291
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2252.                   fscanf(fp,"%d",&nr_feature);
....
2291.         model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=153
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2257	2291
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2257.                   fscanf(fp,"%lf",&bias);
....
2291.         model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 24:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=154
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2269	2291
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2269.                          fscanf(fp,"%d",&model_->label[i]);
....
2291.         model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=155
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2224	2291
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2224.              fscanf(fp,"%80s",cmd);
....
2291.         model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=156
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2227	2291
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2227.                   fscanf(fp,"%80s",cmd);
....
2291.         model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 27:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=157
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2291
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2247.                   fscanf(fp,"%d",&nr_class);
....
2291.         model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 28:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=158

Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2291
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2252.                   fscanf(fp,"%d",&nr_feature);
....
2291.         model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 29:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=159
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2257	2291
Object	fp	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2257.                   fscanf(fp,"%lf",&bias);
....
2291.         model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 30:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=500

	51&pathid=160
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2269	2291
Object	fp	BinaryExpr

Code Snippet
File Name   nmap/linear.cpp
Method   struct model *load_model(const char *model_file_name)

```
....
2269.                        fscanf(fp,"%d",&model_->label[i]);
....
2291.          model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 31:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=161
Status	New

The size of the buffer used by *load_model in w_size, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2224	2291
Object	fp	w_size

Code Snippet
File Name   nmap/linear.cpp
Method   struct model *load_model(const char *model_file_name)

```
....
2224.              fscanf(fp,"%80s",cmd);
....
2291.          model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 32:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=162
Status	New

The size of the buffer used by *load_model in w_size, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2227	2291
Object	fp	w_size

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2227.                     fscanf(fp,"%80s",cmd);
....
2291.          model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 33:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=163
Status	New

The size of the buffer used by *load_model in w_size, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2291
Object	fp	w_size

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2247.                     fscanf(fp,"%d",&nr_class);
....
2291.          model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 34:

Severity	Low
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=164
Status	New

The size of the buffer used by *load_model in w_size, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2291
Object	fp	w_size

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2252.                   fscanf(fp,"%d",&nr_feature);
....
2291.       model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 35:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=165
Status	New

The size of the buffer used by *load_model in w_size, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2257	2291
Object	fp	w_size

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2257.                   fscanf(fp,"%lf",&bias);
....
2291.       model_->w=Malloc(double, w_size*nr_w);
```

Heuristic Buffer Overflow malloc\Path 36:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=166
Status	New

The size of the buffer used by *load_model in w_size, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to fp, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2269	2291
Object	fp	w_size

Code Snippet

File Name nmap/linear.cpp

Method struct model *load_model(const char *model_file_name)

```
....
2269.                             fscanf(fp,"%d",&model_->label[i]);
....
2291.            model_->w=Malloc(double, w_size*nr_w);
```

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=1050061&projectid=50051&pathid=75
Status	New

The knownhost_writeline method calls the snprintf function, at line 997 of nmap/knownhost.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	nmap/knownhost.c	nmap/knownhost.c
Line	1114	1114
Object	snprintf	snprintf

Code Snippet

File Name nmap/knownhost.c

Method	knownhost_writeline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1114.                   snprintf(buf, buflen, "|1|%s|%s %s %s %s\n",
saltalloc,
```

Unchecked Return Value\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=76
Status	New

The knownhost_writeline method calls the snprintf function, at line 997 of nmap/knownhost.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	nmap/knownhost.c	nmap/knownhost.c
Line	1117	1117
Object	snprintf	snprintf

Code Snippet

File Name	nmap/knownhost.c
Method	knownhost_writeline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1117.                   snprintf(buf, buflen, "|1|%s|%s %s %s\n",
saltalloc, namealloc,
```

Unchecked Return Value\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=77
Status	New

The knownhost_writeline method calls the snprintf function, at line 997 of nmap/knownhost.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	nmap/knownhost.c	nmap/knownhost.c
Line	1120	1120
Object	snprintf	snprintf

Code Snippet

File Name nmap/knownhost.c
Method knownhost_writeline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1120.                   snprintf(buf, buflen, "|1|%s|%s %s %s\n",
saltalloc, namealloc,
```

Unchecked Return Value\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=78
Status	New

The knownhost_writeline method calls the snprintf function, at line 997 of nmap/knownhost.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	nmap/knownhost.c	nmap/knownhost.c
Line	1123	1123
Object	snprintf	snprintf

Code Snippet
File Name nmap/knownhost.c
Method knownhost_writeline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1123.                   snprintf(buf, buflen, "|1|%s|%s %s\n", saltalloc,
namealloc,
```

Unchecked Return Value\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=79
Status	New

The knownhost_writeline method calls the snprintf function, at line 997 of nmap/knownhost.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	nmap/knownhost.c	nmap/knownhost.c
Line	1136	1136
Object	snprintf	snprintf

Code Snippet	
File Name	nmap/knownhost.c
Method	knownhost_writeline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1136.                   snprintf(buf, buflen, "%s %s %s %s\n", node-
>name,
```

Unchecked Return Value\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=80
Status	New

The knownhost_writeline method calls the snprintf function, at line 997 of nmap/knownhost.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	nmap/knownhost.c	nmap/knownhost.c
Line	1139	1139
Object	snprintf	snprintf

Code Snippet	
File Name	nmap/knownhost.c
Method	knownhost_writeline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1139.                   snprintf(buf, buflen, "%s %s %s\n", node->name,
node->key,
```

Unchecked Return Value\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=81
Status	New

The knownhost_writeline method calls the snprintf function, at line 997 of nmap/knownhost.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	nmap/knownhost.c	nmap/knownhost.c
Line	1142	1142
Object	snprintf	snprintf

Code Snippet
File Name nmap/knownhost.c
Method knownhost_writeline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1142.                  snprintf(buf, buflen, "%s %s %s\n", node->name,
key_type_name,
```

Unchecked Return Value\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=82
Status	New

The knownhost_writeline method calls the snprintf function, at line 997 of nmap/knownhost.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	nmap/knownhost.c	nmap/knownhost.c
Line	1145	1145
Object	snprintf	snprintf

Code Snippet
File Name nmap/knownhost.c
Method knownhost_writeline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1145.                  snprintf(buf, buflen, "%s %s\n", node->name,
node->key);
```

Unchecked Return Value\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=83
Status	New

The icode_to_fcode method calls the snprintf function, at line 2876 of nmap/optimize.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	nmap/optimize.c	nmap/optimize.c
Line	2900	2900

Object	snprintf	snprintf

Code Snippet

File Name	nmap/optimize.c
Method	icode_to_fcode(struct icode *ic, struct block *root, u_int *lenp,

```
....
2900.              (void)snprintf(errbuf, PCAP_ERRBUF_SIZE,
```

Unchecked Return Value\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=84
Status	New

The install_bpf_program method calls the snprintf function, at line 2945 of nmap/optimize.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	nmap/optimize.c	nmap/optimize.c
Line	2953	2953
Object	snprintf	snprintf

Code Snippet

File Name	nmap/optimize.c
Method	install_bpf_program(pcap_t *p, struct bpf_program *fp)

```
....
2953.              snprintf(p->errbuf, sizeof(p->errbuf),
```

Unchecked Return Value\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=85
Status	New

The parse_command_line method calls the sprintf function, at line 140 of nmap/train.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	nmap/train.c	nmap/train.c
Line	226	226

Object	sprintf	sprintf

Code Snippet
File Name nmap/train.c
Method void parse_command_line(int argc, char **argv, char *input_file_name, char *model_file_name)

```
....
226.              sprintf(model_file_name,"%s.model",p);
```

Unchecked Return Value\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=86
Status	New

The group_classes method calls the count function, at line 1768 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	1797	1797
Object	count	count

Code Snippet
File Name nmap/linear.cpp
Method static void group_classes(const problem *prob, int *nr_class_ret, int **label_ret, int **start_ret, int **count_ret, int *perm)

```
....
1797.                        count = (int
*)realloc(count,max_nr_class*sizeof(int));
```

Unchecked Return Value\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=87
Status	New

The train method calls the BinaryExpr function, at line 1896 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp

Line	1941	1941
Object	BinaryExpr	BinaryExpr

Code Snippet
File Name nmap/linear.cpp
Method model* train(const problem *prob, const parameter *param)

```
....
1941.        feature_node **x = Malloc(feature_node *,l);
```

Unchecked Return Value\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=88
Status	New

The train method calls the x function, at line 1896 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	1949	1949
Object	x	x

Code Snippet
File Name nmap/linear.cpp
Method model* train(const problem *prob, const parameter *param)

```
....
1949.        sub_prob.x = Malloc(feature_node *,sub_prob.l);
```

Unchecked Return Value\Path 15:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=89
Status	New

The cross_validation method calls the x function, at line 2018 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	2044	2044
Object	x	x

Code Snippet

File Name	nmap/linear.cpp
Method	void cross_validation(const problem *prob, const parameter *param, int nr_fold, int *target)

```
....
2044.               subprob.x = Malloc(struct feature_node*,subprob.l);
```

Unchecked Return Value\Path 16:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=90
Status	New

The read_problem method calls the x function, at line 241 of nmap/train.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	nmap/train.c	nmap/train.c
Line	279	279
Object	x	x

Code Snippet

File Name	nmap/train.c
Method	void read_problem(const char *filename)

```
....
279.          prob.x = Malloc(struct feature_node *,prob.l);
```

Unchecked Return Value\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=91
Status	New

The group_classes method calls the count function, at line 1768 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	1774	1774
Object	count	count

Code Snippet	
File Name	nmap/linear.cpp
Method	static void group_classes(const problem *prob, int *nr_class_ret, int **label_ret, int **start_ret, int **count_ret, int *perm)

```
....
1774.       int *count = Malloc(int,max_nr_class);
```

Unchecked Return Value\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=92
Status	New

The group_classes method calls the data_label function, at line 1768 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	1775	1775
Object	data_label	data_label

Code Snippet	
File Name	nmap/linear.cpp
Method	static void group_classes(const problem *prob, int *nr_class_ret, int **label_ret, int **start_ret, int **count_ret, int *perm)

```
....
1775.       int *data_label = Malloc(int,l);
```

Unchecked Return Value\Path 19:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=93
Status	New

The group_classes method calls the start function, at line 1768 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	1805	1805
Object	start	start

Code Snippet
File Name nmap/linear.cpp
Method static void group_classes(const problem *prob, int *nr_class_ret, int **label_ret, int **start_ret, int **count_ret, int *perm)

```
....
1805.       int *start = Malloc(int,nr_class);
```

Unchecked Return Value\Path 20:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=94
Status	New

The train method calls the model_ function, at line 1896 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	1902	1902
Object	model_	model_

Code Snippet
File Name nmap/linear.cpp
Method model* train(const problem *prob, const parameter *param)

```
....
1902.       model *model_ = Malloc(model,1);
```

Unchecked Return Value\Path 21:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=95
Status	New

The train method calls the perm function, at line 1896 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	1915	1915
Object	perm	perm

Code Snippet
File Name nmap/linear.cpp

Method	model* train(const problem *prob, const parameter *param)

```
....
1915.       int *perm = Malloc(int,l);
```

Unchecked Return Value\Path 22:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=96
Status	New

The train method calls the weighted_C function, at line 1896 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	1926	1926
Object	weighted_C	weighted_C

Code Snippet

File Name	nmap/linear.cpp
Method	model* train(const problem *prob, const parameter *param)

```
....
1926.       double *weighted_C = Malloc(double, nr_class);
```

Unchecked Return Value\Path 23:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=97
Status	New

The train method calls the w function, at line 1896 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	1983	1983
Object	w	w

Code Snippet

File Name	nmap/linear.cpp
Method	model* train(const problem *prob, const parameter *param)

```
....
1983.                   double *w=Malloc(double, w_size);
```

Unchecked Return Value\Path 24:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=98
Status	New

The cross_validation method calls the fold_start function, at line 2018 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	2021	2021
Object	fold_start	fold_start

Code Snippet
File Name nmap/linear.cpp
Method void cross_validation(const problem *prob, const parameter *param, int nr_fold, int *target)

```
....
2021.       int *fold_start = Malloc(int,nr_fold+1);
```

Unchecked Return Value\Path 25:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=99
Status	New

The cross_validation method calls the perm function, at line 2018 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	2023	2023
Object	perm	perm

Code Snippet
File Name nmap/linear.cpp

Method	void cross_validation(const problem *prob, const parameter *param, int nr_fold, int *target)

```
....
2023.        int *perm = Malloc(int,l);
```

Unchecked Return Value\Path 26:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=100
Status	New

The predict method calls the dec_values function, at line 2113 of nmap/linear.cpp. 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	nmap/linear.cpp	nmap/linear.cpp
Line	2115	2115
Object	dec_values	dec_values

Code Snippet

File Name	nmap/linear.cpp
Method	int predict(const model *model_, const feature_node *x)

```
....
2115.        double *dec_values = Malloc(double, model_->nr_class);
```

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=1050061&projectid=50051&pathid=175
Status	New

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	229	229
Object	key_type_len	key_type_len

Code Snippet	
File Name	nmap/knownhost.c
Method	knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
229.           entry->key_type_name[key_type_len] = 0;
```

Unchecked Array Index\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=176
Status	New

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	241	241
Object	commentlen	commentlen

Code Snippet	
File Name	nmap/knownhost.c
Method	knownhost_add(LIBSSH2_KNOWNHOSTS *hosts,

```
....
241.           entry->comment[commentlen] = 0; /* force a terminating
zero trailer */
```

Unchecked Array Index\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=177
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1244	1244
Object	ind	ind

Code Snippet	
File Name	nmap/linear.cpp
Method	static void solve_l1r_l2_svc(

```
....
1244.                              b[ind] = b_new;
```

Unchecked Array Index\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=178
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1559	1559
Object	ind	ind

Code Snippet
File Name nmap/linear.cpp
Method static void solve_l1r_lr(

```
....
1559.                               xTd[ind] += x->value*z;
```

Unchecked Array Index\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=179
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2050	2050
Object	k	k

Code Snippet
File Name nmap/linear.cpp
Method void cross_validation(const problem *prob, const parameter *param, int nr_fold, int *target)

```
....
2050.                     subprob.x[k] = prob->x[perm[j]];
```

Unchecked Array Index\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=180
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2051	2051
Object	k	k

Code Snippet
File Name nmap/linear.cpp
Method void cross_validation(const problem *prob, const parameter *param, int nr_fold, int *target)

```
....
2051.                    subprob.y[k] = prob->y[perm[j]];
```

Unchecked Array Index\Path 7:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=181
Status New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2056	2056
Object	k	k

Code Snippet
File Name nmap/linear.cpp
Method void cross_validation(const problem *prob, const parameter *param, int nr_fold, int *target)

```
....
2056.                    subprob.x[k] = prob->x[perm[j]];
```

Unchecked Array Index\Path 8:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=182
Status New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2057	2057

Object	k	k

Code Snippet
File Name nmap/linear.cpp
Method void cross_validation(const problem *prob, const parameter *param, int nr_fold, int *target)

```
....
2057.                        subprob.y[k] = prob->y[perm[j]];
```

Unchecked Array Index\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=183
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	397	397
Object	level	level

Code Snippet
File Name nmap/optimize.c
Method find_levels_r(opt_state_t *opt_state, struct icode *ic, struct block *b)

```
....
397.          opt_state->levels[level] = b;
```

Unchecked Array Index\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=184
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	447	447
Object	dom	dom

Code Snippet
File Name nmap/optimize.c
Method find_dom(opt_state_t *opt_state, struct block *root)

```
....
447.                        SET_INSERT(b->dom, b->id);
```

Unchecked Array Index\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=185
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	459	459
Object	edom	edom

Code Snippet

File Name	nmap/optimize.c
Method	propedom(opt_state_t *opt_state, struct edge *ep)

```
....
459.            SET_INSERT(ep->edom, ep->id);
```

Unchecked Array Index\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=186
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	520	520
Object	closure	closure

Code Snippet

File Name	nmap/optimize.c
Method	find_closure(opt_state_t *opt_state, struct block *root)

```
....
520.                        SET_INSERT(b->closure, b->id);
```

Unchecked Array Index\Path 13:

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

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=187
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	763	763
Object	hash	hash

Code Snippet

File Name	nmap/optimize.c
Method	F(opt_state_t *opt_state, int code, bpf_u_int32 v0, bpf_u_int32 v1)

```
....
763.        opt_state->hashtbl[hash] = p;
```

Unchecked Array Index\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=188
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2478	2478
Object	n	n

Code Snippet

File Name	nmap/optimize.c
Method	number_blks_r(opt_state_t *opt_state, struct icode *ic, struct block *p)

```
....
2478.       opt_state->blocks[n] = p;
```

Heuristic 2nd Order Buffer Overflow malloc

Query Path:
CPP\Cx\CPP Heuristic\Heuristic 2nd Order Buffer Overflow malloc Version:0

Categories

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

Description

Heuristic 2nd Order Buffer Overflow malloc\Path 1:

Severity	Low
Result State	To Verify

Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=120
Status	New

The size of the buffer used by *load_model in nr_class, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2267
Object	Address	nr_class

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2247.                   fscanf(fp,"%d",&nr_class);
....
2267.                   model_->label = Malloc(int,nr_class);
```

Heuristic 2nd Order Buffer Overflow malloc\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=121
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2267
Object	Address	BinaryExpr

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2247.                   fscanf(fp,"%d",&nr_class);
....
2267.                   model_->label = Malloc(int,nr_class);
```

Heuristic 2nd Order Buffer Overflow malloc\Path 3:

Severity	Low

Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=122
Status	New

The size of the buffer used by *load_model in nr_w, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2291
Object	Address	nr_w

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2247.                       fscanf(fp,"%d",&nr_class);
....
2291.          model_->w=Malloc(double, w_size*nr_w);
```

Heuristic 2nd Order Buffer Overflow malloc\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=123
Status	New

The size of the buffer used by *load_model in nr_w, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2291
Object	Address	nr_w

Code Snippet
File Name nmap/linear.cpp
Method struct model *load_model(const char *model_file_name)

```
....
2252.                       fscanf(fp,"%d",&nr_feature);
....
2291.          model_->w=Malloc(double, w_size*nr_w);
```

Heuristic 2nd Order Buffer Overflow malloc\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=124
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2291
Object	Address	BinaryExpr

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2247.                       fscanf(fp,"%d",&nr_class);
....
2291.           model_->w=Malloc(double, w_size*nr_w);
```

Heuristic 2nd Order Buffer Overflow malloc\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=125
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2291
Object	Address	BinaryExpr

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2252.                       fscanf(fp,"%d",&nr_feature);
....
2291.           model_->w=Malloc(double, w_size*nr_w);
```

Heuristic 2nd Order Buffer Overflow malloc\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=126
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2291
Object	Address	BinaryExpr

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2247.                         fscanf(fp,"%d",&nr_class);
....
2291.           model_->w=Malloc(double, w_size*nr_w);
```

Heuristic 2nd Order Buffer Overflow malloc\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=127
Status	New

The size of the buffer used by *load_model in BinaryExpr, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2291
Object	Address	BinaryExpr

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2252.                         fscanf(fp,"%d",&nr_feature);
....
2291.           model_->w=Malloc(double, w_size*nr_w);
```

Heuristic 2nd Order Buffer Overflow malloc\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=128
Status	New

The size of the buffer used by *load_model in w_size, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2247	2291
Object	Address	w_size

Code Snippet

File Name nmap/linear.cpp

Method struct model *load_model(const char *model_file_name)

```
....
2247.                       fscanf(fp,"%d",&nr_class);
....
2291.         model_->w=Malloc(double, w_size*nr_w);
```

Heuristic 2nd Order Buffer Overflow malloc\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=129
Status	New

The size of the buffer used by *load_model in w_size, at line 2206 of nmap/linear.cpp, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *load_model passes to Address, at line 2206 of nmap/linear.cpp, to overwrite the target buffer.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2252	2291
Object	Address	w_size

Code Snippet

File Name nmap/linear.cpp

Method struct model *load_model(const char *model_file_name)

```
....
2252.                fscanf(fp,"%d",&nr_feature);
....
2291.        model_->w=Malloc(double, w_size*nr_w);
```

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=1050061&projectid=50051&pathid=111
Status	New

The variable declared in null at nmap/blast.c in line 446 is not initialized when it is used by in at nmap/blast.c in line 383.

	Source	Destination
File	nmap/blast.c	nmap/blast.c
Line	453	394
Object	null	in

Code Snippet
File Name   nmap/blast.c
Method   int main(void)

```
....
453.       ret = blast(inf, stdin, outf, stdout, &left, NULL);
```

File Name   nmap/blast.c
Method   int blast(blast_in infun, void *inhow, blast_out outfun, void *outhow,

```
....
394.            s.in = *in;
```

NULL Pointer Dereference\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=500

	51&pathid=112
Status	New

The variable declared in null at nmap/blast.c in line 446 is not initialized when it is used by s at nmap/blast.c in line 383.

	Source	Destination
File	nmap/blast.c	nmap/blast.c
Line	453	394
Object	null	s

Code Snippet

File Name nmap/blast.c

Method int main(void)

```
....
453.        ret = blast(inf, stdin, outf, stdout, &left, NULL);
```

File Name nmap/blast.c

Method int blast(blast_in infun, void *inhow, blast_out outfun, void *outhow,

```
....
394.            s.in = *in;
```

NULL Pointer Dereference\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=113
Status	New

The variable declared in null at nmap/knownhost.c in line 119 is not initialized when it is used by ext at nmap/knownhost.c in line 119.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	126	125
Object	null	ext

Code Snippet

File Name nmap/knownhost.c

Method static struct libssh2_knownhost *knownhost_to_external(struct known_host *node)

```
....
126.                         LIBSSH2_KNOWNHOST_TYPE_PLAIN)? node->name:NULL;
....
125.        ext->name = ((node->typemask & LIBSSH2_KNOWNHOST_TYPE_MASK) ==
```

NULL Pointer Dereference\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=114
Status	New

The variable declared in null at nmap/ncat_main.c in line 218 is not initialized when it is used by node at nmap/ncat_main.c in line 159.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	223	164
Object	null	node

Code Snippet

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

```
....
223.        struct host_list_node *allow_host_list = NULL;
```

File Name	nmap/ncat_main.c
Method	static void host_list_add_spec(struct host_list_node **list, char *spec)

```
....
164.        node->next = *list;
```

NULL Pointer Dereference\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=115
Status	New

The variable declared in null at nmap/ncat_main.c in line 218 is not initialized when it is used by node at nmap/ncat_main.c in line 159.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c

Line	224	164
Object	null	node

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

```
....
224.        struct host_list_node *deny_host_list = NULL;
```

File Name nmap/ncat_main.c

Method static void host_list_add_spec(struct host_list_node **list, char *spec)

```
....
164.        node->next = *list;
```

NULL Pointer Dereference\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=116
Status	New

The variable declared in null at nmap/ncat_main.c in line 218 is not initialized when it is used by node at nmap/ncat_main.c in line 168.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	223	173
Object	null	node

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

```
....
223.        struct host_list_node *allow_host_list = NULL;
```

File Name nmap/ncat_main.c

Method static void host_list_add_filename(struct host_list_node **list, char *filename)

```
....
173.        node->next = *list;
```

NULL Pointer Dereference\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=117
Status	New

The variable declared in null at nmap/ncat_main.c in line 218 is not initialized when it is used by node at nmap/ncat_main.c in line 168.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	224	173
Object	null	node

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

```
....
224.      struct host_list_node *deny_host_list = NULL;
```

File Name nmap/ncat_main.c

Method static void host_list_add_filename(struct host_list_node **list, char *filename)

```
....
173.      node->next = *list;
```

NULL Pointer Dereference\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=118
Status	New

The variable declared in null at nmap/pcre_compile.c in line 5801 is not initialized when it is used by re at nmap/pcre_compile.c in line 5801.

	Source	Destination
File	nmap/pcre_compile.c	nmap/pcre_compile.c
Line	6054	6054
Object	null	re

Code Snippet
File Name nmap/pcre_compile.c
Method pcre_compile2(const char *pattern, int options, int *errorcodeptr,

```
....
6054.   re->tables = (tables == _pcre_default_tables)? NULL : tables;
```

NULL Pointer Dereference\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=119
Status	New

The variable declared in 0 at nmap/lgc.c in line 125 is not initialized when it is used by g at nmap/lgc.c in line 652.

	Source	Destination
File	nmap/lgc.c	nmap/lgc.c
Line	137	655
Object	0	g

Code Snippet
File Name nmap/lgc.c
Method static GCObject **getgclist (GCObject *o) {

```
....
137.      default: lua_assert(0); return 0;
```

▼

File Name nmap/lgc.c

Method static lu_mem propagatemark (global_State *g) {

```
....
655.    g->gray = *getgclist(o);  /* remove from 'gray' list */
```

Use of Insufficiently Random Values

Query Path:
CPP\Cx\CPP Low Visibility\Use of Insufficiently Random Values Version:0

Categories

FISMA 2014: Media Protection
NIST SP 800-53: SC-28 Protection of Information at Rest (P1)
OWASP Top 10 2017: A3-Sensitive Data Exposure

Description

Use of Insufficiently Random Values\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=69

Status	New

Method Solver_MCSVM_CS::Solve at line 480 of nmap/linear.cpp uses a weak method rand to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	522	522
Object	rand	rand

Code Snippet
File Name nmap/linear.cpp
Method void Solver_MCSVM_CS::Solve(double *w)

```
....
522.                    int j = i+rand()%(active_size-i);
```

Use of Insufficiently Random Values\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=70
Status	New

Method solve_l2r_l1l2_svc at line 711 of nmap/linear.cpp uses a weak method rand to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	774	774
Object	rand	rand

Code Snippet
File Name nmap/linear.cpp
Method static void solve_l2r_l1l2_svc(

```
....
774.                    int j = i+rand()%(active_size-i);
```

Use of Insufficiently Random Values\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=71
Status	New

Method solve_l2r_lr_dual at line 912 of nmap/linear.cpp uses a weak method rand to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	957	957
Object	rand	rand

Code Snippet
File Name nmap/linear.cpp
Method void solve_l2r_lr_dual(const problem *prob, double *w, double eps, double Cp, double Cn)

```
....
957.                    int j = i+rand()%(l-i);
```

Use of Insufficiently Random Values\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=72
Status	New

Method solve_l1r_l2_svc at line 1075 of nmap/linear.cpp uses a weak method rand to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1134	1134
Object	rand	rand

Code Snippet
File Name nmap/linear.cpp
Method static void solve_l1r_l2_svc(

```
....
1134.                   int i = j+rand()%(active_size-j);
```

Use of Insufficiently Random Values\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=73
Status	New

Method solve_l1r_lr at line 1358 of nmap/linear.cpp uses a weak method rand to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	1497	1497
Object	rand	rand

Code Snippet
File Name nmap/linear.cpp
Method static void solve_l1r_lr(

```
....
1497.                              int i = j+rand()%(QP_active_size-j);
```

Use of Insufficiently Random Values\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=74
Status	New

Method cross_validation at line 2018 of nmap/linear.cpp uses a weak method rand to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2028	2028
Object	rand	rand

Code Snippet
File Name nmap/linear.cpp
Method void cross_validation(const problem *prob, const parameter *param, int nr_fold, int *target)

```
....
2028.                int j = i+rand()%(l-i);
```

Incorrect Permission Assignment For Critical Resources

Query Path:
CPP\Cx\CPP Low Visibility\Incorrect Permission Assignment For Critical Resources Version:1

Categories

FISMA 2014: Access Control
NIST SP 800-53: AC-3 Access Enforcement (P1)
OWASP Top 10 2017: A2-Broken Authentication

Description

Incorrect Permission Assignment For Critical Resources\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=345
Status	New

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	966	966
Object	file	file

Code Snippet
File Name nmap/knownhost.c
Method libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
966.        file = fopen(filename, FOPEN_READTEXT);
```

Incorrect Permission Assignment For Critical Resources\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=346
Status	New

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	1207	1207
Object	file	file

Code Snippet
File Name nmap/knownhost.c
Method libssh2_knownhost_writefile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1207.        file = fopen(filename, FOPEN_WRITETEXT);
```

Incorrect Permission Assignment For Critical Resources\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=347
Status	New

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	194	194
Object	fd	fd

Code Snippet

File Name	nmap/ncat_main.c
Method	static void host_list_to_set(struct addrset *set, struct host_list_node *list)

```
....
194.                fd = fopen(node->spec, "r");
```

Incorrect Permission Assignment For Critical Resources\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=348
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2173	2173
Object	fp	fp

Code Snippet

File Name	nmap/linear.cpp
Method	int save_model(const char *model_file_name, const struct model *model_)

```
....
2173.         FILE *fp = fopen(model_file_name,"w");
```

Incorrect Permission Assignment For Critical Resources\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=349
Status	New

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2208	2208
Object	fp	fp

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2208.        FILE *fp = fopen(model_file_name,"r");
```

Incorrect Permission Assignment For Critical Resources\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=350
Status	New

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	245	245
Object	fp	fp

Code Snippet

File Name	nmap/train.c
Method	void read_problem(const char *filename)

```
....
245.        FILE *fp = fopen(filename,"r");
```

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=1050061&projectid=50051&pathid=352
Status	New

The libssh2_knownhost_readfile method in nmap/knownhost.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	966	966
Object	fopen	fopen

Code Snippet

File Name nmap/knownhost.c

Method	libssh2_knownhost_readfile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
966.        file = fopen(filename, FOPEN_READTEXT);
```

TOCTOU\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=353
Status	New

The libssh2_knownhost_writefile method in nmap/knownhost.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	1207	1207
Object	fopen	fopen

Code Snippet	
File Name	nmap/knownhost.c
Method	libssh2_knownhost_writefile(LIBSSH2_KNOWNHOSTS *hosts,

```
....
1207.        file = fopen(filename, FOPEN_WRITETEXT);
```

TOCTOU\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=354
Status	New

The save_model method in nmap/linear.cpp file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2173	2173
Object	fopen	fopen

Code Snippet	
File Name	nmap/linear.cpp
Method	int save_model(const char *model_file_name, const struct model *model_)

```
....
2173.        FILE *fp = fopen(model_file_name,"w");
```

TOCTOU\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=355
Status	New

The *load_model method in nmap/linear.cpp file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2208	2208
Object	fopen	fopen

Code Snippet

File Name	nmap/linear.cpp
Method	struct model *load_model(const char *model_file_name)

```
....
2208.        FILE *fp = fopen(model_file_name,"r");
```

TOCTOU\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=356
Status	New

The host_list_to_set method in nmap/ncat_main.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	194	194
Object	fopen	fopen

Code Snippet

File Name	nmap/ncat_main.c
Method	static void host_list_to_set(struct addrset *set, struct host_list_node *list)

```
....
194.                    fd = fopen(node->spec, "r");
```

TOCTOU\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=357
Status	New

The read_problem method in nmap/train.c file utilizes fopen that is accessed by other concurrent functionality in a way that is not thread-safe, which may result in a Race Condition over this resource.

	Source	Destination
File	nmap/train.c	nmap/train.c
Line	245	245
Object	fopen	fopen

Code Snippet

File Name	nmap/train.c
Method	void read_problem(const char *filename)

```
....
245.            FILE *fp = fopen(filename,"r");
```

Potential Off by One Error in Loops

Query Path:
CPP\Cx\CPP Heuristic\Potential Off by One Error in Loops Version:1

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.1 - Injection flaws - particularly SQL injection
NIST SP 800-53: SI-16 Memory Protection (P1)
OWASP Top 10 2017: A1-Injection

Description

Potential Off by One Error in Loops\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=104
Status	New

The buffer allocated by <= in nmap/blast.c at line 191 does not correctly account for the actual size of the value, resulting in an incorrect allocation that is off by one.

	Source	Destination
File	nmap/blast.c	nmap/blast.c

Line	212	212
Object	<=	<=

Code Snippet
File Name nmap/blast.c
Method local int construct(struct huffman *h, const unsigned char *rep, int n)

```
....
212.        for (len = 0; len <= MAXBITS; len++)
```

Potential Off by One Error in Loops\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=105
Status	New

The buffer allocated by <= in nmap/linear.cpp at line 2018 does not correctly account for the actual size of the value, resulting in an incorrect allocation that is off by one.

	Source	Destination
File	nmap/linear.cpp	nmap/linear.cpp
Line	2031	2031
Object	<=	<=

Code Snippet
File Name nmap/linear.cpp
Method void cross_validation(const problem *prob, const parameter *param, int nr_fold, int *target)

```
....
2031.        for(i=0;i<=nr_fold;i++)
```

Potential Off by One Error in Loops\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=106
Status	New

The buffer allocated by <= in nmap/puff.c at line 340 does not correctly account for the actual size of the value, resulting in an incorrect allocation that is off by one.

	Source	Destination
File	nmap/puff.c	nmap/puff.c
Line	348	348

Object	<=	<=

Code Snippet

File Name nmap/puff.c

Method local int construct(struct huffman *h, const short *length, int n)

```
....
348.       for (len = 0; len <= MAXBITS; len++)
```

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=1050061&projectid=50051&pathid=101
Status	New

	Source	Destination
File	nmap/lobject.c	nmap/lobject.c
Line	508	508
Object	sizeof	sizeof

Code Snippet

File Name nmap/lobject.c

Method const char *luaO_pushvfstring (lua_State *L, const char *fmt, va_list argp) {

```
....
508.          const int sz = 3 * sizeof(void*) + 8; /* enough space for
'%p' */
```

Use of Sizeof On a Pointer Type\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=102
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	2714	2714
Object	sizeof	sizeof

Code Snippet

File Name	nmap/optimize.c
Method	convert_code_r(conv_state_t *conv_state, struct icode *ic, struct block *p)

```
....
2714.           offset = (struct slist **)calloc(slen, sizeof(struct
slist *));
```

Sizeof Pointer Argument

Query Path:

CPP\Cx\CPP Low Visibility\Sizeof Pointer Argument Version:0

Description

Sizeof Pointer Argument\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=173
Status	New

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	692	692
Object	saltbuf	sizeof

Code Snippet

File Name	nmap/knownhost.c
Method	static int hashed_hostline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
692.          if(saltlen >= (sizeof(saltbuf)-1)) /* weird length */
```

Sizeof Pointer Argument\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=174
Status	New

	Source	Destination
File	nmap/knownhost.c	nmap/knownhost.c
Line	709	709
Object	hostbuf	sizeof

Code Snippet

File Name	nmap/knownhost.c
Method	static int hashed_hostline(LIBSSH2_KNOWNHOSTS *hosts,

```
....
709.          if(hostlen >= sizeof(hostbuf)-1)
```

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=1050061&projectid=50051&pathid=68
Status	New

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	315	315
Object	getopt_long	getopt_long

Code Snippet

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

```
....
315.          int c = getopt_long(argc, argv,
"46UCc:e:g:G:i:km:hp:d:lo:x:ts:uvw:nz",
```

Potential Precision Problem

Query Path:
CPP\Cx\CPP Buffer Overflow\Potential Precision Problem Version:0

Categories

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

Description

Potential Precision Problem\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=130
Status	New

The size of the buffer used by parse_command_line in "%s.model", at line 140 of nmap/train.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that parse_command_line passes to "%s.model", at line 140 of nmap/train.c, to overwrite the target buffer.

	Source	Destination

File	nmap/train.c	nmap/train.c
Line	226	226
Object	"%s.model"	"%s.model"

Code Snippet

File Name nmap/train.c

Method void parse_command_line(int argc, char **argv, char *input_file_name, char *model_file_name)

```
....
226.            sprintf(model_file_name,"%s.model",p);
```

Arithmenic Operation On Boolean

Query Path:

CPP\Cx\CPP Low Visibility\Arithmenic Operation On Boolean Version:1

Categories

FISMA 2014: Audit And Accountability

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

Description

Arithmenic Operation On Boolean\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=172
Status	New

	Source	Destination
File	nmap/optimize.c	nmap/optimize.c
Line	392	392
Object	BinaryExpr	BinaryExpr

Code Snippet

File Name nmap/optimize.c

Method find_levels_r(opt_state_t *opt_state, struct icode *ic, struct block *b)

```
....
392.            level = MAX(JT(b)->level, JF(b)->level) + 1;
```

Exposure of System Data to Unauthorized Control Sphere

Query Path:

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

Categories

FISMA 2014: Configuration Management

NIST SP 800-53: AC-3 Access Enforcement (P1)

Description

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

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050061&projectid=50051&pathid=351
Status	New

The system data read by main in the file nmap/ncat_main.c at line 218 is potentially exposed by main found in nmap/ncat_main.c at line 218.

	Source	Destination
File	nmap/ncat_main.c	nmap/ncat_main.c
Line	559	559
Object	perror	perror

Code Snippet

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

```
....
559.                        perror("Cannot set mode");
```

Buffer Overflow Indexes

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.

- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Buffer Overflow StrcpyStrcat

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Buffer Overflow IndexFromInput

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Buffer Overflow OutOfBound

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

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

Divide By Zero

Risk

What might happen

When a program divides a number by zero, an exception will be raised. If this exception is not handled by the application, unexpected results may occur, including crashing the application. This can be considered a DoS (Denial of Service) attack, if an external user has control of the value of the denominator or can cause this error to occur.

Cause

How does it happen

The program receives an unexpected value, and uses it for division without filtering, validation, or verifying that the value is not zero. The application does not explicitly handle this error or prevent division by zero from occuring.

General Recommendations

How to avoid it

- Before dividing by an unknown value, validate the number and explicitly ensure it does not evaluate to zero.
- Validate all untrusted input from all sources, in particular verifying that it is not zero before dividing with it.
- Verify output of methods, calculations, dictionary lookups, and so on, and ensure it is not zero before dividing with the result.
- Ensure divide-by-zero errors are caught and handled appropriately.

Source Code Examples

Java

Divide by Zero

```java
public float getAverage(HttpServletRequest req) {
      int total = Integer.parseInt(req.getParameter("total"));
      int count = Integer.parseInt(req.getParameter("count"));

      return total / count;
}
```

Checked Division

```java
public float getAverage(HttpServletRequest req) {
      int total = Integer.parseInt(req.getParameter("total"));
      int count = Integer.parseInt(req.getParameter("count"));
```

```
     if (count > 0)
            return total / count;
     else
            return 0;
}
```

Wrong Size t Allocation

Risk

What might happen

Incorrect allocation of memory may result in unexpected behavior by either overwriting sections of memory with unexpected values. Under certain conditions where both an incorrect allocation of memory and the values being written can be controlled by an attacker, such an issue may result in execution of malicious code.

Cause

How does it happen

Some memory allocation functions require a size value to be provided as a parameter. The allocated size should be derived from the provided value, by providing the length value of the intended source, multiplied by the size of that length. Failure to perform the correct arithmetic to obtain the exact size of the value will likely result in the source overflowing its destination.

General Recommendations

How to avoid it

- Always perform the correct arithmetic to determine size.
- Specifically for memory allocation, calculate the allocation size from the allocation source:
    - Derive the size value from the length of intended source to determine the amount of units to be processed.
    - Always programmatically consider the size of the each unit and their conversion to memory units - for example, by using sizeof() on the unit's type.
    - Memory allocation should be a multiplication of the amount of units being written, times the size of each unit.

Source Code Examples

CPP

Allocating and Assigning Memory without Sizeof Arithmetic

```
int *ptr;
ptr = (int*)malloc(5);
for (int i = 0; i < 5; i++)
{
      ptr[i] = i * 2 + 1;
}
```

Allocating and Assigning Memory with Sizeof Arithmetic

```
int *ptr;
ptr = (int*)malloc(5 * sizeof(int));
```

```
for (int i = 0; i < 5; i++)
{
      ptr[i] = i * 2 + 1;
}
```

Incorrect Arithmetic of Multi-Byte String Allocation

```
wchar_t * dest;
dest = (wchar_t *)malloc(wcslen(source) + 1); // Would not crash for a short "source"
wcscpy((wchar_t *)dest, source);
wprintf(L"Dest: %s\r\n", dest);
```

Correct Arithmetic of Multi-Byte String Allocation

```
wchar_t * dest;
dest = (wchar_t *)malloc((wcslen(source) + 1) * sizeof(wchar_t));
wcscpy((wchar_t *)dest, source);
wprintf(L"Dest: %s\r\n", dest);
```

Char Overflow

Risk

What might happen

Assigning large data types into smaller data types, without proper checks and explicit casting, will lead to undefined behavior and unintentional effects, such as data corruption (e.g. value wraparound, wherein maximum values become minimum values); system crashes; infinite loops; logic errors, such as bypassing of security mechanisms; or even buffer overflows leading to arbitrary code execution.

Cause

How does it happen

This flaw can occur when implicitly casting numerical data types of a larger size, into a variable with a data type of a smaller size. This forces the program to discard some bits of information from the number. Depending on how the numerical data types are stored in memory, this is often the bits with the highest value, causing substantial corruption of the stored number. Alternatively, the sign bit of a signed integer could be lost, completely reversing the intention of the number.

General Recommendations

How to avoid it

- Avoid casting larger data types to smaller types.
- Prefer promoting the target variable to a large enough data type.
- If downcasting is necessary, always check that values are valid and in range of the target type, before casting

Source Code Examples

CPP

Unsafe Downsize Casting

```
int unsafe_addition(short op1, int op2) {

    // op2 gets forced from int into a short
    short total = op1 + op2;

    return total;
}
```

Safer Use of Proper Data Types

```
int safe_addition(short op1, int op2) {

    // total variable is of type int, the largest type that is needed
    int total = 0;

    // check if total will overflow available integer size
    if (INT_MAX - abs(op2) > op1)
    {
        total = op1 + op2;
    }
    else
    {
        // instead of overflow, saturate (but this is not always a good thing)
        total = INT_MAX
    }

    return total;
}
```

Short Overflow

Risk

What might happen

Assigning large data types into smaller data types, without proper checks and explicit casting, will lead to undefined behavior and unintentional effects, such as data corruption (e.g. value wraparound, wherein maximum values become minimum values); system crashes; infinite loops; logic errors, such as bypassing of security mechanisms; or even buffer overflows leading to arbitrary code execution.

Cause

How does it happen

This flaw can occur when implicitly casting numerical data types of a larger size, into a variable with a data type of a smaller size. This forces the program to discard some bits of information from the number. Depending on how the numerical data types are stored in memory, this is often the bits with the highest value, causing substantial corruption of the stored number. Alternatively, the sign bit of a signed integer could be lost, completely reversing the intention of the number.

General Recommendations

How to avoid it

- Avoid casting larger data types to smaller types.
- Prefer promoting the target variable to a large enough data type.
- If downcasting is necessary, always check that values are valid and in range of the target type, before casting

Source Code Examples

Dangerous Functions

Risk

What might happen

Use of dangerous functions may expose varying risks associated with each particular function, with potential impact of improper usage of these functions varying significantly. The presence of such functions indicates a flaw in code maintenance policies and adherence to secure coding practices, in a way that has allowed introducing known dangerous code into the application.

Cause

How does it happen

A dangerous function has been identified within the code. Functions are often deemed dangerous to use for numerous reasons, as there are different sets of vulnerabilities associated with usage of such functions. For example, some string copy and concatenation functions are vulnerable to Buffer Overflow, Memory Disclosure, Denial of Service and more. Use of these functions is not recommended.

General Recommendations

How to avoid it

- Deploy a secure and recommended alternative to any functions that were identified as dangerous.
    - If no secure alternative is found, conduct further researching and testing to identify whether current usage successfully sanitizes and verifies values, and thus successfully avoids the use-cases for whom the function is indeed dangerous
- Conduct a periodical review of methods that are in use, to ensure that all external libraries and built-in functions are up-to-date and whose use has not been excluded from best secure coding practices.

Source Code Examples

CPP

Buffer Overflow in gets()

```
int main()

{

      char buf[10];

      printf("Please enter your name: ");
      gets(buf); // veryveryverylongname
      if (buf == ACCEPTED_NAME)
      {
             // Do something
      }
      return 0;
}
```

Safe reading from user

```
int main()
{
       char buf[10];

       printf("Please enter your name: ");
       fgets(buf, sizeof(buf), stdin); //setting the amount of bytes to read
       if (buf == ACCEPTED_NAME)
       {
              //Do something
       }
       return 0;
}
```

Unsafe function for string copy

```
int main(int argc, char* argv[])
{
       char buf[10];
       strcpy(buf, argv[1]); // overflow occurs when len(argv[1]) > 10 bytes

       return 0;
}
```

Safe string copy

```
int main(int argc, char* argv[])
{
       char buf[10];
       strncpy(buf, argv[1], sizeof(buf));
       buf[9]= '\0'; //strncpy doesn't NULL terminates

       return 0;
}
```

Unsafe format string

```
int main(int argc, char* argv[])
{
       printf(argv[1]); // If argv[1] contains a format token, such as %s,%x or %d, will cause
an access violation
       return 0;
}
```

Safe format string

```
int main(int argc, char* argv[])
{
      printf("%s", argv[1]); // Second parameter is not a formattable string

      return 0;
}
```

Failure to Release Memory Before Removing Last Reference ('Memory Leak')

Weakness ID: 401 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The software does not sufficiently track and release allocated memory after it has been used, which slowly consumes remaining memory.

Extended Description

This is often triggered by improper handling of malformed data or unexpectedly interrupted sessions.

Terminology Notes

"memory leak" has sometimes been used to describe other kinds of issues, e.g. for information leaks in which the contents of memory are inadvertently leaked (CVE-2003-0400 is one such example of this terminology conflict).

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C

C++

Modes of Introduction

Memory leaks have two common and sometimes overlapping causes:

- Error conditions and other exceptional circumstances
- Confusion over which part of the program is responsible for freeing the memory

Common Consequences

Scope	Effect
Availability	Most memory leaks result in general software reliability problems, but if an attacker can intentionally trigger a memory leak, the attacker might be able to launch a denial of service attack (by crashing or hanging the program) or take advantage of other unexpected program behavior resulting from a low memory condition.

Likelihood of Exploit

Medium

Demonstrative Examples

Example 1

The following C function leaks a block of allocated memory if the call to read() fails to return the expected number of bytes:

(Bad Code)

Example Language: **C**

```
char* getBlock(int fd) {
char* buf = (char*) malloc(BLOCK_SIZE);
if (!buf) {
return NULL;
}
if (read(fd, buf, BLOCK_SIZE) != BLOCK_SIZE) {

return NULL;
}
return buf;
}
```

Example 2

Here the problem is that every time a connection is made, more memory is allocated. So if one just opened up more and more connections, eventually the machine would run out of memory.

(Bad Code)

Example Language: **C**

```
bar connection(){
foo = malloc(1024);
return foo;
}
endConnection(bar foo) {

free(foo);
}
int main() {

while(1) //thread 1
//On a connection
foo=connection(); //thread 2
//When the connection ends
endConnection(foo)
}
```

Observed Examples

Reference	Description
CVE-2005-3119	Memory leak because function does not free() an element of a data structure.
CVE-2004-0427	Memory leak when counter variable is not decremented.
CVE-2002-0574	Memory leak when counter variable is not decremented.
CVE-2005-3181	Kernel uses wrong function to release a data structure, preventing data from being properly tracked by other code.
CVE-2004-0222	Memory leak via unknown manipulations as part of protocol test suite.
CVE-2001-0136	Memory leak via a series of the same command.

Potential Mitigations

Pre-design: Use a language or compiler that performs automatic bounds checking.

Phase: Architecture and Design

Use an abstraction library to abstract away risky APIs. Not a complete solution.

Pre-design through Build: The Boehm-Demers-Weiser Garbage Collector or valgrind can be used to detect leaks in code. This is not a complete solution as it is not 100% effective.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	398	Indicator of Poor Code Quality	**Seven Pernicious Kingdoms (primary)700**
ChildOf	Category	399	Resource Management Errors	**Development Concepts (primary)699**
ChildOf	Category	633	Weaknesses that Affect Memory	**Resource-specific Weaknesses (primary)631**
ChildOf	Category	730	OWASP Top Ten 2004 Category A9 - Denial of Service	**Weaknesses in OWASP Top Ten (2004) (primary)711**
ChildOf	Weakness Base	772	Missing Release of Resource after Effective	**Research Concepts (primary)1000**

			Lifetime	
MemberOf	View	630	Weaknesses Examined by SAMATE	**Weaknesses Examined by SAMATE (primary)630**
CanFollow	Weakness Class	390	Detection of Error Condition Without Action	Research Concepts1000

Relationship Notes

This is often a resultant weakness due to improper handling of malformed data or early termination of sessions.

Affected Resources

- Memory

Functional Areas

- Memory management

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
PLOVER			Memory leak
7 Pernicious Kingdoms			Memory Leak
CLASP			Failure to deallocate data
OWASP Top Ten 2004	A9	CWE More Specific	Denial of Service

White Box Definitions

A weakness where the code path has:

1. start statement that allocates dynamically allocated memory resource

2. end statement that loses identity of the dynamically allocated memory resource creating situation where dynamically allocated memory resource is never relinquished

Where "loses" is defined through the following scenarios:

1. identity of the dynamic allocated memory resource never obtained

2. the statement assigns another value to the data element that stored the identity of the dynamically allocated memory resource and there are no aliases of that data element

3. identity of the dynamic allocated memory resource obtained but never passed on to function for memory resource release

4. the data element that stored the identity of the dynamically allocated resource has reached the end of its scope at the statement and there are no aliases of that data element

References

J. Whittaker and H. Thompson. "How to Break Software Security". Addison Wesley. 2003.

Content History

Submissions

Submission Date	Submitter	Organization	Source
	PLOVER		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-01		KDM Analytics	External
	added/updated white box definitions		
2008-08-15		Veracode	External
	Suggested OWASP Top Ten 2004 mapping		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, References, Relationship Notes, Taxonomy Mappings, Terminology Notes		
2008-10-14	CWE Content Team	MITRE	Internal
	updated Description		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Other Notes		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Name		
2009-07-17	KDM Analytics		External
	Improved the White Box Definition		

2009-07-27	CWE Content Team	MITRE	Internal
	updated White Box Definitions		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Modes of Introduction, Other Notes		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

Previous Entry Names

Change Date	Previous Entry Name
2008-04-11	Memory Leak
2009-05-27	Failure to Release Memory Before Removing Last Reference (aka 'Memory Leak')

BACK TO TOP

Use of Uninitialized Variable

Weakness ID: 457 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The code uses a variable that has not been initialized, leading to unpredictable or unintended results.

Extended Description

In some languages, such as C, an uninitialized variable contains contents of previously-used memory. An attacker can sometimes control or read these contents.

Time of Introduction

- Implementation

Applicable Platforms

Languages

C: *(Sometimes)*

C++: *(Sometimes)*

Perl: *(Often)*

All

Common Consequences

Scope	Effect
Availability Integrity	Initial variables usually contain junk, which can not be trusted for consistency. This can lead to denial of service conditions, or modify control flow in unexpected ways. In some cases, an attacker can "pre-initialize" the variable using previous actions, which might enable code execution. This can cause a race condition if a lock variable check passes when it should not.
Authorization	Strings that are not initialized are especially dangerous, since many functions expect a null at the end -- and only at the end -- of a string.

Likelihood of Exploit

High

Demonstrative Examples

Example 1

The following switch statement is intended to set the values of the variables aN and bN, but in the default case, the programmer has accidentally set the value of aN twice. As a result, bN will have an undefined value.

(Bad Code)

Example Language: **C**

```
switch (ctl) {
case -1:
aN = 0;
bN = 0;
break;
case 0:
aN = i;
bN = -i;
break;
case 1:
aN = i + NEXT_SZ;
bN = i - NEXT_SZ;
break;
default:
```

```
aN = -1;
aN = -1;
break;
}
repaint(aN, bN);
```

Most uninitialized variable issues result in general software reliability problems, but if attackers can intentionally trigger the use of an uninitialized variable, they might be able to launch a denial of service attack by crashing the program. Under the right circumstances, an attacker may be able to control the value of an uninitialized variable by affecting the values on the stack prior to the invocation of the function.

Example 2

Example Languages: **C++ and Java**

```
int foo;
void bar() {
if (foo==0)
/.../
/../
}
```

Observed Examples

Reference	Description
CVE-2008-0081	Uninitialized variable leads to code execution in popular desktop application.
CVE-2007-4682	Crafted input triggers dereference of an uninitialized object pointer.
CVE-2007-3468	Crafted audio file triggers crash when an uninitialized variable is used.
CVE-2007-2728	Uninitialized random seed variable used.

Potential Mitigations

Phase: Implementation

Assign all variables to an initial value.

Phase: Build and Compilation

Most compilers will complain about the use of uninitialized variables if warnings are turned on.

Phase: Requirements

The choice could be made to use a language that is not susceptible to these issues.

Phase: Architecture and Design

Mitigating technologies such as safe string libraries and container abstractions could be introduced.

Other Notes

Before variables are initialized, they generally contain junk data of what was left in the memory that the variable takes up. This data is very rarely useful, and it is generally advised to pre-initialize variables or set them to their first values early. If one forgets -- in the C language -- to initialize, for example a char *, many of the simple string libraries may often return incorrect results as they expect the null termination to be at the end of a string.

Stack variables in C and C++ are not initialized by default. Their initial values are determined by whatever happens to be in their location on the stack at the time the function is invoked. Programs should never use the value of an uninitialized variable. It is not uncommon for programmers to use an uninitialized variable in code that handles errors or other rare and exceptional circumstances. Uninitialized variable warnings can sometimes indicate the presence of a typographic error in the code.

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	Weakness Base	456	Missing Initialization	**Development Concepts (primary)699 Research Concepts**

				(primary)1000
MemberOf	View	630	Weaknesses Examined by SAMATE	**Weaknesses Examined by SAMATE (primary)630**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Uninitialized variable
7 Pernicious Kingdoms			Uninitialized Variable

White Box Definitions

A weakness where the code path has:

1. start statement that defines variable
2. end statement that accesses the variable
3. the code path does not contain a statement that assigns value to the variable

References

mercy. "Exploiting Uninitialized Data". Jan 2006. < http://www.felinemenace.org/~mercy/papers/UBehavior/UBehavior.zip>.

Microsoft Security Vulnerability Research & Defense. "MS08-014 : The Case of the Uninitialized Stack Variable Vulnerability". 2008-03-11. <http://blogs.technet.com/swi/archive/2008/03/11/the-case-of-the-uninitialized-stack-variable-vulnerability.aspx>.

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, Description, Relationships, Observed Example, Other Notes, References, Taxonomy Mappings		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Common Consequences, Demonstrative Examples, Potential Mitigations		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		

Previous Entry Names

Change Date	Previous Entry Name
2008-04-11	Uninitialized Variable

BACK TO TOP

Use of Zero Initialized Pointer

Risk

What might happen

A null pointer dereference is likely to cause a run-time exception, a crash, or other unexpected behavior.

Cause

How does it happen

Variables which are declared without being assigned will implicitly retain a null value until they are assigned. The null value can also be explicitly set to a variable, to ensure clear out its contents. Since null is not really a value, it may not have object variables and methods, and any attempt to access contents of a null object, instead of verifying it is set beforehand, will result in a null pointer dereference exception.

General Recommendations

How to avoid it

- For any variable that is created, ensure all logic flows between declaration and use assign a non-null value to the variable first.
- Enforce null checks on any received variable or object before it is dereferenced, to ensure it does not contain a null assigned to it elsewhere.
- Consider the need to assign null values in order to overwrite initialized variables. Consider reassigning or releasing these variables instead.

Source Code Examples

CPP

Explicit NULL Dereference

```
char * input = NULL;
printf("%s", input);
```

Implicit NULL Dereference

```
char * input;
printf("%s", input);
```

Java

Explicit Null Dereference

```
Object o = null;
out.println(o.getClass());
```

Stored Buffer Overflow boundcpy

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

CPP

Overflowing Buffers

```
const int BUFFER_SIZE = 10;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
{
    strcpy(buffer, inputString);
}
```

Checked Buffers

```
const int BUFFER_SIZE = 10;
const int MAX_INPUT_SIZE = 256;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
```

```
{
    if (strnlen(inputString, MAX_INPUT_SIZE) < sizeof(buffer))
    {
        strncpy(buffer, inputString, sizeof(buffer));
    }
}
```

Stored Buffer Overflow fgets

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Use of Function with Inconsistent Implementations

Weakness ID: 474 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The code uses a function that has inconsistent implementations across operating systems and versions, which might cause security-relevant portability problems.

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C: *(Often)*

PHP: *(Often)*

All

Potential Mitigations

Do not accept inconsistent behavior from the API specifications when the deviant behavior increase the risk level.

Other Notes

The behavior of functions in this category varies by operating system, and at times, even by operating system version. Implementation differences can include:

- Slight differences in the way parameters are interpreted leading to inconsistent results.

- Some implementations of the function carry significant security risks.

- The function might not be defined on all platforms.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	398	Indicator of Poor Code Quality	**Development Concepts (primary)699** **Seven Pernicious Kingdoms (primary)700** **Research Concepts (primary)1000**
ParentOf	Weakness Variant	589	Call to Non-ubiquitous API	**Research Concepts (primary)1000**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
7 Pernicious Kingdoms			Inconsistent Implementations

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	7 Pernicious Kingdoms		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Eric Dalci	Cigital	External
	updated Potential Mitigations, Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Relationships, Other Notes, Taxonomy Mappings		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2008-04-11	Inconsistent Implementations		

BACK TO TOP

Use of Insufficiently Random Values

Risk

What might happen

Random values are often used as a mechanism to prevent malicious users from guessing a value, such as a password, encryption key, or session identifier. Depending on what this random value is used for, an attacker would be able to predict the next numbers generated, or previously generated values. This could enable the attacker to hijack another user's session, impersonate another user, or crack an encryption key (depending on what the pseudo-random value was used for).

Cause

How does it happen

The application uses a weak method of generating pseudo-random values, such that other numbers could be determined from a relatively small sample size. Since the pseudo-random number generator used is designed for statistically uniform distribution of values, it is approximately deterministic. Thus, after collecting a few generated values (e.g. by creating a few individual sessions, and collecting the sessionids), it would be possible for an attacker to calculate another sessionid.

Specifically, if this pseudo-random value is used in any security context, such as passwords, keys, or secret identifiers, an attacker would be able to predict the next numbers generated, or previously generated values.

General Recommendations

How to avoid it

Generic Guidance:

- Whenever unpredicatable numbers are required in a security context, use a cryptographically strong random number generator, instead of a statistical pseudo-random generator.
- Use the cryptorandom generator that is built-in to your language or platform, and ensure it is securely seeded. Do not seed the generator with a weak, non-random seed. (In most cases, the default is securely random).
- Ensure you use a long enough random value, to make brute-force attacks unfeasible.

Specific Recommendations:

- Do not use the statistical pseudo-random number generator, use the cryptorandom generator instead. In Java, this is the SecureRandom class.

Source Code Examples

Java

Use of a weak pseudo-random number generator

```
Random random = new Random();

long sessNum = random.nextLong();

String sessionId = sessNum.toString();
```

Cryptographically secure random number generator

```
SecureRandom random = new SecureRandom();

byte sessBytes[] = new byte[32];

random.nextBytes(sessBytes);

String sessionId = new String(sessBytes);
```

Objc

Use of a weak pseudo-random number generator

```
long sessNum = rand();
NSString* sessionId = [NSString stringWithFormat:@"%ld", sessNum];
```

Cryptographically secure random number generator

```
UInt32 sessBytes;
SecRandomCopyBytes(kSecRandomDefault, sizeof(sessBytes), (uint8_t*)&sessBytes);

NSString* sessionId = [NSString stringWithFormat:@"%llu", sessBytes];
```

Swift

Use of a weak pseudo-random number generator

```
let sessNum = rand();
let sessionId = String(format:"%ld", sessNum)
```

Cryptographically secure random number generator

```
var sessBytes: UInt32 = 0
withUnsafeMutablePointer(&sessBytes, { (sessBytesPointer) -> Void in
    let castedPointer = unsafeBitCast(sessBytesPointer, UnsafeMutablePointer<UInt8>.self)
    SecRandomCopyBytes(kSecRandomDefault, sizeof(UInt32), castedPointer)
})

let sessionId = String(format:"%llu", sessBytes)
```

Unchecked Return Value

Risk

What might happen

A program that does not check function return values could cause the application to enter an undefined state. This could lead to unexpected behavior and unintended consequences, including inconsistent data, system crashes or other error-based exploits.

Cause

How does it happen

The application calls a system function, but does not receive or check the result of this funciton. These functions often return error codes in the result, or share other status codes with it's caller. The application simply ignores this result value, losing this vital information.

General Recommendations

How to avoid it

- Always check the result of any called function that returns a value, and verify the result is an expected value.

- Ensure the calling function responds to all possible return values.

- Expect runtime errors and handle them gracefully. Explicitly define a mechanism for handling unexpected errors.

Source Code Examples

CPP

Unchecked Memory Allocation

```
buff = (char*) malloc(size);
strncpy(buff, source, size);
```

Safer Memory Allocation

```
buff = (char*) malloc(size+1);
if (buff==NULL) exit(1);

strncpy(buff, source, size);
buff[size] = '\0';
```

Use of sizeof() on a Pointer Type

Weakness ID: 467 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The code calls sizeof() on a malloced pointer type, which always returns the wordsize/8. This can produce an unexpected result if the programmer intended to determine how much memory has been allocated.

Time of Introduction

- Implementation

Applicable Platforms

Languages

C

C++

Common Consequences

Scope	Effect
Integrity	This error can often cause one to allocate a buffer that is much smaller than what is needed, leading to resultant weaknesses such as buffer overflows.

Likelihood of Exploit

High

Demonstrative Examples

Example 1

Care should be taken to ensure sizeof returns the size of the data structure itself, and not the size of the pointer to the data structure.

In this example, sizeof(foo) returns the size of the pointer.

(Bad Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(foo));
```

In this example, sizeof(*foo) returns the size of the data structure and not the size of the pointer.

(Good Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(*foo));
```

Example 2

This example defines a fixed username and password. The AuthenticateUser() function is intended to accept a username and a password from an untrusted user, and check to ensure that it matches the username and password. If the username and password match, AuthenticateUser() is intended to indicate that authentication succeeded.

(Bad Code)

```
/* Ignore CWE-259 (hard-coded password) and CWE-309 (use of password system for authentication) for this example. */

char *username = "admin";
char *pass = "password";

int AuthenticateUser(char *inUser, char *inPass) {
```

```
printf("Sizeof username = %d\n", sizeof(username));
printf("Sizeof pass = %d\n", sizeof(pass));

if (strncmp(username, inUser, sizeof(username))) {
printf("Auth failure of username using sizeof\n");
return(AUTH_FAIL);
}
/* Because of CWE-467, the sizeof returns 4 on many platforms and architectures. */
if (! strncmp(pass, inPass, sizeof(pass))) {
printf("Auth success of password using sizeof\n");
return(AUTH_SUCCESS);
}
else {
printf("Auth fail of password using sizeof\n");
return(AUTH_FAIL);
}
}

int main (int argc, char **argv)
{
int authResult;

if (argc < 3) {
ExitError("Usage: Provide a username and password");
}
authResult = AuthenticateUser(argv[1], argv[2]);
if (authResult != AUTH_SUCCESS) {
ExitError("Authentication failed");
}
else {
DoAuthenticatedTask(argv[1]);
}
}
```

In AuthenticateUser(), because sizeof() is applied to a parameter with an array type, the sizeof() call might return 4 on many modern architectures. As a result, the strncmp() call only checks the first four characters of the input password, resulting in a partial comparison (CWE-187), leading to improper authentication (CWE-287).

Because of the partial comparison, any of these passwords would still cause authentication to succeed for the "admin" user:

(Attack)

```
pass5
passABCDEFGH
passWORD
```

Because only 4 characters are checked, this significantly reduces the search space for an attacker, making brute force attacks more feasible.

The same problem also applies to the username, so values such as "adminXYZ" and "administrator" will succeed for the username.

Potential Mitigations

Phase: Implementation

Use expressions such as "sizeof(*pointer)" instead of "sizeof(pointer)", unless you intend to run sizeof() on a pointer type to gain some platform independence or if you are allocating a variable on the stack.

Other Notes

The use of sizeof() on a pointer can sometimes generate useful information. An obvious case is to find out the wordsize on a platform. More often than not, the appearance of sizeof(pointer) indicates a bug.

Weakness Ordinalities

Ordinality	Description
Primary	*(where the weakness exists independent of other weaknesses)*

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	465	Pointer Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	682	Incorrect Calculation	**Research Concepts (primary)1000**
ChildOf	Category	737	CERT C Secure Coding Section 03 - Expressions (EXP)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
CanPrecede	Weakness Base	131	Incorrect Calculation of Buffer Size	Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Use of sizeof() on a pointer type
CERT C Secure Coding	ARR01-C		Do not apply the sizeof operator to a pointer when taking the size of an array
CERT C Secure Coding	EXP01-C		Do not take the size of a pointer to determine the size of the pointed-to type

White Box Definitions

A weakness where code path has:

1. end statement that passes an identity of a dynamically allocated memory resource to a sizeof operator
2. start statement that allocates the dynamically allocated memory resource

References

Robert Seacord. "EXP01-A. Do not take the sizeof a pointer to determine the size of a type". <https://www.securecoding.cert.org/confluence/display/seccode/EXP01-A.+Do+not+take+the+sizeof+a+pointer+to+determine+the+size+of+a+type>.

Content History

Submissions

Submission Date	Submitter	Organization	Source
	CLASP		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-01		KDM Analytics	External
	added/updated white box definitions		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

BACK TO TOP

Potential Off by One Error in Loops

Risk

What might happen

An off by one error may result in overwriting or over-reading of unintended memory; in most cases, this can result in unexpected behavior and even application crashes. In other cases, where allocation can be controlled by an attacker, a combination of variable assignment and an off by one error can result in execution of malicious code.

Cause

How does it happen

Often when designating variables to memory, a calculation error may occur when determining size or length that is off by one.

For example in loops, when allocating an array of size 2, its cells are counted as 0,1 - therefore, if a For loop iterator on the array is incorrectly set with the start condition i=0 and the continuation condition i<=2, three cells will be accessed instead of 2, and an attempt will be made to write or read cell [2], which was not originally allocated, resulting in potential corruption of memory outside the bounds of the originally assigned array.

Another example occurs when a null-byte terminated string, in the form of a character array, is copied without its terminating null-byte. Without the null-byte, the string representation is unterminated, resulting in certain functions to over-read memory as they expect the missing null terminator.

General Recommendations

How to avoid it

- Always ensure that a given iteration boundary is correct:
    - With array iterations, consider that arrays begin with cell 0 and end with cell n-1, for a size n array.
    - With character arrays and null-byte terminated string representations, consider that the null byte is required and should not be overwritten or ignored; ensure functions in use are not vulnerable to off-by-one, specifically for instances where null-bytes are automatically appended after the buffer, instead of in place of its last character.
- Where possible, use safe functions that manage memory and are not prone to off-by-one errors.

Source Code Examples

CPP

Off-By-One in For Loop

```
int *ptr;
ptr = (int*)malloc(5 * sizeof(int));
for (int i = 0; i <= 5; i++)
{
      ptr[i] = i * 2 + 1; // ptr[5] will be set, but is out of bounds
}
```

Proper Iteration in For Loop

```
int *ptr;
ptr = (int*)malloc(5 * sizeof(int));
for (int i = 0; i < 5; i++)
{
      ptr[i] = i * 2 + 1; // ptr[0-4] are well defined
}
```

Off-By-One in strncat

```
strncat(buf, input, sizeof(buf) - strlen(buf)); // actual value should be sizeof(buf)-
strlen(buf)-1 - this form will overwrite the terminating nullbyte
```

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 malloc

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Potential Precision Problem

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Heuristic Buffer Overflow malloc

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Indicator of Poor Code Quality

Weakness ID: 398 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The code has features that do not directly introduce a weakness or vulnerability, but indicate that the product has not been carefully developed or maintained.

Extended Description

Programs are more likely to be secure when good development practices are followed. If a program is complex, difficult to maintain, not portable, or shows evidence of neglect, then there is a higher likelihood that weaknesses are buried in the code.

Time of Introduction

- Architecture and Design
- Implementation

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	18	Source Code	**Development Concepts (primary)699**
ChildOf	Weakness Class	710	Coding Standards Violation	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	107	Struts: Unused Validation Form	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	110	Struts: Validator Without Form Field	**Research Concepts (primary)1000**
ParentOf	Category	399	Resource Management Errors	**Development Concepts (primary)699**
ParentOf	Weakness Base	401	Failure to Release Memory Before Removing Last Reference ('Memory Leak')	**Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Base	404	Improper Resource Shutdown or Release	Development Concepts699 **Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Variant	415	Double Free	**Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Base	416	Use After Free	**Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Variant	457	Use of Uninitialized Variable	**Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Base	474	Use of Function with Inconsistent Implementations	**Development Concepts (primary)699** **Seven Pernicious Kingdoms (primary)700** **Research Concepts (primary)1000**
ParentOf	Weakness Base	475	Undefined Behavior for Input to API	**Development Concepts (primary)699** **Seven Pernicious Kingdoms (primary)700**
ParentOf	Weakness Base	476	NULL Pointer	**Development**

			Dereference	**Concepts (primary)699** **Seven Pernicious Kingdoms (primary)700** **Research Concepts (primary)1000**
ParentOf	Weakness Base	477	Use of Obsolete Functions	**Development Concepts (primary)699** **Seven Pernicious Kingdoms (primary)700** **Research Concepts (primary)1000**
ParentOf	Weakness Variant	478	Missing Default Case in Switch Statement	**Development Concepts (primary)699**
ParentOf	Weakness Variant	479	Unsafe Function Call from a Signal Handler	**Development Concepts (primary)699**
ParentOf	Weakness Variant	483	Incorrect Block Delimitation	**Development Concepts (primary)699**
ParentOf	Weakness Base	484	Omitted Break Statement in Switch	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Variant	546	Suspicious Comment	**Development Concepts (primary)699** **Research Concepts (primary)1000**
ParentOf	Weakness Variant	547	Use of Hard-coded, Security-relevant Constants	**Development Concepts (primary)699** **Research Concepts (primary)1000**
ParentOf	Weakness Variant	561	Dead Code	**Development Concepts (primary)699** **Research Concepts (primary)1000**
ParentOf	Weakness Base	562	Return of Stack Variable Address	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Variant	563	Unused Variable	**Development Concepts (primary)699** **Research Concepts (primary)1000**
ParentOf	Category	569	Expression Issues	**Development Concepts (primary)699**
ParentOf	Weakness Variant	585	Empty Synchronized Block	**Development Concepts (primary)699** **Research Concepts (primary)1000**
ParentOf	Weakness Variant	586	Explicit Call to Finalize()	**Development Concepts (primary)699**
ParentOf	Weakness Variant	617	Reachable Assertion	**Development Concepts (primary)699**
ParentOf	Weakness Base	676	Use of Potentially Dangerous Function	**Development Concepts (primary)699** **Research Concepts (primary)1000**
MemberOf	View	700	Seven Pernicious Kingdoms	**Seven Pernicious Kingdoms (primary)700**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name

7 Pernicious Kingdoms			Code Quality

Content History

Submissions

Submission Date	Submitter	Organization	Source
	7 Pernicious Kingdoms		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Description, Relationships, Taxonomy Mappings		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Relationships		

Previous Entry Names

Change Date	Previous Entry Name
2008-04-11	Code Quality

BACK TO TOP

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 updated Time of Introduction	Cigital	External
2008-08-01	added/updated white box definitions	KDM Analytics	External
2008-09-08	CWE Content Team updated Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities	MITRE	Internal
2008-11-24	CWE Content Team updated Relationships, Taxonomy Mappings	MITRE	Internal
2009-03-10	CWE Content Team updated Demonstrative Examples	MITRE	Internal
2009-12-28	CWE Content Team updated Demonstrative Examples	MITRE	Internal
2010-02-16	CWE Content Team updated Relationships	MITRE	Internal

BACK TO TOP

Improper Validation of Array Index

Weakness ID: 129 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The product uses untrusted input when calculating or using an array index, but the product does not validate or incorrectly validates the index to ensure the index references a valid position within the array.

Alternate Terms

out-of-bounds array index

index-out-of-range

array index underflow

Time of Introduction

- Implementation

Applicable Platforms

Languages

C: *(Often)*

C++: *(Often)*

Language-independent

Common Consequences

Scope	Effect
Integrity Availability	Unchecked array indexing will very likely result in the corruption of relevant memory and perhaps instructions, leading to a crash, if the values are outside of the valid memory area.
Integrity	If the memory corrupted is data, rather than instructions, the system will continue to function with improper values.
Confidentiality Integrity	Unchecked array indexing can also trigger out-of-bounds read or write operations, or operations on the wrong objects; i.e., "buffer overflows" are not always the result. This may result in the exposure or modification of sensitive data.
Integrity	If the memory accessible by the attacker can be effectively controlled, it may be possible to execute arbitrary code, as with a standard buffer overflow and possibly without the use of large inputs if a precise index can be controlled.
Integrity Availability Confidentiality	A single fault could allow either an overflow (CWE-788) or underflow (CWE-786) of the array index. What happens next will depend on the type of operation being performed out of bounds, but can expose sensitive information, cause a system crash, or possibly lead to arbitrary code execution.

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

This weakness can often be detected using automated static analysis tools. Many modern tools use data flow analysis or constraint-based techniques to minimize the number of false positives.

Automated static analysis generally does not account for environmental considerations when reporting out-of-bounds memory operations. This can make it difficult for users to determine which warnings should be investigated first. For example, an analysis tool might report array index errors that originate from command line arguments in a program that is not expected to run with setuid or other special privileges.

Effectiveness: High

This is not a perfect solution, since 100% accuracy and coverage are not feasible.

Automated Dynamic Analysis

This weakness can be detected using dynamic tools and techniques that interact with the software using large test suites with many diverse inputs, such as fuzz testing (fuzzing), robustness testing, and fault injection. The software's operation may slow down, but it should not become unstable, crash, or generate incorrect results.

Black Box

Black box methods might not get the needed code coverage within limited time constraints, and a dynamic test might not produce any noticeable side effects even if it is successful.

Demonstrative Examples

Example 1

The following C/C++ example retrieves the sizes of messages for a pop3 mail server. The message sizes are retrieved from a socket that returns in a buffer the message number and the message size, the message number (num) and size (size) are extracted from the buffer and the message size is placed into an array using the message number for the array index.

(Bad Code)

Example Language: **C**

```
/* capture the sizes of all messages */
int getsizes(int sock, int count, int *sizes) {
...
char buf[BUFFER_SIZE];
int ok;
int num, size;

// read values from socket and added to sizes array
while ((ok = gen_recv(sock, buf, sizeof(buf))) == 0)
{

// continue read from socket until buf only contains '.'
if (DOTLINE(buf))
break;
else if (sscanf(buf, "%d %d", &num, &size) == 2)
sizes[num - 1] = size;
}
...
}
```

In this example the message number retrieved from the buffer could be a value that is outside the allowable range of indices for the array and could possibly be a negative number. Without proper validation of the value to be used for the array index an array overflow could occur and could potentially lead to unauthorized access to memory addresses and system crashes. The value of the array index should be validated to ensure that it is within the allowable range of indices for the array as in the following code.

(Good Code)

Example Language: **C**

```
/* capture the sizes of all messages */
int getsizes(int sock, int count, int *sizes) {
...
char buf[BUFFER_SIZE];
int ok;
int num, size;

// read values from socket and added to sizes array
while ((ok = gen_recv(sock, buf, sizeof(buf))) == 0)
{

// continue read from socket until buf only contains '.'
if (DOTLINE(buf))
break;
else if (sscanf(buf, "%d %d", &num, &size) == 2) {
if (num > 0 && num <= (unsigned)count)
sizes[num - 1] = size;
else
/* warn about possible attempt to induce buffer overflow */
report(stderr, "Warning: ignoring bogus data for message sizes returned by server.\n");
}
}
...
}
```

Example 2

In the code snippet below, an unchecked integer value is used to reference an object in an array.

(Bad Code)

Example Language: **Java**

```
public String getValue(int index) {
return array[index];
}
```

If index is outside of the range of the array, this may result in an ArrayIndexOutOfBounds Exception being raised.

Example 3

In the following Java example the method displayProductSummary is called from a Web service servlet to retrieve product summary information for display to the user. The servlet obtains the integer value of the product number from the user and passes it to the displayProductSummary method. The displayProductSummary method passes the integer value of the product number to the getProductSummary method which obtains the product summary from the array object containing the project summaries using the integer value of the product number as the array index.

(Bad Code)

Example Language: **Java**

```
// Method called from servlet to obtain product information
public String displayProductSummary(int index) {

String productSummary = new String("");

try {
String productSummary = getProductSummary(index);

} catch (Exception ex) {...}

return productSummary;
}

public String getProductSummary(int index) {
return products[index];
}
```

In this example the integer value used as the array index that is provided by the user may be outside the allowable range of indices for the array which may provide unexpected results or may comes the application to fail. The integer value used for the array index should be validated to ensure that it is within the allowable range of indices for the array as in the following code.

(Good Code)

Example Language: **Java**

```
// Method called from servlet to obtain product information
public String displayProductSummary(int index) {

String productSummary = new String("");

try {
String productSummary = getProductSummary(index);

} catch (Exception ex) {...}

return productSummary;
}

public String getProductSummary(int index) {
String productSummary = "";

if ((index >= 0) && (index < MAX_PRODUCTS)) {
productSummary = products[index];
}
else {
System.err.println("index is out of bounds");
throw new IndexOutOfBoundsException();
}

return productSummary;
}
```

An alternative in Java would be to use one of the collection objects such as ArrayList that will automatically generate an exception if an attempt is made to access an array index that is out of bounds.

(Good Code)

Example Language: **Java**

```
ArrayList productArray = new ArrayList(MAX_PRODUCTS);
...
try {
productSummary = (String) productArray.get(index);
} catch (IndexOutOfBoundsException ex) {...}
```

Observed Examples

Reference	Description
CVE-2005-0369	large ID in packet used as array index
CVE-2001-1009	negative array index as argument to POP LIST command
CVE-2003-0721	Integer signedness error leads to negative array index
CVE-2004-1189	product does not properly track a count and a maximum number, which can lead to resultant array index overflow.
CVE-2007-5756	chain: device driver for packet-capturing software allows access to an unintended IOCTL with resultant array index error.

Potential Mitigations

Phase: Architecture and Design

Strategies: Input Validation; Libraries or Frameworks

Use an input validation framework such as Struts or the OWASP ESAPI Validation API. If you use Struts, be mindful of weaknesses covered by the CWE-101 category.

Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.

Even though client-side checks provide minimal benefits with respect to server-side security, they are still useful. First, they can support intrusion detection. If the server receives input that should have been rejected by the client, then it may be an indication of an attack. Second, client-side error-checking can provide helpful feedback to the user about the expectations for valid input. Third, there may be a reduction in server-side processing time for accidental input errors, although this is typically a small savings.

Phase: Requirements

Strategy: Language Selection

Use a language with features that can automatically mitigate or eliminate out-of-bounds indexing errors.

For example, Ada allows the programmer to constrain the values of a variable and languages such as Java and Ruby will allow the programmer to handle exceptions when an out-of-bounds index is accessed.

Phase: Implementation

Strategy: Input Validation

Assume all input is malicious. Use an "accept known good" input validation strategy (i.e., use a whitelist). Reject any input that does not strictly conform to specifications, or transform it into something that does. Use a blacklist to reject any unexpected inputs and detect potential attacks.

When accessing a user-controlled array index, use a stringent range of values that are within the target array. Make sure that you do not allow negative values to be used. That is, verify the minimum as well as the maximum of the range of acceptable values.

Phase: Implementation

Be especially careful to validate your input when you invoke code that crosses language boundaries, such as from an interpreted language to native code. This could create an unexpected interaction between the language boundaries. Ensure that you are not violating any of the expectations of the language with which you are interfacing. For example, even though Java may not be susceptible to buffer overflows, providing a large argument in a call to native code might trigger an overflow.

Weakness Ordinalities

Ordinality	Description
Resultant	The most common condition situation leading to unchecked array indexing is the use of loop index variables as buffer indexes. If the end condition for the loop is subject to a flaw, the index can grow or shrink unbounded, therefore causing a buffer overflow or underflow. Another common situation leading to this condition is the use of a function's return value, or the resulting value of a calculation directly as an index in to a buffer.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	20	Improper Input Validation	**Development Concepts (primary)699** **Research Concepts (primary)1000**
ChildOf	Category	189	Numeric Errors	Development Concepts699
ChildOf	Category	633	Weaknesses that Affect Memory	**Resource-specific Weaknesses (primary)631**
ChildOf	Category	738	CERT C Secure Coding Section 04 - Integers (INT)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
ChildOf	Category	802	2010 Top 25 - Risky Resource Management	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
CanPrecede	Weakness Class	119	Failure to Constrain Operations within the Bounds of a Memory Buffer	Research Concepts1000
CanPrecede	Weakness Variant	789	Uncontrolled Memory Allocation	Research Concepts1000
PeerOf	Weakness Base	124	Buffer Underwrite ('Buffer Underflow')	Research Concepts1000

Theoretical Notes

An improperly validated array index might lead directly to the always-incorrect behavior of "access of array using out-of-bounds index."

Affected Resources

- Memory

f Causal Nature

Explicit

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Unchecked array indexing
PLOVER			INDEX - Array index overflow
CERT C Secure Coding	ARR00-C		Understand how arrays work
CERT C Secure Coding	ARR30-C		Guarantee that array indices are within the valid range
CERT C Secure Coding	ARR38-C		Do not add or subtract an integer to a pointer if the resulting value does not refer to a valid array element
CERT C Secure Coding	INT32-C		Ensure that operations on signed integers do not result in overflow

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
100	Overflow Buffers	

References

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 5, "Array Indexing Errors" Page 144. 2nd Edition. Microsoft. 2002.

Content History

Submissions

Submission Date	Submitter	Organization	Source
	CLASP		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Sean Eidemiller	Cigital	External
	added/updated demonstrative examples		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Common Consequences		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Description, Name, Relationships		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Observed Examples, Other Notes, Potential Mitigations, Theoretical Notes, Weakness Ordinalities		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Demonstrative Examples, Detection Factors, Likelihood of Exploit, Potential Mitigations, References, Related Attack Patterns, Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Related Attack Patterns		

Previous Entry Names

Change Date	Previous Entry Name
2009-10-29	Unchecked Array Indexing

BACK TO TOP

Improper Access Control (Authorization)

Weakness ID: 285 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software does not perform or incorrectly performs access control checks across all potential execution paths.

Extended Description

When access control checks are not applied consistently - or not at all - users are able to access data or perform actions that they should not be allowed to perform. This can lead to a wide range of problems, including information leaks, denial of service, and arbitrary code execution.

Alternate Terms

AuthZ:	"AuthZ" is typically used as an abbreviation of "authorization" within the web application security community. It is also distinct from "AuthC," which is an abbreviation of "authentication." The use of "Auth" as an abbreviation is discouraged, since it could be used for either authentication or authorization.

Time of Introduction

- Architecture and Design
- Implementation
- Operation

Applicable Platforms

Languages

Language-independent

Technology Classes

Web-Server: *(Often)*

Database-Server: *(Often)*

Modes of Introduction

A developer may introduce authorization weaknesses because of a lack of understanding about the underlying technologies. For example, a developer may assume that attackers cannot modify certain inputs such as headers or cookies.

Authorization weaknesses may arise when a single-user application is ported to a multi-user environment.

Common Consequences

Scope	Effect
Confidentiality	An attacker could read sensitive data, either by reading the data directly from a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to read the data.
Integrity	An attacker could modify sensitive data, either by writing the data directly to a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to write the data.
Integrity	An attacker could gain privileges by modifying or reading critical data directly, or by accessing insufficiently-protected, privileged functionality.

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

Automated static analysis is useful for detecting commonly-used idioms for authorization. A tool may be able to analyze related configuration files, such as .htaccess in Apache web servers, or detect the usage of commonly-used authorization libraries.

Generally, automated static analysis tools have difficulty detecting custom authorization schemes. In addition, the software's design may include some functionality that is accessible to any user and does not require an authorization check; an automated technique that detects the absence of authorization may report false positives.

Effectiveness: Limited

Automated Dynamic Analysis

Automated dynamic analysis may find many or all possible interfaces that do not require authorization, but manual analysis is required to determine if the lack of authorization violates business logic

Manual Analysis

This weakness can be detected using tools and techniques that require manual (human) analysis, such as penetration testing, threat modeling, and interactive tools that allow the tester to record and modify an active session.

Specifically, manual static analysis is useful for evaluating the correctness of custom authorization mechanisms.

Effectiveness: Moderate

These may be more effective than strictly automated techniques. This is especially the case with weaknesses that are related to design and business rules. However, manual efforts might not achieve desired code coverage within limited time constraints.

Demonstrative Examples

Example 1

The following program could be part of a bulletin board system that allows users to send private messages to each other. This program intends to authenticate the user before deciding whether a private message should be displayed. Assume that LookupMessageObject() ensures that the $id argument is numeric, constructs a filename based on that id, and reads the message details from that file. Also assume that the program stores all private messages for all users in the same directory.

(Bad Code)

Example Language: **Perl**

```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

```java
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