proxydroid Scan Report

Project Name	proxydroid
Scan Start	Friday, June 21, 2024 10:42:05 PM
Preset	Checkmarx Default
Scan Time	00h:04m:35s
Lines Of Code Scanned	4013
Files Scanned	4
Report Creation Time	Friday, June 21, 2024 10:47:30 PM
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046
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

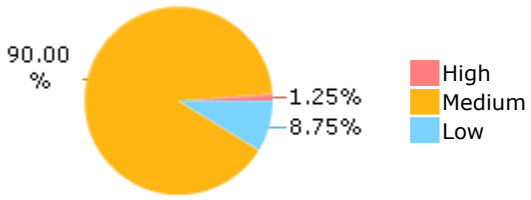
Results Limit

Results limit per query was set to 50

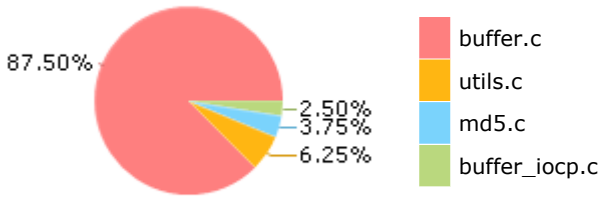
Selected Queries

Selected queries are listed in Result Summary

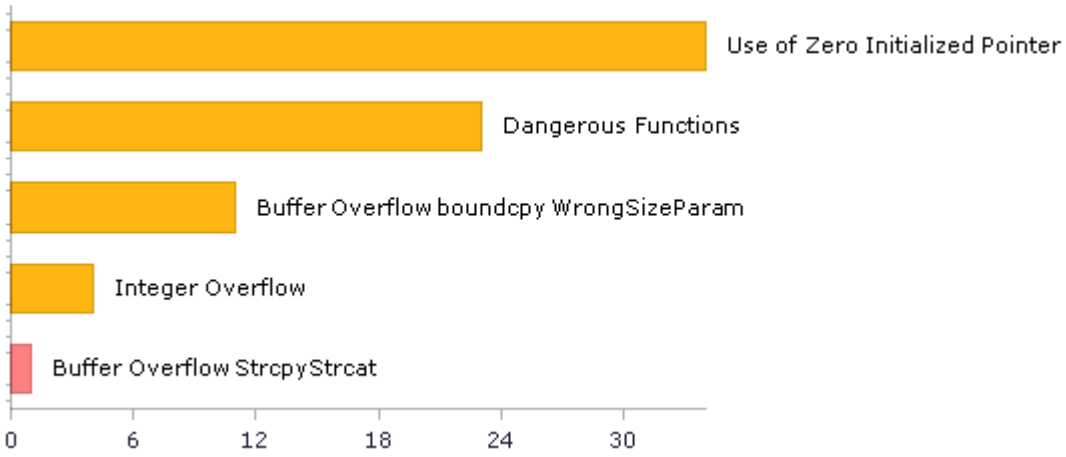
Result Summary



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	16	15
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	2	2
A3-Sensitive Data Exposure	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	0	0
A4-XML External Entities (XXE)	App. Specific	AVERAGE	COMMON	EASY	SEVERE	App. Specific	0	0
A5-Broken Access Control*	App. Specific	AVERAGE	COMMON	AVERAGE	SEVERE	App. Specific	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	23	23
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	23	23
A10-Unvalidated Redirects and Forwards	USERS BROWSERS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0

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

Scan Summary - PCI DSS v3.2

Category	Issues Found	Best Fix Locations
PCI DSS (3.2) - 6.5.1 - Injection flaws - particularly SQL injection	0	0
PCI DSS (3.2) - 6.5.2 - Buffer overflows	18	17
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.	0	0
Audit And Accountability*	Organizations must: (i) create, protect, and retain information system audit records to the extent needed to enable the monitoring, analysis, investigation, and reporting of unlawful, unauthorized, or inappropriate information system activity; and (ii) ensure that the actions of individual information system users can be uniquely traced to those users so they can be held accountable for their actions.	0	0
Configuration Management	Organizations must: (i) establish and maintain baseline configurations and inventories of organizational information systems (including hardware, software, firmware, and documentation) throughout the respective system development life cycles; and (ii) establish and enforce security configuration settings for information technology products employed in organizational information systems.	0	0
Identification And Authentication*	Organizations must identify information system users, processes acting on behalf of users, or devices and authenticate (or verify) the identities of those users, processes, or devices, as a prerequisite to allowing access to organizational information systems.	2	2
Media Protection	Organizations must: (i) protect information system media, both paper and digital; (ii) limit access to information on information system media to authorized users; and (iii) sanitize or destroy information system media before disposal or release for reuse.	0	0
System And Communications Protection	Organizations must: (i) monitor, control, and protect organizational communications (i.e., information transmitted or received by organizational information systems) at the external boundaries and key internal boundaries of the information systems; and (ii) employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational information systems.	0	0
System And Information Integrity	Organizations must: (i) identify, report, and correct information and information system flaws in a timely manner; (ii) provide protection from malicious code at appropriate locations within organizational information systems; and (iii) monitor information system security alerts and advisories and take appropriate actions in response.	4	4

***** 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)	2	2
AC-4 Information Flow Enforcement (P1)	0	0
AC-6 Least Privilege (P1)	0	0
AU-9 Protection of Audit Information (P1)	0	0
CM-6 Configuration Settings (P2)	0	0
IA-5 Authenticator Management (P1)	0	0
IA-6 Authenticator Feedback (P2)	0	0
IA-8 Identification and Authentication (Non-Organizational Users) (P1)	0	0
SC-12 Cryptographic Key Establishment and Management (P1)	0	0
SC-13 Cryptographic Protection (P1)	0	0
SC-17 Public Key Infrastructure Certificates (P1)	0	0
SC-18 Mobile Code (P2)	0	0
SC-23 Session Authenticity (P1)*	0	0
SC-28 Protection of Information at Rest (P1)	0	0
SC-4 Information in Shared Resources (P1)	0	0
SC-5 Denial of Service Protection (P1)*	36	7
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*	7	6
SI-11 Error Handling (P2)*	1	1
SI-15 Information Output Filtering (P0)	0	0
SI-16 Memory Protection (P1)	0	0

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

Scan Summary - OWASP Mobile Top 10 2016

Category	Description	Issues Found	Best Fix Locations
M1-Improper Platform Usage	This category covers misuse of a platform feature or failure to use platform security controls. It might include Android intents, platform permissions, misuse of TouchID, the Keychain, or some other security control that is part of the mobile operating system. There are several ways that mobile apps can experience this risk.	0	0
M2-Insecure Data Storage	This category covers insecure data storage and unintended data leakage.	0	0
M3-Insecure Communication	This category covers poor handshaking, incorrect SSL versions, weak negotiation, cleartext communication of sensitive assets, etc.	0	0
M4-Insecure Authentication	This category captures notions of authenticating the end user or bad session management. This can include: -Failing to identify the user at all when that should be required -Failure to maintain the user's identity when it is required -Weaknesses in session management	0	0
M5-Insufficient Cryptography	The code applies cryptography to a sensitive information asset. However, the cryptography is insufficient in some way. Note that anything and everything related to TLS or SSL goes in M3. Also, if the app fails to use cryptography at all when it should, that probably belongs in M2. This category is for issues where cryptography was attempted, but it wasnt done correctly.	0	0
M6-Insecure Authorization	This is a category to capture any failures in authorization (e.g., authorization decisions in the client side, forced browsing, etc.). It is distinct from authentication issues (e.g., device enrolment, user identification, etc.). If the app does not authenticate users at all in a situation where it should (e.g., granting anonymous access to some resource or service when authenticated and authorized access is required), then that is an authentication failure not an authorization failure.	0	0
M7-Client Code Quality	This category is the catch-all for code-level implementation problems in the mobile client. That's distinct from server-side coding mistakes. This would capture things like buffer overflows, format string vulnerabilities, and various other code-level mistakes where the solution is to rewrite some code that's running on the mobile device.	0	0
M8-Code Tampering	This category covers binary patching, local resource modification, method hooking, method swizzling, and dynamic memory modification. Once the application is delivered to the mobile device, the code and data resources are resident there. An attacker can either directly modify the code, change the contents of memory dynamically, change or replace the system APIs that the application uses, or	0	0

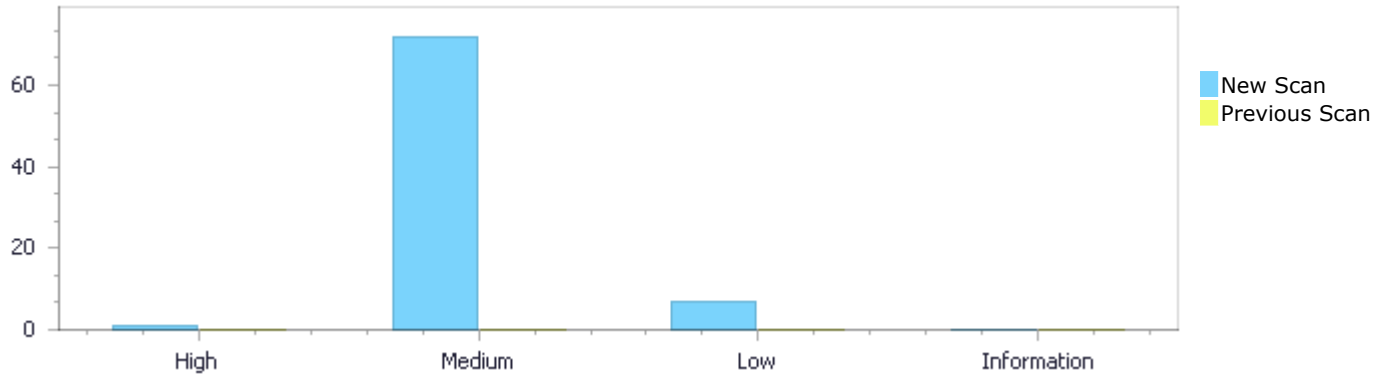
	modify the application's data and resources. This can provide the attacker a direct method of subverting the intended use of the software for personal or monetary gain.		
M9-Reverse Engineering	This category includes analysis of the final core binary to determine its source code, libraries, algorithms, and other assets. Software such as IDA Pro, Hopper, otool, and other binary inspection tools give the attacker insight into the inner workings of the application. This may be used to exploit other nascent vulnerabilities in the application, as well as revealing information about back end servers, cryptographic constants and ciphers, and intellectual property.	0	0
M10-Extraneous Functionality	Often, developers include hidden backdoor functionality or other internal development security controls that are not intended to be released into a production environment. For example, a developer may accidentally include a password as a comment in a hybrid app. Another example includes disabling of 2-factor authentication during testing.	0	0

Scan Summary - Custom

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

Results Distribution By Status First scan of the project

	High	Medium	Low	Information	Total
New Issues	1	72	7	0	80
Recurrent Issues	0	0	0	0	0
Total	1	72	7	0	80
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	1	72	7	0	80
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	1	72	7	0	80

Result Summary

Vulnerability Type	Occurrences	Severity
Buffer Overflow StrcpyStrcat	1	High
Use of Zero Initialized Pointer	34	Medium
Dangerous Functions	23	Medium
Buffer Overflow boundcpy WrongSizeParam	11	Medium
Integer Overflow	4	Medium

Heuristic 2nd Order Buffer Overflow read	2	Low
Improper Resource Access Authorization	2	Low
NULL Pointer Dereference	2	Low
Unchecked Return Value	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
proxydroid/buffer.c	64
proxydroid/utils.c	4
proxydroid/md5.c	3
proxydroid/buffer_iocp.c	2

Scan Results Details

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=1050056&projectid=50046&pathid=13
Status	New

The size of the buffer used by *red_inet_ntop in buffer, at line 217 of proxydroid/utils.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *red_inet_ntop passes to buffer, at line 217 of proxydroid/utils.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/utils.c	proxydroid/utils.c
Line	217	241
Object	buffer	buffer

Code Snippet
File Name proxydroid/utils.c
Method char *red_inet_ntop(const struct sockaddr_in* sa, char* buffer, size_t buffer_size)

```
....
217.  char *red_inet_ntop(const struct sockaddr_in* sa, char* buffer,
size_t buffer_size)
....
241.              strcpy(buffer, placeholder);
```

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=1050056&projectid=50046&pathid=45
Status	New

The variable declared in first at proxydroid/buffer.c in line 739 is not initialized when it is used by buffer at proxydroid/buffer.c in line 2003.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	742	2028
Object	first	buffer

Code Snippet
File Name proxydroid/buffer.c
Method ZERO_CHAIN(struct evbuffer *dst)

```
....
742.        dst->first = NULL;
```

File Name proxydroid/buffer.c
Method _evbuffer_read_setup_vecs(struct evbuffer *buf, ev_ssize_t howmuch,

```
....
2028.             vecs[i].iov_base = CHAIN_SPACE_PTR(chain);
```

Use of Zero Initialized Pointer\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=46
Status	New

The variable declared in first at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 814.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	742	823
Object	first	last

Code Snippet
File Name proxydroid/buffer.c
Method ZERO_CHAIN(struct evbuffer *dst)

```
....
742.        dst->first = NULL;
```

File Name proxydroid/buffer.c

Method COPY_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
823.        dst->last = src->last;
```

Use of Zero Initialized Pointer\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=47
Status	New

The variable declared in last at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 814.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	743	823
Object	last	last

Code Snippet

File Name proxydroid/buffer.c

Method ZERO_CHAIN(struct evbuffer *dst)

```
....
743.        dst->last = NULL;
```

File Name proxydroid/buffer.c

Method COPY_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
823.        dst->last = src->last;
```

Use of Zero Initialized Pointer\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=48
Status	New

The variable declared in Pointer at proxydroid/buffer.c in line 753 is not initialized when it is used by last at proxydroid/buffer.c in line 814.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	761	823
Object	Pointer	last

Code Snippet
File Name proxydroid/buffer.c
Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
761.              *first = *last = NULL;
```

File Name proxydroid/buffer.c

Method COPY_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
823.        dst->last = src->last;
```

Use of Zero Initialized Pointer\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=49
Status	New

The variable declared in first at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 828.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	742	837
Object	first	last

Code Snippet
File Name proxydroid/buffer.c
Method ZERO_CHAIN(struct evbuffer *dst)

```
....
742.        dst->first = NULL;
```

File Name proxydroid/buffer.c

Method	APPEND_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
837.        dst->last = src->last;
```

Use of Zero Initialized Pointer\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=50
Status	New

The variable declared in last at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 828.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	743	837
Object	last	last

Code Snippet

File Name	proxydroid/buffer.c
Method	ZERO_CHAIN(struct evbuffer *dst)

```
....
743.        dst->last = NULL;
```

File Name	proxydroid/buffer.c
Method	APPEND_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
837.        dst->last = src->last;
```

Use of Zero Initialized Pointer\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=51
Status	New

The variable declared in Pointer at proxydroid/buffer.c in line 753 is not initialized when it is used by last at proxydroid/buffer.c in line 828.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c

Line	761	837
Object	Pointer	last

Code Snippet
File Name proxydroid/buffer.c
Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
761.              *first = *last = NULL;
```

File Name proxydroid/buffer.c

Method APPEND_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
837.        dst->last = src->last;
```

Use of Zero Initialized Pointer\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=52
Status	New

The variable declared in Pointer at proxydroid/buffer.c in line 753 is not initialized when it is used by last at proxydroid/buffer.c in line 797.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	761	808
Object	Pointer	last

Code Snippet
File Name proxydroid/buffer.c
Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
761.              *first = *last = NULL;
```

File Name proxydroid/buffer.c

Method RESTORE_PINNED(struct evbuffer *src, struct evbuffer_chain *pinned,

```
....
808.        src->last = last;
```

Use of Zero Initialized Pointer\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=53
Status	New

The variable declared in last at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 797.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	743	808
Object	last	last

Code Snippet

File Name proxydroid/buffer.c

Method ZERO_CHAIN(struct evbuffer *dst)

```
....
743.        dst->last = NULL;
```

File Name proxydroid/buffer.c

Method RESTORE_PINNED(struct evbuffer *src, struct evbuffer_chain *pinned,

```
....
808.        src->last = last;
```

Use of Zero Initialized Pointer\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=54
Status	New

The variable declared in first at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 797.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	742	808
Object	first	last

Code Snippet

File Name proxydroid/buffer.c

Method ZERO_CHAIN(struct evbuffer *dst)

```
....
742.        dst->first = NULL;
```

File Name proxydroid/buffer.c

Method RESTORE_PINNED(struct evbuffer *src, struct evbuffer_chain *pinned,

```
....
808.        src->last = last;
```

Use of Zero Initialized Pointer\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=55
Status	New

The variable declared in last at proxydroid/buffer.c in line 739 is not initialized when it is used by first at proxydroid/buffer.c in line 814.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	743	818
Object	last	first

Code Snippet

File Name proxydroid/buffer.c

Method ZERO_CHAIN(struct evbuffer *dst)

```
....
743.        dst->last = NULL;
```

File Name proxydroid/buffer.c

Method COPY_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
818.        dst->first = src->first;
```

Use of Zero Initialized Pointer\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=56
Status	New

The variable declared in Pointer at proxydroid/buffer.c in line 753 is not initialized when it is used by first at proxydroid/buffer.c in line 814.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	761	818
Object	Pointer	first

Code Snippet
File Name proxydroid/buffer.c
Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
761.              *first = *last = NULL;
```

File Name proxydroid/buffer.c

Method COPY_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
818.        dst->first = src->first;
```

Use of Zero Initialized Pointer\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=57
Status	New

The variable declared in first at proxydroid/buffer.c in line 739 is not initialized when it is used by first at proxydroid/buffer.c in line 814.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	742	818
Object	first	first

Code Snippet
File Name proxydroid/buffer.c
Method ZERO_CHAIN(struct evbuffer *dst)

```
....
742.        dst->first = NULL;
```

File Name proxydroid/buffer.c

Method	COPY_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
818.        dst->first = src->first;
```

Use of Zero Initialized Pointer\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=58
Status	New

The variable declared in last at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 828.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	743	832
Object	last	last

Code Snippet

File Name proxydroid/buffer.c

Method ZERO_CHAIN(struct evbuffer *dst)

```
....
743.        dst->last = NULL;
```

File Name proxydroid/buffer.c

Method APPEND_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
832.        dst->last->next = src->first;
```

Use of Zero Initialized Pointer\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=59
Status	New

The variable declared in Pointer at proxydroid/buffer.c in line 753 is not initialized when it is used by last at proxydroid/buffer.c in line 828.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c

Line	761	832
Object	Pointer	last

Code Snippet
File Name proxydroid/buffer.c
Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
761.              *first = *last = NULL;
```

File Name proxydroid/buffer.c

Method APPEND_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
832.        dst->last->next = src->first;
```

Use of Zero Initialized Pointer\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=60
Status	New

The variable declared in first at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 828.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	742	832
Object	first	last

Code Snippet
File Name proxydroid/buffer.c
Method ZERO_CHAIN(struct evbuffer *dst)

```
....
742.        dst->first = NULL;
```

File Name proxydroid/buffer.c

Method APPEND_CHAIN(struct evbuffer *dst, struct evbuffer *src)

```
....
832.        dst->last->next = src->first;
```

Use of Zero Initialized Pointer\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=61
Status	New

The variable declared in last at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 753.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	743	789
Object	last	last

Code Snippet

File Name proxydroid/buffer.c

Method ZERO_CHAIN(struct evbuffer *dst)

```
....
743.        dst->last = NULL;
```

File Name proxydroid/buffer.c

Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
789.             src->last = *src->last_with_datap;
```

Use of Zero Initialized Pointer\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=62
Status	New

The variable declared in Pointer at proxydroid/buffer.c in line 753 is not initialized when it is used by last at proxydroid/buffer.c in line 753.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	761	789
Object	Pointer	last

Code Snippet

File Name proxydroid/buffer.c

Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
761.              *first = *last = NULL;
....
789.              src->last = *src->last_with_datap;
```

Use of Zero Initialized Pointer\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=63
Status	New

The variable declared in first at proxydroid/buffer.c in line 739 is not initialized when it is used by last at proxydroid/buffer.c in line 753.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	742	789
Object	first	last

Code Snippet

File Name proxydroid/buffer.c

Method ZERO_CHAIN(struct evbuffer *dst)

```
....
742.        dst->first = NULL;
```

File Name proxydroid/buffer.c

Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
789.              src->last = *src->last_with_datap;
```

Use of Zero Initialized Pointer\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=64
Status	New

The variable declared in Pointer at proxydroid/buffer.c in line 753 is not initialized when it is used by first at proxydroid/buffer.c in line 797.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c

Line	761	807
Object	Pointer	first

Code Snippet
File Name proxydroid/buffer.c
Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
761.              *first = *last = NULL;
```

File Name proxydroid/buffer.c

Method RESTORE_PINNED(struct evbuffer *src, struct evbuffer_chain *pinned,

```
....
807.          src->first = pinned;
```

Use of Zero Initialized Pointer\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=65
Status	New

The variable declared in last at proxydroid/buffer.c in line 739 is not initialized when it is used by first at proxydroid/buffer.c in line 797.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	743	807
Object	last	first

Code Snippet
File Name proxydroid/buffer.c
Method ZERO_CHAIN(struct evbuffer *dst)

```
....
743.          dst->last = NULL;
```

File Name proxydroid/buffer.c

Method RESTORE_PINNED(struct evbuffer *src, struct evbuffer_chain *pinned,

```
....
807.          src->first = pinned;
```

Use of Zero Initialized Pointer\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=66
Status	New

The variable declared in first at proxydroid/buffer.c in line 739 is not initialized when it is used by first at proxydroid/buffer.c in line 797.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	742	807
Object	first	first

Code Snippet

File Name proxydroid/buffer.c

Method ZERO_CHAIN(struct evbuffer *dst)

```
....
742.        dst->first = NULL;
```

File Name proxydroid/buffer.c

Method RESTORE_PINNED(struct evbuffer *src, struct evbuffer_chain *pinned,

```
....
807.        src->first = pinned;
```

Use of Zero Initialized Pointer\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=67
Status	New

The variable declared in last at proxydroid/buffer.c in line 739 is not initialized when it is used by buffer at proxydroid/buffer.c in line 753.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	743	781
Object	last	buffer

Code Snippet

File Name proxydroid/buffer.c

Method ZERO_CHAIN(struct evbuffer *dst)

```
....
743.        dst->last = NULL;
```

File Name proxydroid/buffer.c

Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
781.              memcpy(tmp->buffer, chain->buffer + chain->misalign,
```

Use of Zero Initialized Pointer\Path 24:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=68
Status	New

The variable declared in Pointer at proxydroid/buffer.c in line 753 is not initialized when it is used by buffer at proxydroid/buffer.c in line 753.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	761	781
Object	Pointer	buffer

Code Snippet

File Name proxydroid/buffer.c

Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
761.              *first = *last = NULL;
....
781.              memcpy(tmp->buffer, chain->buffer + chain->misalign,
```

Use of Zero Initialized Pointer\Path 25:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=69
Status	New

The variable declared in first at proxydroid/buffer.c in line 739 is not initialized when it is used by buffer at proxydroid/buffer.c in line 753.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c

Line	742	781
Object	first	buffer

Code Snippet
File Name proxydroid/buffer.c
Method ZERO_CHAIN(struct evbuffer *dst)

```
....
742.        dst->first = NULL;
```

File Name proxydroid/buffer.c

Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
781.              memcpy(tmp->buffer, chain->buffer + chain->misalign,
```

Use of Zero Initialized Pointer\Path 26:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=70
Status	New

The variable declared in chain at proxydroid/buffer.c in line 2498 is not initialized when it is used by buffer at proxydroid/buffer.c in line 1036.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2556	1062
Object	chain	buffer

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_search_range(struct evbuffer *buffer, const char *what, size_t len, const struct evbuffer_ptr *start, const struct evbuffer_ptr *end)

```
....
2556.       pos._internal.chain = NULL;
```

File Name proxydroid/buffer.c

Method evbuffer_copyout(struct evbuffer *buf, void *data_out, size_t datlen)

```
....
1062.             memcpy(data, chain->buffer + chain->misalign, chain-
>off);
```

Use of Zero Initialized Pointer\Path 27:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=71
Status	New

The variable declared in chain at proxydroid/buffer.c in line 2498 is not initialized when it is used by buffer at proxydroid/buffer.c in line 1036.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2556	1072
Object	chain	buffer

Code Snippet

File Name proxydroid/buffer.c

Method evbuffer_search_range(struct evbuffer *buffer, const char *what, size_t len, const struct evbuffer_ptr *start, const struct evbuffer_ptr *end)

```
....
2556.       pos._internal.chain = NULL;
```

File Name proxydroid/buffer.c

Method evbuffer_copyout(struct evbuffer *buf, void *data_out, size_t datlen)

```
....
1072.             memcpy(data, chain->buffer + chain->misalign, datlen);
```

Use of Zero Initialized Pointer\Path 28:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=72
Status	New

The variable declared in chain at proxydroid/buffer.c in line 2498 is not initialized when it is used by next at proxydroid/buffer.c in line 1175.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c

Line	2556	1268
Object	chain	next

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_search_range(struct evbuffer *buffer, const char *what, size_t len, const struct evbuffer_ptr *start, const struct evbuffer_ptr *end)

```
....
2556.       pos._internal.chain = NULL;
```

File Name proxydroid/buffer.c
Method evbuffer_pullup(struct evbuffer *buf, ev_ssize_t size)

```
....
1268.       tmp->next = chain;
```

Use of Zero Initialized Pointer\Path 29:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=73
Status	New

The variable declared in chain at proxydroid/buffer.c in line 2498 is not initialized when it is used by last at proxydroid/buffer.c in line 1175.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2556	1265
Object	chain	last

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_search_range(struct evbuffer *buffer, const char *what, size_t len, const struct evbuffer_ptr *start, const struct evbuffer_ptr *end)

```
....
2556.       pos._internal.chain = NULL;
```

File Name proxydroid/buffer.c
Method evbuffer_pullup(struct evbuffer *buf, ev_ssize_t size)

```
....
1265.             buf->last = tmp;
```

Use of Zero Initialized Pointer\Path 30:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=74
Status	New

The variable declared in chain at proxydroid/buffer.c in line 2498 is not initialized when it is used by first at proxydroid/buffer.c in line 1175.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2556	1239
Object	chain	first

Code Snippet

File Name proxydroid/buffer.c

Method evbuffer_search_range(struct evbuffer *buffer, const char *what, size_t len, const struct evbuffer_ptr *start, const struct evbuffer_ptr *end)

```
....
2556.       pos._internal.chain = NULL;
```

File Name proxydroid/buffer.c

Method evbuffer_pullup(struct evbuffer *buf, ev_ssize_t size)

```
....
1239.             buf->first = tmp;
```

Use of Zero Initialized Pointer\Path 31:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=75
Status	New

The variable declared in buffer at proxydroid/buffer.c in line 2736 is not initialized when it is used by first at proxydroid/buffer.c in line 284.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c

Line	2763	292
Object	buffer	first

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_add_file(struct evbuffer *outbuf, int fd,

```
....
2763.             chain->buffer = NULL;    /* no reading possible */
```

File Name proxydroid/buffer.c

Method evbuffer_chain_insert(struct evbuffer *buf,

```
....
292.              buf->first = buf->last = chain;
```

Use of Zero Initialized Pointer\Path 32:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=76
Status	New

The variable declared in buffer at proxydroid/buffer.c in line 2736 is not initialized when it is used by last at proxydroid/buffer.c in line 284.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2763	292
Object	buffer	last

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_add_file(struct evbuffer *outbuf, int fd,

```
....
2763.             chain->buffer = NULL;    /* no reading possible */
```

File Name proxydroid/buffer.c

Method evbuffer_chain_insert(struct evbuffer *buf,

```
....
292.              buf->first = buf->last = chain;
```

Use of Zero Initialized Pointer\Path 33:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=77
Status	New

The variable declared in buffer at proxydroid/buffer.c in line 2736 is not initialized when it is used by last at proxydroid/buffer.c in line 284.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2763	309
Object	buffer	last

Code Snippet

File Name proxydroid/buffer.c

Method evbuffer_add_file(struct evbuffer *outbuf, int fd,

```
....
2763.             chain->buffer = NULL;    /* no reading possible */
```

File Name proxydroid/buffer.c

Method evbuffer_chain_insert(struct evbuffer *buf,

```
....
309.              buf->last = chain;
```

Use of Zero Initialized Pointer\Path 34:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=78
Status	New

The variable declared in buffer at proxydroid/buffer.c in line 2736 is not initialized when it is used by last at proxydroid/buffer.c in line 284.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2763	300
Object	buffer	last

Code Snippet

File Name proxydroid/buffer.c

Method evbuffer_add_file(struct evbuffer *outbuf, int fd,

```
....
2763.             chain->buffer = NULL;    /* no reading possible */
```

File Name proxydroid/buffer.c

Method evbuffer_chain_insert(struct evbuffer *buf,

```
....
300.                     buf->last->next = chain;
```

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=1050056&projectid=50046&pathid=22
Status	New

The dangerous function, memcpy, was found in use at line 753 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	781	781
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
781.             memcpy(tmp->buffer, chain->buffer + chain->misalign,
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=23
Status	New

The dangerous function, memcpy, was found in use at line 1036 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1062	1062
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_copyout(struct evbuffer *buf, void *data_out, size_t datlen)

```
....
1062.             memcpy(data, chain->buffer + chain->misalign, chain-
>off);
```

Dangerous Functions\Path 3:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=24
Status New

The dangerous function, memcpy, was found in use at line 1036 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1072	1072
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_copyout(struct evbuffer *buf, void *data_out, size_t datlen)

```
....
1072.             memcpy(data, chain->buffer + chain->misalign, datlen);
```

Dangerous Functions\Path 4:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=25
Status New

The dangerous function, memcpy, was found in use at line 1175 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1249	1249
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_pullup(struct evbuffer *buf, ev_ssize_t size)

```
....
1249.             memcpy(buffer, chain->buffer + chain->misalign, chain-
>off);
```

Dangerous Functions\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=26
Status	New

The dangerous function, memcpy, was found in use at line 1175 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1261	1261
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_pullup(struct evbuffer *buf, ev_ssize_t size)

```
....
1261.             memcpy(buffer, chain->buffer + chain->misalign, size);
```

Dangerous Functions\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=27
Status	New

The dangerous function, memcpy, was found in use at line 1417 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1428	1428
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_search_eol(struct evbuffer *buffer,

```
....
1428.             memcpy(&it, start, sizeof(it));
```

Dangerous Functions\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=28
Status	New

The dangerous function, memcpy, was found in use at line 1417 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1441	1441
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_search_eol(struct evbuffer *buffer,

```
....
1441.             memcpy(&it2, &it, sizeof(it));
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=29
Status	New

The dangerous function, memcpy, was found in use at line 1535 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c

Line	1564	1564
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_add(struct evbuffer *buf, const void *data_in, size_t datlen)

```
....
1564.                    memcpy(chain->buffer + chain->misalign + chain-
>off,
```

Dangerous Functions\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=30
Status	New

The dangerous function, memcpy, was found in use at line 1535 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1575	1575
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_add(struct evbuffer *buf, const void *data_in, size_t datlen)

```
....
1575.                    memcpy(chain->buffer + chain->off, data,
datlen);
```

Dangerous Functions\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=31
Status	New

The dangerous function, memcpy, was found in use at line 1535 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1597	1597

Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_add(struct evbuffer *buf, const void *data_in, size_t datlen)

```
....
1597.             memcpy(chain->buffer + chain->misalign + chain->off,
```

Dangerous Functions\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=32
Status	New

The dangerous function, memcpy, was found in use at line 1535 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1607	1607
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_add(struct evbuffer *buf, const void *data_in, size_t datlen)

```
....
1607.       memcpy(tmp->buffer, data, datlen);
```

Dangerous Functions\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=33
Status	New

The dangerous function, memcpy, was found in use at line 1621 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1650	1650
Object	memcpy	memcpy

Code Snippet	
File Name	proxydroid/buffer.c
Method	evbuffer_prepend(struct evbuffer *buf, const void *data, size_t datlen)

```
....
1650.                    memcpy(chain->buffer + chain->misalign - datlen,
```

Dangerous Functions\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=34
Status	New

The dangerous function, memcpy, was found in use at line 1621 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1659	1659
Object	memcpy	memcpy

Code Snippet	
File Name	proxydroid/buffer.c
Method	evbuffer_prepend(struct evbuffer *buf, const void *data, size_t datlen)

```
....
1659.                    memcpy(chain->buffer,
```

Dangerous Functions\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=35
Status	New

The dangerous function, memcpy, was found in use at line 1621 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1682	1682
Object	memcpy	memcpy

Code Snippet	
File Name	proxydroid/buffer.c
Method	evbuffer_prepend(struct evbuffer *buf, const void *data, size_t datlen)

```
....
1682.       memcpy(tmp->buffer + tmp->misalign, data, datlen);
```

Dangerous Functions\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=36
Status	New

The dangerous function, memcpy, was found in use at line 1721 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1804	1804
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_expand_singlechain(struct evbuffer *buf, size_t datlen)

```
....
1804.             memcpy(tmp->buffer, chain->buffer + chain->misalign,
```

Dangerous Functions\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=37
Status	New

The dangerous function, memcpy, was found in use at line 2498 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2508	2508
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_search_range(struct evbuffer *buffer, const char *what, size_t len, const struct evbuffer_ptr *start, const struct evbuffer_ptr *end)

```
....
2508.            memcpy(&pos, start, sizeof(pos));
```

Dangerous Functions\Path 17:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=38
Status	New

The dangerous function, memcpy, was found in use at line 2615 in proxydroid/buffer.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2647	2647
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_add_vprintf(struct evbuffer *buf, const char *fmt, va_list ap)

```
....
2647.            va_copy(aq, ap);
```

Dangerous Functions\Path 18:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=39
Status	New

The dangerous function, memcpy, was found in use at line 131 in proxydroid/md5.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/md5.c	proxydroid/md5.c
Line	168	168
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/md5.c
Method static void md5_process(md5_state_t *pms, const md5_byte_t *data /*[64]*/)

```
....
168.                            memcpy(xbuf, data, 64);
```

Dangerous Functions\Path 19:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=40
Status	New

The dangerous function, memcpy, was found in use at line 320 in proxydroid/md5.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/md5.c	proxydroid/md5.c
Line	340	340
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/md5.c
Method void md5_append(md5_state_t *pms, const md5_byte_t *data, int nbytes)

```
....
340.                memcpy(pms->buf + offset, p, copy);
```

Dangerous Functions\Path 20:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=41
Status	New

The dangerous function, memcpy, was found in use at line 320 in proxydroid/md5.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/md5.c	proxydroid/md5.c
Line	354	354
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/md5.c
Method void md5_append(md5_state_t *pms, const md5_byte_t *data, int nbytes)

```
....
354.                memcpy(pms->buf, p, left);
```

Dangerous Functions\Path 21:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=42
Status	New

The dangerous function, memcpy, was found in use at line 30 in proxydroid/utils.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/utils.c	proxydroid/utils.c
Line	65	65
Object	memcpy	memcpy

Code Snippet
File Name proxydroid/utils.c
Method int red_recv_udp_pkt(int fd, char *buf, size_t buflen, struct sockaddr_in *inaddr, struct sockaddr_in *toaddr)

```
....
65.                     memcpy(toaddr, cmsgaddr, sizeof(*toaddr));
```

Dangerous Functions\Path 22:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=43
Status	New

The dangerous function, strcpy, was found in use at line 217 in proxydroid/utils.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/utils.c	proxydroid/utils.c
Line	241	241
Object	strcpy	strcpy

Code Snippet
File Name proxydroid/utils.c
Method char *red_inet_ntop(const struct sockaddr_in* sa, char* buffer, size_t buffer_size)

```
....
241.              strcpy(buffer, placeholder);
```

Dangerous Functions\Path 23:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=44
Status	New

The dangerous function, strlen, was found in use at line 217 in proxydroid/utils.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	proxydroid/utils.c	proxydroid/utils.c
Line	237	237
Object	strlen	strlen

Code Snippet
File Name proxydroid/utils.c
Method char *red_inet_ntop(const struct sockaddr_in* sa, char* buffer, size_t buffer_size)

```
....
237.              len = strlen(retval);
```

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=1050056&projectid=50046&pathid=2
Status	New

The size of the buffer used by evbuffer_search_eol in it, at line 1417 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_search_eol passes to it, at line 1417 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c

Line	1441	1441
Object	it	it

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_search_eol(struct evbuffer *buffer,

```
....
1441.             memcpy(&it2, &it, sizeof(it));
```

Buffer Overflow boundcpy WrongSizeParam\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=3
Status	New

The size of the buffer used by evbuffer_add_vprintf in va_list, at line 2615 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_add_vprintf passes to va_list, at line 2615 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2647	2647
Object	va_list	va_list

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_add_vprintf(struct evbuffer *buf, const char *fmt, va_list ap)

```
....
2647.             va_copy(aq, ap);
```

Buffer Overflow boundcpy WrongSizeParam\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=4
Status	New

The size of the buffer used by evbuffer_launch_write in ->, at line 170 of proxydroid/buffer_iocp.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_launch_write passes to ->, at line 170 of proxydroid/buffer_iocp.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer_iocp.c	proxydroid/buffer_iocp.c

Line	199	199
Object	->	->

Code Snippet
File Name proxydroid/buffer_iocp.c
Method evbuffer_launch_write(struct evbuffer *buf, ev_ssize_t at_most,

```
....
199.        memset(buf_o->buffers, 0, sizeof(buf_o->buffers));
```

Buffer Overflow boundcpy WrongSizeParam\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=5
Status	New

The size of the buffer used by evbuffer_launch_read in ->, at line 244 of proxydroid/buffer_iocp.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_launch_read passes to ->, at line 244 of proxydroid/buffer_iocp.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer_iocp.c	proxydroid/buffer_iocp.c
Line	265	265
Object	->	->

Code Snippet
File Name proxydroid/buffer_iocp.c
Method evbuffer_launch_read(struct evbuffer *buf, size_t at_most,

```
....
265.        memset(buf_o->buffers, 0, sizeof(buf_o->buffers));
```

Buffer Overflow boundcpy WrongSizeParam\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=6
Status	New

The size of the buffer used by PRESERVE_PINNED in chain, at line 753 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that PRESERVE_PINNED passes to chain, at line 753 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c

Line	782	782
Object	chain	chain

Code Snippet
File Name proxydroid/buffer.c
Method PRESERVE_PINNED(struct evbuffer *src, struct evbuffer_chain **first,

```
....
782.                    chain->off);
```

Buffer Overflow boundcpy WrongSizeParam\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=7
Status	New

The size of the buffer used by evbuffer_copyout in chain, at line 1036 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_copyout passes to chain, at line 1036 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1062	1062
Object	chain	chain

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_copyout(struct evbuffer *buf, void *data_out, size_t datlen)

```
....
1062.            memcpy(data, chain->buffer + chain->misalign, chain-
>off);
```

Buffer Overflow boundcpy WrongSizeParam\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=8
Status	New

The size of the buffer used by evbuffer_pullup in chain, at line 1175 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_pullup passes to chain, at line 1175 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1249	1249

Object	chain	chain

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_pullup(struct evbuffer *buf, ev_ssize_t size)

```
....
1249.            memcpy(buffer, chain->buffer + chain->misalign, chain-
>off);
```

Buffer Overflow boundcpy WrongSizeParam\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=9
Status	New

The size of the buffer used by evbuffer_pullup in size, at line 1175 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_pullup passes to size, at line 1175 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1261	1261
Object	size	size

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_pullup(struct evbuffer *buf, ev_ssize_t size)

```
....
1261.            memcpy(buffer, chain->buffer + chain->misalign, size);
```

Buffer Overflow boundcpy WrongSizeParam\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=10
Status	New

The size of the buffer used by evbuffer_expand_singlechain in chain, at line 1721 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_expand_singlechain passes to chain, at line 1721 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1805	1805

Object	chain	chain

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_expand_singlechain(struct evbuffer *buf, size_t datlen)

```
....
1805.                  chain->off);
```

Buffer Overflow boundcpy WrongSizeParam\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=11
Status	New

The size of the buffer used by evbuffer_chain_align in chain, at line 1696 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_chain_align passes to chain, at line 1696 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1700	1700
Object	chain	chain

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_chain_align(struct evbuffer_chain *chain)

```
....
1700.       memmove(chain->buffer, chain->buffer + chain->misalign,
chain->off);
```

Buffer Overflow boundcpy WrongSizeParam\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=12
Status	New

The size of the buffer used by evbuffer_ptr_memcmp in n_comparable, at line 2458 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_ptr_memcmp passes to n_comparable, at line 2458 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2479	2479

Object	n_comparable	n_comparable

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_ptr_memcmp(const struct evbuffer *buf, const struct evbuffer_ptr *pos,

```
....
2479.                   n_comparable);
```

Integer Overflow

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

Categories

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

Description

Integer Overflow\Path 1:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=18
Status New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 1085 of proxydroid/buffer.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1113	1113
Object	AssignExpr	AssignExpr

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_remove_buffer(struct evbuffer *src, struct evbuffer *dst,

```
....
1113.             result = (int)datlen; /*XXXX should return
ev_ssize_t*/
```

Integer Overflow\Path 2:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=19
Status New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 1085 of proxydroid/buffer.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	1167	1167
Object	AssignExpr	AssignExpr

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_remove_buffer(struct evbuffer *src, struct evbuffer *dst,

```
....
1167.       result = (int)nread;/*XXXX should change return type */
```

Integer Overflow\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=20
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 2059 of proxydroid/buffer.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2119	2119
Object	AssignExpr	AssignExpr

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_read(struct evbuffer *buf, evutil_socket_t fd, int howmuch)

```
....
2119.                           n = bytesRead;
```

Integer Overflow\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=21
Status	New

A variable of a larger data type, AssignExpr, is being assigned to a smaller data type, in 2235 of proxydroid/buffer.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2275	2275
Object	AssignExpr	AssignExpr

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_write_iovec(struct evbuffer *buffer, evutil_socket_t fd,

```
....
2275.                    n = bytesSent;
```

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=1050056&projectid=50046&pathid=14
Status	New

The variable declared in null at proxydroid/buffer.c in line 2492 is not initialized when it is used by _internal at proxydroid/buffer.c in line 2498.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2494	2517
Object	null	_internal

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_search(struct evbuffer *buffer, const char *what, size_t len, const struct evbuffer_ptr *start)

```
....
2494.        return evbuffer_search_range(buffer, what, len, start,
NULL);
```

File Name proxydroid/buffer.c

Method	evbuffer_search_range(struct evbuffer *buffer, const char *what, size_t len, const struct evbuffer_ptr *start, const struct evbuffer_ptr *end)

```
....
2517.             last_chain = end->_internal.chain;
```

NULL Pointer Dereference\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=15
Status	New

The variable declared in 0 at proxydroid/buffer.c in line 614 is not initialized when it is used by vec at proxydroid/buffer.c in line 614.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	630	630
Object	0	vec

Code Snippet	
File Name	proxydroid/buffer.c
Method	evbuffer_reserve_space(struct evbuffer *buf, ev_ssize_t size,

```
....
630.             vec[0].iov_len = (size_t) CHAIN_SPACE_LEN(chain);
```

Heuristic 2nd Order Buffer Overflow read

Query Path:
CPP\Cx\CPP Heuristic\Heuristic 2nd Order Buffer Overflow read Version:0

Categories

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

Description

Heuristic 2nd Order Buffer Overflow read\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=16
Status	New

The size of the buffer used by evbuffer_readfile in iov_len, at line 2185 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_readfile passes to iov_base, at line 2185 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2209	2216
Object	iov_base	iov_len

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_readfile(struct evbuffer *buf, evutil_socket_t fd, ev_ssize_t howmuch)

```
....
2209.       n = read((int)fd, v[0].iov_base, (unsigned
int)v[0].iov_len);
....
2216.             n = read((int)fd, v[1].iov_base, (unsigned
int)v[1].iov_len);
```

Heuristic 2nd Order Buffer Overflow read\Path 2:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=17
Status New

The size of the buffer used by evbuffer_readfile in iov_len, at line 2185 of proxydroid/buffer.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that evbuffer_readfile passes to iov_base, at line 2185 of proxydroid/buffer.c, to overwrite the target buffer.

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2209	2216
Object	iov_base	iov_len

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_readfile(struct evbuffer *buf, evutil_socket_t fd, ev_ssize_t howmuch)

```
....
2209.       n = read((int)fd, v[0].iov_base, (unsigned
int)v[0].iov_len);
....
2216.             n = read((int)fd, v[1].iov_base, (unsigned
int)v[1].iov_len);
```

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)

Description

Improper Resource Access Authorization\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=79
Status	New

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2209	2209
Object	iov_base	iov_base

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_readfile(struct evbuffer *buf, evutil_socket_t fd, ev_ssize_t howmuch)

```
....
2209.        n = read((int)fd, v[0].iov_base, (unsigned
int)v[0].iov_len);
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=80
Status	New

	Source	Destination
File	proxydroid/buffer.c	proxydroid/buffer.c
Line	2216	2216
Object	iov_base	iov_base

Code Snippet
File Name proxydroid/buffer.c
Method evbuffer_readfile(struct evbuffer *buf, evutil_socket_t fd, ev_ssize_t howmuch)

```
....
2216.              n = read((int)fd, v[1].iov_base, (unsigned
int)v[1].iov_len);
```

Unchecked Return Value

Query Path:
CPP\Cx\CPP Low Visibility\Unchecked Return Value Version:1

Categories

Description

Unchecked Return Value\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1050056&projectid=50046&pathid=1
Status	New

The *red_inet_ntop method calls the snprintf function, at line 217 of proxydroid/utils.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	proxydroid/utils.c	proxydroid/utils.c
Line	238	238
Object	snprintf	snprintf

Code Snippet

File Name proxydroid/utils.c

Method char *red_inet_ntop(const struct sockaddr_in* sa, char* buffer, size_t buffer_size)

```
....
238.              snprintf(buffer + len, buffer_size - len, ":%d",
ntohs(port));
```

Buffer Overflow StrcpyStrcat

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Buffer Overflow boundcpy WrongSizeParam

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

CPP

Overflowing Buffers

```
const int BUFFER_SIZE = 10;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
{
    strcpy(buffer, inputString);
}
```

Checked Buffers

```
const int BUFFER_SIZE = 10;
const int MAX_INPUT_SIZE = 256;
```

```
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
{
    if (strnlen(inputString, MAX_INPUT_SIZE) < sizeof(buffer))
    {
        strncpy(buffer, inputString, sizeof(buffer));
    }
}
```

Integer Overflow

Risk

What might happen

Assigning large data types into smaller data types, without proper checks and explicit casting, will lead to undefined behavior and unintentional effects, such as data corruption (e.g. value wraparound, wherein maximum values become minimum values); system crashes; infinite loops; logic errors, such as bypassing of security mechanisms; or even buffer overflows leading to arbitrary code execution.

Cause

How does it happen

This flaw can occur when implicitly casting numerical data types of a larger size, into a variable with a data type of a smaller size. This forces the program to discard some bits of information from the number. Depending on how the numerical data types are stored in memory, this is often the bits with the highest value, causing substantial corruption of the stored number. Alternatively, the sign bit of a signed integer could be lost, completely reversing the intention of the number.

General Recommendations

How to avoid it

- Avoid casting larger data types to smaller types.
- Prefer promoting the target variable to a large enough data type.
- If downcasting is necessary, always check that values are valid and in range of the target type, before casting

---

Source Code Examples

CPP

Unsafe Downsize Casting

```
int unsafe_addition(short op1, int op2) {

    // op2 gets forced from int into a short
    short total = op1 + op2;

    return total;
}
```

Safer Use of Proper Data Types

```
int safe_addition(short op1, int op2) {

    // total variable is of type int, the largest type that is needed
    int total = 0;

    // check if total will overflow available integer size
    if (INT_MAX - abs(op2) > op1)
```

```
    {
        total = op1 + op2;
    }
    else
    {
        // instead of overflow, saturate (but this is not always a good thing)
        total = INT_MAX
    }

    return total;
}
```

Dangerous Functions

Risk

What might happen

Use of dangerous functions may expose varying risks associated with each particular function, with potential impact of improper usage of these functions varying significantly. The presence of such functions indicates a flaw in code maintenance policies and adherence to secure coding practices, in a way that has allowed introducing known dangerous code into the application.

Cause

How does it happen

A dangerous function has been identified within the code. Functions are often deemed dangerous to use for numerous reasons, as there are different sets of vulnerabilities associated with usage of such functions. For example, some string copy and concatenation functions are vulnerable to Buffer Overflow, Memory Disclosure, Denial of Service and more. Use of these functions is not recommended.

General Recommendations

How to avoid it

- Deploy a secure and recommended alternative to any functions that were identified as dangerous.
    - If no secure alternative is found, conduct further researching and testing to identify whether current usage successfully sanitizes and verifies values, and thus successfully avoids the use-cases for whom the function is indeed dangerous
- Conduct a periodical review of methods that are in use, to ensure that all external libraries and built-in functions are up-to-date and whose use has not been excluded from best secure coding practices.

Source Code Examples

CPP

Buffer Overflow in gets()

```
int main()
{
	char buf[10];

	printf("Please enter your name: ");
	gets(buf); // veryveryverylongname
	if (buf == ACCEPTED_NAME)
	{
		// Do something
	}
	return 0;
}
```

Safe reading from user

```
int main()
{
	char buf[10];

	printf("Please enter your name: ");
	fgets(buf, sizeof(buf), stdin); //setting the amount of bytes to read
	if (buf == ACCEPTED_NAME)
	{
		//Do something
	}
	return 0;
}
```

Unsafe function for string copy

```
int main(int argc, char* argv[])
{
	char buf[10];
	strcpy(buf, argv[1]); // overflow occurs when len(argv[1]) > 10 bytes

	return 0;
}
```

Safe string copy

```
int main(int argc, char* argv[])
{
	char buf[10];
	strncpy(buf, argv[1], sizeof(buf));
	buf[9]= '\0'; //strncpy doesn't NULL terminates

	return 0;
}
```

Unsafe format string

```
int main(int argc, char* argv[])
{
	printf(argv[1]); // If argv[1] contains a format token, such as %s,%x or %d, will cause
an access violation
	return 0;
}
```

Safe format string

```
int main(int argc, char* argv[])
{
      printf("%s", argv[1]); // Second parameter is not a formattable string
      return 0;
}
```

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());
```

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';
```

NULL Pointer Dereference

Risk

What might happen

A null pointer dereference is likely to cause a run-time exception, a crash, or other unexpected behavior.

Cause

How does it happen

Variables which are declared without being assigned will implicitly retain a null value until they are assigned. The null value can also be explicitly set to a variable, to ensure clear out its contents. Since null is not really a value, it may not have object variables and methods, and any attempt to access contents of a null object, instead of verifying it is set beforehand, will result in a null pointer dereference exception.

General Recommendations

How to avoid it

- For any variable that is created, ensure all logic flows between declaration and use assign a non-null value to the variable first.
- Enforce null checks on any received variable or object before it is dereferenced, to ensure it does not contain a null assigned to it elsewhere.
- Consider the need to assign null values in order to overwrite initialized variables. Consider reassigning or releasing these variables instead.

Source Code Examples

Heuristic 2nd Order Buffer Overflow read

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.
Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

Improper Access Control (Authorization)

Weakness ID: 285 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software does not perform or incorrectly performs access control checks across all potential execution paths.

Extended Description

When access control checks are not applied consistently - or not at all - users are able to access data or perform actions that they should not be allowed to perform. This can lead to a wide range of problems, including information leaks, denial of service, and arbitrary code execution.

Alternate Terms

AuthZ:	"AuthZ" is typically used as an abbreviation of "authorization" within the web application security community. It is also distinct from "AuthC," which is an abbreviation of "authentication." The use of "Auth" as an abbreviation is discouraged, since it could be used for either authentication or authorization.

Time of Introduction

- Architecture and Design
- Implementation
- Operation

Applicable Platforms

Languages

Language-independent

Technology Classes

Web-Server: *(Often)*

Database-Server: *(Often)*

Modes of Introduction

A developer may introduce authorization weaknesses because of a lack of understanding about the underlying technologies. For example, a developer may assume that attackers cannot modify certain inputs such as headers or cookies.

Authorization weaknesses may arise when a single-user application is ported to a multi-user environment.

Common Consequences

Scope	Effect
Confidentiality	An attacker could read sensitive data, either by reading the data directly from a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to read the data.
Integrity	An attacker could modify sensitive data, either by writing the data directly to a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to write the data.
Integrity	An attacker could gain privileges by modifying or reading critical data directly, or by accessing insufficiently-protected, privileged functionality.

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

Automated static analysis is useful for detecting commonly-used idioms for authorization. A tool may be able to analyze related configuration files, such as .htaccess in Apache web servers, or detect the usage of commonly-used authorization libraries.

Generally, automated static analysis tools have difficulty detecting custom authorization schemes. In addition, the software's design may include some functionality that is accessible to any user and does not require an authorization check; an automated technique that detects the absence of authorization may report false positives.

Effectiveness: Limited

Automated Dynamic Analysis

Automated dynamic analysis may find many or all possible interfaces that do not require authorization, but manual analysis is required to determine if the lack of authorization violates business logic

Manual Analysis

This weakness can be detected using tools and techniques that require manual (human) analysis, such as penetration testing, threat modeling, and interactive tools that allow the tester to record and modify an active session.

Specifically, manual static analysis is useful for evaluating the correctness of custom authorization mechanisms.

Effectiveness: Moderate

These may be more effective than strictly automated techniques. This is especially the case with weaknesses that are related to design and business rules. However, manual efforts might not achieve desired code coverage within limited time constraints.

Demonstrative Examples

Example 1

The following program could be part of a bulletin board system that allows users to send private messages to each other. This program intends to authenticate the user before deciding whether a private message should be displayed. Assume that LookupMessageObject() ensures that the $id argument is numeric, constructs a filename based on that id, and reads the message details from that file. Also assume that the program stores all private messages for all users in the same directory.

(Bad Code)

Example Language: **Perl**

```
sub DisplayPrivateMessage {
my($id) = @_;
my $Message = LookupMessageObject($id);
print "From: " . encodeHTML($Message->{from}) . "<br>\n";
print "Subject: " . encodeHTML($Message->{subject}) . "\n";
print "<hr>\n";
print "Body: " . encodeHTML($Message->{body}) . "\n";
}

my $q = new CGI;
# For purposes of this example, assume that CWE-309 and
# CWE-523 do not apply.
if (! AuthenticateUser($q->param('username'), $q->param('password'))) {
ExitError("invalid username or password");
}

my $id = $q->param('id');
DisplayPrivateMessage($id);
```

While the program properly exits if authentication fails, it does not ensure that the message is addressed to the user. As a result, an authenticated attacker could provide any arbitrary identifier and read private messages that were intended for other users.

One way to avoid this problem would be to ensure that the "to" field in the message object matches the username of the authenticated user.

Observed Examples

Reference	Description
CVE-2009-3168	Web application does not restrict access to admin scripts, allowing authenticated users to reset administrative passwords.

CVE-2009-2960	Web application does not restrict access to admin scripts, allowing authenticated users to modify passwords of other users.
CVE-2009-3597	Web application stores database file under the web root with insufficient access control (CWE-219), allowing direct request.
CVE-2009-2282	Terminal server does not check authorization for guest access.
CVE-2009-3230	Database server does not use appropriate privileges for certain sensitive operations.
CVE-2009-2213	Gateway uses default "Allow" configuration for its authorization settings.
CVE-2009-0034	Chain: product does not properly interpret a configuration option for a system group, allowing users to gain privileges.
CVE-2008-6123	Chain: SNMP product does not properly parse a configuration option for which hosts are allowed to connect, allowing unauthorized IP addresses to connect.
CVE-2008-5027	System monitoring software allows users to bypass authorization by creating custom forms.
CVE-2008-7109	Chain: reliance on client-side security (CWE-602) allows attackers to bypass authorization using a custom client.
CVE-2008-3424	Chain: product does not properly handle wildcards in an authorization policy list, allowing unintended access.
CVE-2009-3781	Content management system does not check access permissions for private files, allowing others to view those files.
CVE-2008-4577	ACL-based protection mechanism treats negative access rights as if they are positive, allowing bypass of intended restrictions.
CVE-2008-6548	Product does not check the ACL of a page accessed using an "include" directive, allowing attackers to read unauthorized files.
CVE-2007-2925	Default ACL list for a DNS server does not set certain ACLs, allowing unauthorized DNS queries.
CVE-2006-6679	Product relies on the X-Forwarded-For HTTP header for authorization, allowing unintended access by spoofing the header.
CVE-2005-3623	OS kernel does not check for a certain privilege before setting ACLs for files.
CVE-2005-2801	Chain: file-system code performs an incorrect comparison (CWE-697), preventing defauls ACLs from being properly applied.
CVE-2001-1155	Chain: product does not properly check the result of a reverse DNS lookup because of operator precedence (CWE-783), allowing bypass of DNS-based access restrictions.

Potential Mitigations

Phase: Architecture and Design

Divide your application into anonymous, normal, privileged, and administrative areas. Reduce the attack surface by carefully mapping roles with data and functionality. Use role-based access control (RBAC) to enforce the roles at the appropriate boundaries.

Note that this approach may not protect against horizontal authorization, i.e., it will not protect a user from attacking others with the same role.

Phase: Architecture and Design

Ensure that you perform access control checks related to your business logic. These checks may be different than the access control checks that you apply to more generic resources such as files, connections, processes, memory, and database records. For example, a database may restrict access for medical records to a specific database user, but each record might only be intended to be accessible to the patient and the patient's doctor.

Phase: Architecture and Design

Strategy: Libraries or Frameworks

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness

easier to avoid.

For example, consider using authorization frameworks such as the JAAS Authorization Framework and the OWASP ESAPI Access Control feature.

Phase: Architecture and Design

For web applications, make sure that the access control mechanism is enforced correctly at the server side on every page. Users should not be able to access any unauthorized functionality or information by simply requesting direct access to that page.

One way to do this is to ensure that all pages containing sensitive information are not cached, and that all such pages restrict access to requests that are accompanied by an active and authenticated session token associated with a user who has the required permissions to access that page.

Phases: System Configuration; Installation

Use the access control capabilities of your operating system and server environment and define your access control lists accordingly. Use a "default deny" policy when defining these ACLs.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	254	Security Features	**Seven Pernicious Kingdoms (primary)700**
ChildOf	Weakness Class	284	Access Control (Authorization) Issues	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	721	OWASP Top Ten 2007 Category A10 - Failure to Restrict URL Access	**Weaknesses in OWASP Top Ten (2007) (primary)629**
ChildOf	Category	723	OWASP Top Ten 2004 Category A2 - Broken Access Control	**Weaknesses in OWASP Top Ten (2004) (primary)711**
ChildOf	Category	753	2009 Top 25 - Porous Defenses	**Weaknesses in the 2009 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)750**
ChildOf	Category	803	2010 Top 25 - Porous Defenses	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
ParentOf	Weakness Variant	219	Sensitive Data Under Web Root	**Research Concepts (primary)1000**
ParentOf	Weakness Base	551	Incorrect Behavior Order: Authorization Before Parsing and Canonicalization	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Class	638	Failure to Use Complete Mediation	Research Concepts1000
ParentOf	Weakness Base	804	Guessable CAPTCHA	**Development Concepts (primary)699 Research Concepts (primary)1000**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
7 Pernicious Kingdoms			Missing Access Control
OWASP Top Ten 2007	A10	CWE More Specific	Failure to Restrict URL Access
OWASP Top Ten 2004	A2	CWE More Specific	Broken Access Control

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
1	Accessing Functionality Not Properly Constrained by ACLs	
13	Subverting Environment Variable Values	

17	Accessing, Modifying or Executing Executable Files
87	Forceful Browsing
39	Manipulating Opaque Client-based Data Tokens
45	Buffer Overflow via Symbolic Links
51	Poison Web Service Registry
59	Session Credential Falsification through Prediction
60	Reusing Session IDs (aka Session Replay)
77	Manipulating User-Controlled Variables
76	Manipulating Input to File System Calls
104	Cross Zone Scripting

References

NIST. "Role Based Access Control and Role Based Security". <http://csrc.nist.gov/groups/SNS/rbac/>.

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 4, "Authorization" Page 114; Chapter 6, "Determining Appropriate Access Control" Page 171. 2nd Edition. Microsoft. 2002.

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	7 Pernicious Kingdoms		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-15		Veracode	External
	Suggested OWASP Top Ten 2004 mapping		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Relationships, Other Notes, Taxonomy Mappings		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Common Consequences, Description, Likelihood of Exploit, Name, Other Notes, Potential Mitigations, References, Relationships		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Potential Mitigations		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Description, Related Attack Patterns		
2009-07-27	CWE Content Team	MITRE	Internal
	updated Relationships		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Type		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Demonstrative Examples, Detection Factors, Modes of Introduction, Observed Examples, Relationships		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Detection Factors, Potential Mitigations, References, Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Potential Mitigations		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2009-01-12	Missing or Inconsistent Access Control		

BACK TO TOP

Scanned Languages

Language	Hash Number	Change Date
CPP	4541647240435660	6/19/2024
Common	0105849645654507	6/19/2024