

vul_files_44 Scan Report

Project Name	vul_files_44
Scan Start	Wednesday, January 8, 2025 10:03:43 AM
Preset	Checkmarx Default
Scan Time	01h:31m:32s
Lines Of Code Scanned	287101
Files Scanned	20
Report Creation Time	Wednesday, January 8, 2025 10:36:18 AM
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046
Team	CxServer
Checkmarx Version	8.7.0
Scan Type	Full
Source Origin	LocalPath
Density	2/10000 (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

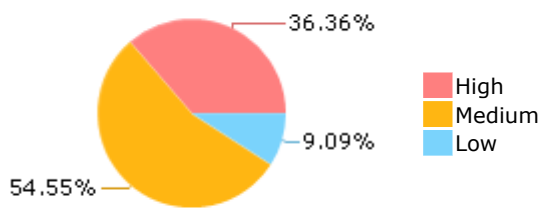
Results Limit

Results limit per query was set to 50

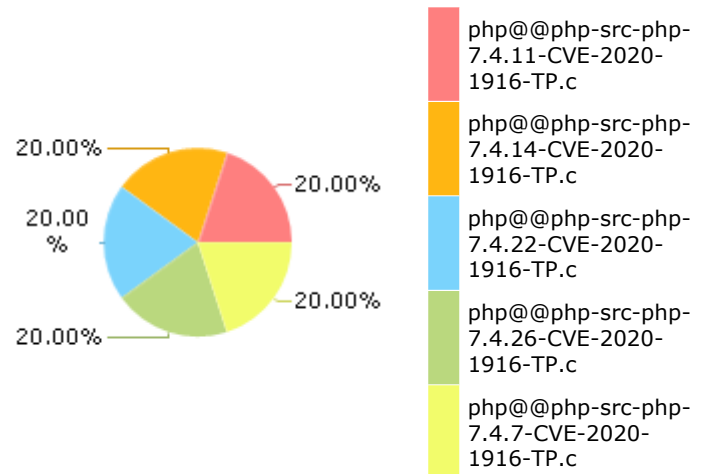
Selected Queries

Selected queries are listed in [Result Summary](#)

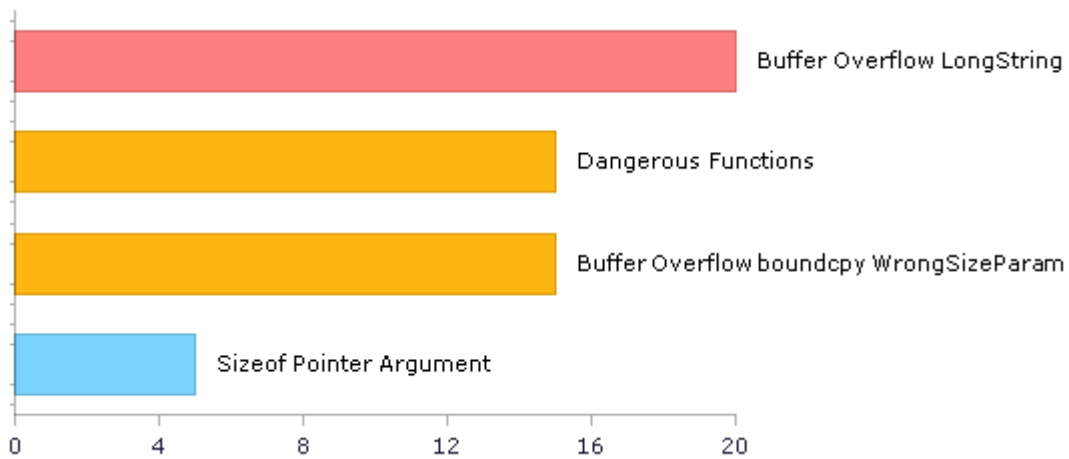
Result Summary



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	35	20
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	0	0
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	15	15
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	15	15
A10-Unvalidated Redirects and Forwards	USERS BROWSERS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0

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

Scan Summary - PCI DSS v3.2

Category	Issues Found	Best Fix Locations
PCI DSS (3.2) - 6.5.1 - Injection flaws - particularly SQL injection	0	0
PCI DSS (3.2) - 6.5.2 - Buffer overflows	35	20
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.	0	0
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.	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 - NIST SP 800-53

Category	Issues Found	Best Fix Locations
AC-12 Session Termination (P2)	0	0
AC-3 Access Enforcement (P1)	0	0
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)*	0	0
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*	20	5
SI-11 Error Handling (P2)*	0	0
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 wasn't 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

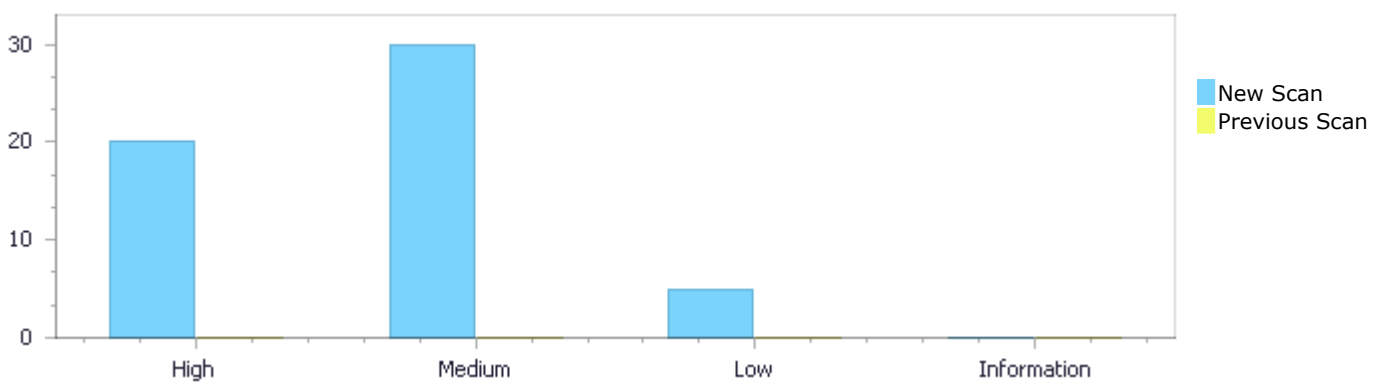
Category	Issues Found	Best Fix Locations
Must audit	0	0
Check	0	0
Optional	0	0

Results Distribution By Status

First scan of the project

	High	Medium	Low	Information	Total
New Issues	20	30	5	0	55
Recurrent Issues	0	0	0	0	0
Total	20	30	5	0	55

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	20	30	5	0	55
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	20	30	5	0	55

Result Summary

Vulnerability Type	Occurrences	Severity
Buffer Overflow LongString	20	High
Buffer Overflow boundcpy WrongSizeParam	15	Medium
Dangerous Functions	15	Medium
Sizeof Pointer Argument	5	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	10
php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	10
php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	10
php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	10
php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	10

Scan Results Details

Buffer Overflow LongString

Query Path:

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

Categories

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

Description

Buffer Overflow LongString\Path 1:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=1
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xc1\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Line	827	602
Object	"8b \xd0\xc1\xd2\xcf\xcc\xd8"	tmp

Code Snippet

File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
827.         const char *test_key = "8b \xd0\xc1\xd2\xcf\xcc\xd8";
```



File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
602.         tmp[0] |= (unsigned char)*ptr; /* correct */
```

Buffer Overflow LongString\Path 2:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=1

Status	046&pathid=2 New
--------	---

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xc1\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Line	827	604
Object	"8b \xd0\xc1\xd2\xcf\xcc\xd8"	tmp

Code Snippet

File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
827.         const char *test_key = "8b \xd0\xc1\xd2\xcf\xcc\xd8";
```



File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
604.         tmp[1] |= (BF_word_signed)(signed char)*ptr; /*
bug */
```

Buffer Overflow LongString\Path 3:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=3
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Line	869	604
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
869.          const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"
"345";
```



File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
604.          tmp[1] |= (BF_word_signed)(signed char)*ptr; /*
bug */
```

Buffer Overflow LongString\Path 4:

Severity High

Result State To Verify

Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=4>

Status New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Line	869	602
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
869.          const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"
"345";
```



File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
602.          tmp[0] |= (unsigned char)*ptr; /* correct */
```

Buffer Overflow LongString\Path 5:

Severity High

Result State To Verify

Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=5
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xc1\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	827	604
Object	"8b \xd0\xc1\xd2\xcf\xcc\xd8"	tmp

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
 Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
827.         const char *test_key = "8b \xd0\xc1\xd2\xcf\xcc\xd8";
```

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
 Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
604.         tmp[1] |= (BF_word_signed)(signed char)*ptr; /*
bug */
```

Buffer Overflow LongString\Path 6:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=6
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xc1\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	827	602
Object	"8b \xd0\xc1\xd2\xcf\xcc\xd8"	tmp

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
827.          const char *test_key = "8b \xd0\xcl\xd2\xcf\xcc\xd8";
```



File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....  
602.          tmp[0] |= (unsigned char)*ptr; /* correct */
```

Buffer Overflow LongString\Path 7:

Severity High
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=7>
Status New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	869	604
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
869.          const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"  
"345";
```



File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....  
604.          tmp[1] |= (BF_word_signed)(signed char)*ptr; /*  
bug */
```

Buffer Overflow LongString\Path 8:

Severity High

Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=8
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	869	602
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
869.             const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"
"345";
```

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
602.             tmp[0] |= (unsigned char)*ptr; /* correct */
```

Buffer Overflow LongString\Path 9:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=9
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xc1\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	827	604
Object	"8b \xd0\xc1\xd2\xcf\xcc\xd8"	tmp

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
827.          const char *test_key = "8b \xd0\xcl\xd2\xcf\xcc\xd8";
```

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
604.          tmp[1] |= (BF_word_signed)(signed char)*ptr; /*
bug */
```

Buffer Overflow LongString\Path 10:

Severity High

Result State To Verify

Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=10>

Status New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xcl\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	827	602
Object	"8b \xd0\xcl\xd2\xcf\xcc\xd8"	tmp

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
827.          const char *test_key = "8b \xd0\xcl\xd2\xcf\xcc\xd8";
```

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
602.          tmp[0] |= (unsigned char)*ptr; /* correct */
```

Buffer Overflow LongString\Path 11:

Severity High

Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=11
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	869	602
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
 Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
869.             const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"
"345";
```

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
 Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
602.             tmp[0] |= (unsigned char)*ptr; /* correct */
```

Buffer Overflow LongString\Path 12:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=12
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	869	604
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
869.             const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"
"345";
```

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
604.             tmp[1] |= (BF_word_signed)(signed char)*ptr; /*
bug */
```

Buffer Overflow LongString\Path 13:

Severity High

Result State To Verify

Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=13>

Status New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xc1\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	827	604
Object	"8b \xd0\xc1\xd2\xcf\xcc\xd8"	tmp

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
827.             const char *test_key = "8b \xd0\xc1\xd2\xcf\xcc\xd8";
```

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
604.             tmp[1] |= (BF_word_signed)(signed char)*ptr; /*
bug */
```

Buffer Overflow LongString\Path 14:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=14
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xc1\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	827	602
Object	"8b \xd0\xc1\xd2\xcf\xcc\xd8"	tmp

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
827.          const char *test_key = "8b \xd0\xc1\xd2\xcf\xcc\xd8";
```



File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....  
602.          tmp[0] |= (unsigned char)*ptr; /* correct */
```

Buffer Overflow LongString\Path 15:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=15
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	869	604
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
869.             const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"
"345";
```

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
604.             tmp[1] |= (BF_word_signed)(signed char)*ptr; /*
bug */
```

Buffer Overflow LongString\Path 16:

Severity High

Result State To Verify

Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=16>

Status New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	869	602
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
869.             const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"
"345";
```

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
602.             tmp[0] |= (unsigned char)*ptr; /* correct */
```


Buffer Overflow LongString\Path 17:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=17
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xc1\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	827	602
Object	"8b \xd0\xc1\xd2\xcf\xcc\xd8"	tmp

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
827.          const char *test_key = "8b \xd0\xc1\xd2\xcf\xcc\xd8";
```



File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....  
602.          tmp[0] |= (unsigned char)*ptr; /* correct */
```

Buffer Overflow LongString\Path 18:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=18
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "8b \xd0\xc1\xd2\xcf\xcc\xd8", at line 824 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	827	604

Object	"8b \xd0\xc1\xd2\xcf\xcc\xd8"	tmp
--------	-------------------------------	-----

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
827.         const char *test_key = "8b \xd0\xc1\xd2\xcf\xcc\xd8";
```

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
....
604.         tmp[1] |= (BF_word_signed)(signed char)*ptr; /*
bug */
```

Buffer Overflow LongString\Path 19:

Severity High

Result State To Verify

Online Results <http://WIN-PJTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=19>

Status New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	869	602
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
869.         const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"
"345";
```

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c

Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
.....
602.                                tmp[0] |= (unsigned char)*ptr; /* correct */
```

Buffer Overflow LongString\Path 20:

Severity	High
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=20
Status	New

The size of the buffer used by BF_set_key in tmp, at line 551 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to "\xff\xa3", at line 824 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	869	604
Object	"\xff\xa3"	tmp

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
.....
869.                                const char *k = "\xff\xa3" "34" "\xff\xff\xff\xa3"
"345";
```

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method static void BF_set_key(const char *key, BF_key expanded, BF_key initial,

```
.....
604.                                tmp[1] |= (BF_word_signed)(signed char)*ptr; /*
bug */
```

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-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=21
Status	New

The size of the buffer used by `*php_crypt_blowfish_rn` in Namespace1438664100, at line 824 of `php@@php-src-php-7.4.11-CVE-2020-1916-TP.c`, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that `*php_crypt_blowfish_rn` passes to Namespace1438664100, at line 824 of `php@@php-src-php-7.4.11-CVE-2020-1916-TP.c`, to overwrite the target buffer.

	Source	Destination
File	<code>php@@php-src-php-7.4.11-CVE-2020-1916-TP.c</code>	<code>php@@php-src-php-7.4.11-CVE-2020-1916-TP.c</code>
Line	853	853
Object	Namespace1438664100	Namespace1438664100

Code Snippet

File Name `php@@php-src-php-7.4.11-CVE-2020-1916-TP.c`
 Method `char *php_crypt_blowfish_rn(const char *key, const char *setting,`

```
....
853.         memcpy(buf.s, test_setting, sizeof(buf.s));
```

Buffer Overflow boundcpy WrongSizeParam\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=22
Status	New

The size of the buffer used by `*BF_crypt` in Namespace1438664100, at line 657 of `php@@php-src-php-7.4.11-CVE-2020-1916-TP.c`, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that `*BF_crypt` passes to Namespace1438664100, at line 657 of `php@@php-src-php-7.4.11-CVE-2020-1916-TP.c`, to overwrite the target buffer.

	Source	Destination
File	<code>php@@php-src-php-7.4.11-CVE-2020-1916-TP.c</code>	<code>php@@php-src-php-7.4.11-CVE-2020-1916-TP.c</code>
Line	706	706
Object	Namespace1438664100	Namespace1438664100

Code Snippet

File Name `php@@php-src-php-7.4.11-CVE-2020-1916-TP.c`
 Method `static char *BF_crypt(const char *key, const char *setting,`

```
....
706.         memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Buffer Overflow boundcpy WrongSizeParam\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=23
Status	New

The size of the buffer used by *php_crypt_blowfish_rn in Namespace615832523, at line 824 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to Namespace615832523, at line 824 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	853	853
Object	Namespace615832523	Namespace615832523

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
853.      memcpy(buf.s, test_setting, sizeof(buf.s));
```

Buffer Overflow boundcpy WrongSizeParam\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=24
Status	New

The size of the buffer used by *BF_crypt in Namespace615832523, at line 657 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *BF_crypt passes to Namespace615832523, at line 657 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	706	706
Object	Namespace615832523	Namespace615832523

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
706.      memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Buffer Overflow boundcpy WrongSizeParam\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=25
Status	New

The size of the buffer used by *php_crypt_blowfish_rn in Namespace1161312550, at line 824 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to Namespace1161312550, at line 824 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	853	853
Object	Namespace1161312550	Namespace1161312550

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
853.         memcpy(buf.s, test_setting, sizeof(buf.s));
```

Buffer Overflow boundcpy WrongSizeParam\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=26
Status	New

The size of the buffer used by *BF_crypt in Namespace1161312550, at line 657 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *BF_crypt passes to Namespace1161312550, at line 657 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	706	706
Object	Namespace1161312550	Namespace1161312550

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....
706.         memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Buffer Overflow boundcpy WrongSizeParam\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=27
Status	New

The size of the buffer used by *php_crypt_blowfish_rn in Namespace321847790, at line 824 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to Namespace321847790, at line 824 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	853	853
Object	Namespace321847790	Namespace321847790

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
 Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
853.         memcpy(buf.s, test_setting, sizeof(buf.s));
```

Buffer Overflow boundcpy WrongSizeParam\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=28
Status	New

The size of the buffer used by *BF_crypt in Namespace321847790, at line 657 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *BF_crypt passes to Namespace321847790, at line 657 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	706	706
Object	Namespace321847790	Namespace321847790

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
706.          memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Buffer Overflow boundcpy WrongSizeParam\Path 9:

Severity Medium
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=29>
Status New

The size of the buffer used by *php_crypt_blowfish_rn in Namespace190471419, at line 824 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to Namespace190471419, at line 824 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	853	853
Object	Namespace190471419	Namespace190471419

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
853.          memcpy(buf.s, test_setting, sizeof(buf.s));
```

Buffer Overflow boundcpy WrongSizeParam\Path 10:

Severity Medium
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=30>
Status New

The size of the buffer used by *BF_crypt in Namespace190471419, at line 657 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *BF_crypt passes to Namespace190471419, at line 657 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	706	706
Object	Namespace190471419	Namespace190471419

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
706.          memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Buffer Overflow boundcpy WrongSizeParam\Path 11:

Severity Medium
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=31>
Status New

The size of the buffer used by *php_crypt_blowfish_rn in Namespace1438664100, at line 824 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to Namespace1438664100, at line 824 of php@@php-src-php-7.4.11-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Line	860	860
Object	Namespace1438664100	Namespace1438664100

Code Snippet

File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
860.          memset(buf.o, 0x55, sizeof(buf.o));
```

Buffer Overflow boundcpy WrongSizeParam\Path 12:

Severity Medium
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=32>
Status New

The size of the buffer used by *php_crypt_blowfish_rn in Namespace615832523, at line 824 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to Namespace615832523, at line 824 of php@@php-src-php-7.4.14-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c

Line	860	860
Object	Namespace615832523	Namespace615832523

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
860.          memset(buf.o, 0x55, sizeof(buf.o));
```

Buffer Overflow boundcpy WrongSizeParam\Path 13:

Severity Medium

Result State To Verify

Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=33>

Status New

The size of the buffer used by *php_crypt_blowfish_rn in Namespace1161312550, at line 824 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to Namespace1161312550, at line 824 of php@@php-src-php-7.4.22-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	860	860
Object	Namespace1161312550	Namespace1161312550

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
860.          memset(buf.o, 0x55, sizeof(buf.o));
```

Buffer Overflow boundcpy WrongSizeParam\Path 14:

Severity Medium

Result State To Verify

Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=34>

Status New

The size of the buffer used by *php_crypt_blowfish_rn in Namespace321847790, at line 824 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to Namespace321847790, at line 824 of php@@php-src-php-7.4.26-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	860	860
Object	Namespace321847790	Namespace321847790

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
 Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
860.          memset(buf.o, 0x55, sizeof(buf.o));
```

Buffer Overflow boundcpy WrongSizeParam\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=35
Status	New

The size of the buffer used by *php_crypt_blowfish_rn in Namespace190471419, at line 824 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that *php_crypt_blowfish_rn passes to Namespace190471419, at line 824 of php@@php-src-php-7.4.7-CVE-2020-1916-TP.c, to overwrite the target buffer.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	860	860
Object	Namespace190471419	Namespace190471419

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
 Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....
860.          memset(buf.o, 0x55, sizeof(buf.o));
```

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-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=41
Status	New

The dangerous function, memcpy, was found in use at line 824 in php@@php-src-php-7.4.11-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Line	853	853
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
853.      memcpy(buf.s, test_setting, sizeof(buf.s));
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=42
Status	New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.11-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Line	706	706
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
706.      memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Dangerous Functions\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=43
Status	New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.11-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Line	776	776
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
776.          memcpy(output, setting, 7 + 22 - 1);
```

Dangerous Functions\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=44
Status	New

The dangerous function, memcpy, was found in use at line 824 in php@@php-src-php-7.4.14-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	853	853
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
853.          memcpy(buf.s, test_setting, sizeof(buf.s));
```

Dangerous Functions\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=45
Status	New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.14-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	706	706
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
706.      memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Dangerous Functions\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=46
Status	New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.14-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	776	776
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
776.         memcpy(output, setting, 7 + 22 - 1);
```

Dangerous Functions\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=47
Status	New

The dangerous function, memcpy, was found in use at line 824 in php@@php-src-php-7.4.22-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	853	853
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
853.         memcpy(buf.s, test_setting, sizeof(buf.s));
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=48
Status	New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.22-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	706	706
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c

Method static char *BF_crypt(const char *key, const char *setting,

```
....  
706.          memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Dangerous Functions\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=49
Status	New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.22-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	776	776
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
776.          memcpy(output, setting, 7 + 22 - 1);
```

Dangerous Functions\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=50
Status	New

The dangerous function, memcpy, was found in use at line 824 in php@@php-src-php-7.4.26-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	853	853
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
853.          memcpy(buf.s, test_setting, sizeof(buf.s));
```

Dangerous Functions\Path 11:

Severity Medium
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=51>
Status New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.26-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	706	706
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
706.          memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Dangerous Functions\Path 12:

Severity Medium
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=52>
Status New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.26-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	776	776
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
776.         memcpy(output, setting, 7 + 22 - 1);
```

Dangerous Functions\Path 13:

Severity Medium
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=53>
Status New

The dangerous function, memcpy, was found in use at line 824 in php@@php-src-php-7.4.7-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	853	853
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
853.         memcpy(buf.s, test_setting, sizeof(buf.s));
```

Dangerous Functions\Path 14:

Severity Medium
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=54>
Status New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.7-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	706	706
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
706.          memcpy(data.ctx.S, BF_init_state.S, sizeof(data.ctx.S));
```

Dangerous Functions\Path 15:

Severity Medium
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=55>
Status New

The dangerous function, memcpy, was found in use at line 657 in php@@php-src-php-7.4.7-CVE-2020-1916-TP.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	776	776
Object	memcpy	memcpy

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method static char *BF_crypt(const char *key, const char *setting,

```
....  
776.          memcpy(output, setting, 7 + 22 - 1);
```

Sizeof Pointer Argument

Query Path:

CPP\Cx\CPP Low Visibility\Sizeof Pointer Argument Version:0

[Description](#)

Sizeof Pointer Argument\Path 1:

Severity Low
Result State To Verify
Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=36>
Status New

	Source	Destination
File	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c	php@@php-src-php-7.4.11-CVE-2020-1916-TP.c
Line	876	876

Object	ai	sizeof
--------	----	--------

Code Snippet

File Name php@@php-src-php-7.4.11-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
876.                                !memcmp(ai, yi, sizeof(ai));
```

Sizeof Pointer Argument\Path 2:

Severity Low

Result State To Verify

Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=37>

Status New

	Source	Destination
File	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c	php@@php-src-php-7.4.14-CVE-2020-1916-TP.c
Line	876	876
Object	ai	sizeof

Code Snippet

File Name php@@php-src-php-7.4.14-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
....  
876.                                !memcmp(ai, yi, sizeof(ai));
```

Sizeof Pointer Argument\Path 3:

Severity Low

Result State To Verify

Online Results <http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=38>

Status New

	Source	Destination
File	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c	php@@php-src-php-7.4.22-CVE-2020-1916-TP.c
Line	876	876
Object	ai	sizeof

Code Snippet

File Name php@@php-src-php-7.4.22-CVE-2020-1916-TP.c

Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
.....  
876.                !memcmp(ai, yi, sizeof(ai));
```

Sizeof Pointer Argument\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=39
Status	New

	Source	Destination
File	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c	php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Line	876	876
Object	ai	sizeof

Code Snippet

File Name php@@php-src-php-7.4.26-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
.....  
876.                !memcmp(ai, yi, sizeof(ai));
```

Sizeof Pointer Argument\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-PTJMSNK3USL/CxWebClient/ViewerMain.aspx?scanid=1020057&projectid=20046&pathid=40
Status	New

	Source	Destination
File	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c	php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Line	876	876
Object	ai	sizeof

Code Snippet

File Name php@@php-src-php-7.4.7-CVE-2020-1916-TP.c
Method char *php_crypt_blowfish_rn(const char *key, const char *setting,

```
.....  
876.                !memcmp(ai, yi, sizeof(ai));
```

Buffer Overflow LongString

Risk

What might happen

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

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In its 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 (strlen(inputString, MAX_INPUT_SIZE) < sizeof(buffer))
    {
        strncpy(buffer, inputString, sizeof(buffer));
    }
}
```

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 its 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

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 sizeof() on a Pointer Type

Weakness ID: 467 (*Weakness Variant*)

Status: Draft

Description

Description Summary

The code calls sizeof() on a malloced pointer type, which always returns the wordsize/8. This can produce an unexpected result if the programmer intended to determine how much memory has been allocated.

Time of Introduction

Implementation

Applicable Platforms

Languages

C

C++

Common Consequences

Scope	Effect
Integrity	This error can often cause one to allocate a buffer that is much smaller than what is needed, leading to resultant weaknesses such as buffer overflows.

Likelihood of Exploit

High

Demonstrative Examples

Example 1

Care should be taken to ensure sizeof returns the size of the data structure itself, and not the size of the pointer to the data structure.

In this example, sizeof(foo) returns the size of the pointer.

(Bad Code)

Example Languages: C and C++

```
double *foo;
...
foo = (double *)malloc(sizeof(foo));
```

In this example, sizeof(*foo) returns the size of the data structure and not the size of the pointer.

(Good Code)

Example Languages: C and C++

```
double *foo;
...
foo = (double *)malloc(sizeof(*foo));
```

Example 2

This example defines a fixed username and password. The AuthenticateUser() function is intended to accept a username and a password from an untrusted user, and check to ensure that it matches the username and password. If the username and password match, AuthenticateUser() is intended to indicate that authentication succeeded.

(Bad Code)

/ Ignore CWE-259 (hard-coded password) and CWE-309 (use of password system for authentication) for this example. */*

```
char *username = "admin";
char *pass = "password";

int AuthenticateUser(char *inUser, char *inPass) {
```

```
printf("Sizeof username = %d\n", sizeof(username));
printf("Sizeof pass = %d\n", sizeof(pass));

if (strcmp(username, inUser, sizeof(username))) {
printf("Auth failure of username using sizeof\n");
return(AUTH_FAIL);
}
/* Because of CWE-467, the sizeof returns 4 on many platforms and architectures. */
if (! strcmp(pass, inPass, sizeof(pass))) {
printf("Auth success of password using sizeof\n");
return(AUTH_SUCCESS);
}
else {
printf("Auth fail of password using sizeof\n");
return(AUTH_FAIL);
}
}

int main (int argc, char **argv)
{
int authResult;

if (argc < 3) {
ExitError("Usage: Provide a username and password");
}
authResult = AuthenticateUser(argv[1], argv[2]);
if (authResult != AUTH_SUCCESS) {
ExitError("Authentication failed");
}
else {
DoAuthenticatedTask(argv[1]);
}
}
```

In `AuthenticateUser()`, because `sizeof()` is applied to a parameter with an array type, the `sizeof()` call might return 4 on many modern architectures. As a result, the `strcmp()` call only checks the first four characters of the input password, resulting in a partial comparison (CWE-187), leading to improper authentication (CWE-287).

Because of the partial comparison, any of these passwords would still cause authentication to succeed for the "admin" user:

(Attack)

```
pass5
passABCDEFGH
passWORD
```

Because only 4 characters are checked, this significantly reduces the search space for an attacker, making brute force attacks more feasible.

The same problem also applies to the username, so values such as "adminXYZ" and "administrator" will succeed for the username.

Potential Mitigations

Phase: Implementation

Use expressions such as "`sizeof(*pointer)`" instead of "`sizeof(pointer)`", unless you intend to run `sizeof()` on a pointer type to gain some platform independence or if you are allocating a variable on the stack.

Other Notes

The use of `sizeof()` on a pointer can sometimes generate useful information. An obvious case is to find out the wordsize on a platform. More often than not, the appearance of `sizeof(pointer)` indicates a bug.

Weakness Ordinalities

Ordinality	Description
Primary	<i>(where the weakness exists independent of other weaknesses)</i>

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	465	Pointer Issues	Development Concepts (primary)699
ChildOf	Weakness Class	682	Incorrect Calculation	Research Concepts (primary)1000
ChildOf	Category	737	CERT C Secure Coding Section 03 - Expressions (EXP)	Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
CanPrecede	Weakness Base	131	Incorrect Calculation of Buffer Size	Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Use of sizeof() on a pointer type
CERT C Secure Coding	ARR01-C		Do not apply the sizeof operator to a pointer when taking the size of an array
CERT C Secure Coding	EXP01-C		Do not take the size of a pointer to determine the size of the pointed-to type

White Box Definitions

A weakness where code path has:

1. end statement that passes an identity of a dynamically allocated memory resource to a sizeof operator
2. start statement that allocates the dynamically allocated memory resource

References

Robert Seacord. "EXP01-A. Do not take the sizeof a pointer to determine the size of a type".
<https://www.securecoding.cert.org/confluence/display/seccode/EXP01-A.+Do+not+take+the+sizeof+a+pointer+to+determine+the+size+of+a+type>.

Content History

Submissions			
Submission Date	Submitter	Organization	Source
	CLASP		Externally Mined
Modifications			
Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-01		KDM Analytics	External
	added/updated white box definitions		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

[BACK TO TOP](#)

Scanned Languages

Language	Hash Number	Change Date
CPP	4541647240435660	1/6/2025
Common	0105849645654507	1/6/2025