

shop.example.com

Пример отчёта · вымышленный интернет-магазин, все данные условные, это не реальный сайт

Дата: 2026-10-28

🛡 Проверено по базам на 28 окт. 2026 г.: NVD, OSV, WPVulnerability + список активно атакуемых уязвимостей (28 окт. 2026 г.)

32/100

Оценка (100 = меньше наблюдаемых рисков)

Возможные риски по публичным признакам. Обсудите приоритетные находки с разработчиком или хостингом. Проверки могут быть неполными; отчёт не даёт гарантий безопасности.

Светофор рисков

● **5** Критичные ⚡ из них 5 активно атакуются прямо сейчас

● **3** Высокие ⚡ из них 1 активно атакуется прямо сейчас

● **8** Средние

● **5** Низкие и информационные

Нажмите на строку — откроется список этих замечаний.

Что исправить в первую очередь

⚡ — Активно атакуется

1. **Критично** ⚡ PHP 8.4.26 уязвим к PHP2Shell — взлом сервера без пароля
2. **Критично** ⚡ Файл .env с паролями от базы, почты и платёжки открыт всем
3. **Критично** ⚡ Папка .git с исходным кодом сайта открыта всем
4. **Критично** ⚡ Секретный ключ базы данных или платежей (Supabase service role)
виден в коде сайта
5. **Критично** ⚡ Секретный ключ ИИ-сервиса (OpenAI)
виден в коде сайта

Критично ⚡ Активно атакуется

1. PHP 8.4.26 уязвим к PHP2Shell — взлом сервера без пароля

Почему опасно сейчас: С конца октября 2026 года боты массово взламывают сайты на PHP через PHP2Shell: без пароля запускают свой код на сервере и забирают базу клиентов. CISA внесла уязвимость в список реально атакуемых.

Что найдено

Программа	PHP
Компонент	php
Версия	8.4.26
Исправлено в версии	8.4.27
Уязвимости	CVE-2026-109618, CVE-2026-109619
Признак	X-Powered-By: PHP/8.4.26
Страница	https://shop.example.com/
Затронутو страниц	1
Уверенность	вероятно (нужно подтвердить)

Шаги исправления

1. Обновите PHP до исправленной версии своей ветки: 8.4.27, 8.3.36, 8.2.35 или 8.5.12 (для вас — 8.4.27 или новее). На хостинге это обычно пункт «Версия PHP» в панели, на своём сервере — обновление пакета php.
2. Если у вас PHP 8.1 или старше — исправления не будет: переходите на 8.3/8.4, а до перехода включите WAF (например, Cloudflare) с правилами против PHP2Shell.
3. Скройте версию PHP: в php.ini выставьте `expose_php = Off`.
4. Проверьте сервер на следы взлома: незнакомые .php-файлы в папках загрузок, новые администраторы, процессы php, которые запускают sh, curl или wget.
5. Как проверить: `curl -sI https://shop.example.com/ | grep -i x-powered-by` — версии PHP в ответе быть не должно, а `php -v` на сервере должен показать исправленную версию.

Примерное время: 30–90 минут; уточните у исполнителя

Источники: www.cisa.gov/news-events/alerts/2026/10/21/critical-vulnerabilities-php-require-immediate-patching · www.bleepingcomputer.com/news/security/php2shell-security-flaws-now-succesfully-exploited-in-the-wild/

► Технические детали

⚡ Активно атакуется

2. Файл .env с паролями от базы, почты и платёжки открыт всем

Почему опасно сейчас: Боты круглосуточно ищут открытые .env и .git: только одна кампания EmeraldWhale собрала так больше 15 000 облачных ключей, и такие запросы к сайтам идут до сих пор.

Что найдено

Путь	/.env
------	-------

Ответ сервера HTTP 200

Уверенность ТОЧНО

Шаги исправления

1. Сразу закройте доступ к скрытым файлам. nginx: `location ~ /\.(!well-known) { deny all; }`
2. Apache (.htaccess): `<FilesMatch "\.">Require all denied </FilesMatch>`
3. Перенесите .env выше папки сайта (public_html/www), чтобы веб-сервер его вообще не видел.
4. Смените ВСЕ пароли и ключи из файла: база данных, почта, платёжный шлюз, API-ключи. Считайте их уже украденными.
5. Поищите в логах запросы GET /.env — так вы поймёте, когда файл скачивали.
6. Как проверить: `curl -s -o /dev/null -w "%{http_code}" https://shop.example.com/.env` — должно быть 403 или 404.

Примерное время: 30–90 минут; уточните у исполнителя

Источники: www.sysdig.com/blog/emeraldwhale · honeylabs.net/blog/credential-scraping-env-and-git

► Технические детали

Критично

⚡ Активно атакуется

3. Папка .git с исходным кодом сайта открыта всем

Почему опасно сейчас: Из открытой .git злоумышленники за минуты скачивают весь код сайта вместе с паролями внутри — именно так кампания EmeraldWhale украла больше 15 000 облачных ключей.

Что найдено

Путь `/.git/config`

Ответ сервера HTTP 200

Уверенность точно

Шаги исправления

1. Закройте доступ к .git. nginx: `location ~ /\.git { deny all; }`
2. Apache (.htaccess): `RedirectMatch 404 /\.git`
3. Публикуйте сайт без папки .git (git archive, CI-сборка или `rsync --exclude .git`).
4. Смените пароли и ключи, которые когда-либо попадали в репозиторий, — история коммитов тоже утекла.
5. Как проверить: `curl -s -o /dev/null -w "%{http_code}" https://shop.example.com/.git/config` — должно быть 403 или 404.

Примерное время: 30–90 минут; уточните у исполнителя

Источники: www.sysdig.com/blog/emeraldwhale · honeylabs.net/blog/credential-scraping-env-and-git

► Технические детали

Критично

⚡ Активно атакуется

4. Секретный ключ базы данных или платежей (Supabase service_role) виден в коде сайта

Почему опасно сейчас: Такой ключ обходит все правила доступа: с ним любой посетитель читает и меняет всю базу — заказы, клиентов, пароли. Из-за похожей ошибки (CVE-2025-48757) открытыми оказались данные 170+ приложений, собранных в Lovable.

Что найдено

Сервис	Supabase service_role
Фрагмент ключа (скрыт)	eyJhbGciOi...[219 chars, redacted]
Где найдено	https://shop.example.com/app/assets/index-8c1d2f.js
Уверенность	вероятно (нужно подтвердить)

Шаги исправления

1. Сразу перевыпустите ключ: Supabase — Project Settings → API Keys (service_role / secret), Stripe — Developers → API keys. Старый ключ считайте украденным.
2. В браузере используйте только публичный ключ (anon / publishable); операции администратора — только на сервере (Edge Functions, API-роут).
3. Включите защиту строк на каждой таблице: `alter table <таблица> enable row level security`; и опишите правила, кто какие строки видит.
4. Проверьте журналы базы и платёжного кабинета на чужие запросы и выгрузки.
5. Как проверить: в JS-файлах сайта не должно быть строк `service_role`, `sb_secret_` или `sk_live_`.

Примерное время: 30–90 минут; уточните у исполнителя

Источники: mattpalmer.io/posts/2025/05/statement-on-CVE-2025-48757/ · supabase.com/docs/guides/database/postgres/row-level-security · escape.tech/blog/methodology-how-we-discovered-vulnerabilities-apps-built-with-vibe-coding/

► Технические детали

5. Секретный ключ ИИ-сервиса (OpenAI) виден в коде сайта

Почему опасно сейчас: Сайты и чат-боты, собранные с помощью AI, часто кладут ключ прямо в JavaScript; боты собирают такие ключи автоматически и сжигают ваш баланс за часы. В исследовании Escape в 5 600 таких приложений нашли 400+ открытых секретов.

Что найдено

Сервис	OpenAI
Фрагмент ключа (скрыт)	sk-proj-Xa...[164 chars, redacted]
Где найдено	https://shop.example.com/wp-content/plugins/shop-ai-assistant/assets/chat.js
Уверенность	вероятно (нужно подтвердить)

Шаги исправления

1. Отзовите ключ прямо сейчас в кабинете провайдера и создайте новый — старый считайте украденным.
2. Поставьте лимит расходов в кабинете провайдера.
3. Перенесите запросы к AI на сервер (серверная функция или API-роут): новый ключ хранится только в переменных окружения сервера и никогда не попадает в браузер.
4. Проверьте в кабинете провайдера, не было ли чужих запросов и расходов.
5. Как проверить: после пересборки поищите в JS-файлах сайта префикс ключа (например sk-) — совпадений быть не должно.

Примерное время: 30–90 минут; уточните у исполнителя

Источники: escape.tech/blog/methodology-how-we-discovered-vulnerabilities-apps-built-with-vibe-coding/

► Технические детали

Высокий

 Активно атакуется

6. Плагин Ninja Forms 3.15.2: уязвимость уже используют для взлома сайтов (CVE-2026-94504)

Почему опасно сейчас: В октябре 2026 года эту уязвимость Ninja Forms массово атакуют: взломщик отправляет через форму на сайте заявку с вредоносным кодом, а когда администратор открывает её в панели WordPress, код срабатывает с правами админа — так создают скрытых администраторов и ставят бэкдор. Исправлено в версии 3.15.4.

Что найдено

Программа	WordPress plugin ninja-forms
Компонент	ninja-forms
Версия	3.15.2
Где найдено	WPVulnerability
Всего уязвимостей	1
Уязвимости	CVE-2026-94504
Исправлено в версии	3.15.4
Признак	readme.txt Stable tag 3.15.2
Уверенность	точно
Главная уязвимость	CVE-2026-94504

Шаги исправления

1. До обновления не открывайте новые заявки в Ninja Forms → Submissions.
2. Обновите плагин до 3.15.4 или новее: Консоль → Плагины → «Обновить». Если обновления нет — отключите плагин.
3. Проверьте Пользователи → Администраторы: удалите незнакомые учётные записи и смените пароли всех админов.
4. Проверьте список плагинов и папку wp-content/uploads на незнакомые файлы; при следах взлома восстановите сайт из чистой копии.
5. Включите автообновление этого плагина и двухфакторный вход для админов.
6. Как проверить: на странице «Плагины» версия не ниже исправленной, повторная проверка SiteCheck больше не показывает это замечание.

Примерное время: 30–90 минут; уточните у исполнителя

Источники: www.bleepingcomputer.com/news/security/ninja-forms-plugin-flaw-exploited-to-hack-wordpress-sites/

Высокий

7. Сайт обращается к ИИ-сервису прямо из браузера

Что найдено

Адреса API OpenAI

Где найдено `https://shop.example.com/wp-content/plugins/shop-ai-assistant/assets/chat.js`

Уверенность вероятно (нужно подтвердить)

Шаги исправления

1. Пропускайте запросы к ИИ через свой сервер: храните ключ там, ограничьте передаваемые данные и вырезайте персональные данные до отправки.

Примерное время: 30–90 минут; уточните у исполнителя

Высокий

8. HTTP-версия сайта не перенаправляет на HTTPS

Что найдено

Адрес после перехода `http://shop.example.com/`

Ответ сервера HTTP 200

Затронуто страниц 1

Уверенность точно

Шаги исправления

1. Зачем: Посетители, набравшие адрес без `https`, работают по незащищённому соединению.
2. Настройте постоянный редирект 301 со всех HTTP-адресов на HTTPS (пример ниже).
3. nginx (server { listen 80; }): `return 301 https://$host$request_uri;`
4. Apache: `RewriteEngine On / RewriteCond %{HTTPS} off / RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]`
5. Как проверить: `curl -sI http://shop.example.com/ | grep -i location`

Примерное время: 30–90 минут; уточните у исполнителя

Средний

9. Нет заголовка Content-Security-Policy

Что найдено

Страница	https://shop.example.com/
Заголовок	content-security-policy
Страницы	https://shop.example.com/, https://shop.example.com/catalog/, https://shop.example.com/cart/
Затронуто страниц	3
Уверенность	точно

Шаги исправления

1. Зачем: CSP разрешает браузеру запускать скрипты только с ваших и доверенных доменов — главный барьер против внедрения чужого кода (XSS, подмена счётчика или чата).
2. Соберите домены, с которых сайт грузит скрипты, стили и фреймы (счётчики, чат, CDN): DevTools → Network.
3. Включите политику в режиме Content-Security-Policy-Report-Only (пример ниже), 1–2 недели смотрите нарушения в консоли браузера и дополняйте список доменов.
4. Когда нарушений нет, переименуйте заголовок в Content-Security-Policy.
5. nginx (server {}): `add_header Content-Security-Policy-Report-Only "default-src 'self'; script-src 'self' https://<домены-скриптов>; object-src 'none'; base-uri 'self'; frame-ancestors 'self'" always;`
6. Apache (.htaccess): `Header always set Content-Security-Policy-Report-Only "default-src 'self'; script-src 'self' https://<домены-скриптов>; object-src 'none'; base-uri 'self'; frame-ancestors 'self'"`
7. Как проверить: `curl -sI https://shop.example.com/ | grep -i content-security-policy`

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

10. Нет заголовка Strict-Transport-Security

Что найдено

Страница	https://shop.example.com/
Заголовок	strict-transport-security
Страницы	https://shop.example.com/, https://shop.example.com/catalog/, https://shop.example.com/cart/
Затронуто страниц	3
Уверенность	точно

Шаги исправления

1. Зачем: HSTS заставляет браузер всегда открывать сайт по HTTPS и не даёт перехватить первый HTTP-запрос в публичном Wi-Fi.
2. Убедитесь, что сайт и все поддомены открываются по HTTPS.
3. Начните с max-age=86400, через неделю без проблем увеличьте до 31536000 (пример ниже).
4. nginx (server {}):
`add_header Strict-Transport-Security "max-age=31536000;
includeSubDomains" always;`
5. Apache (.htaccess):
`Header always set Strict-Transport-Security "max-age=31536000;
includeSubDomains"`
6. Как проверить: `curl -sI https://shop.example.com/ | grep -i strict-transport-security`

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Средний

11. Нет заголовка X-Content-Type-Options

Что найдено

Страница	https://shop.example.com/
Заголовок	x-content-type-options
Страницы	https://shop.example.com/, https://shop.example.com/catalog/, https://shop.example.com/cart/
Затронуто страниц	3
Уверенность	точно

Шаги исправления

1. Зачем: Без nosniff браузер может «угадать» тип файла и выполнить загруженный пользователем файл как скрипт.
2. Отдавайте X-Content-Type-Options: nosniff во всех ответах сайта.
3. nginx (server {}): add_header X-Content-Type-Options "nosniff" always;
4. Apache (.htaccess): Header always set X-Content-Type-Options "nosniff"
5. Как проверить: curl -sI https://shop.example.com/ | grep -i x-content-type-options

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Средний

12. Нет защиты от встраивания в чужие сайты (X-Frame-Options / frame-ancestors)

Что найдено

Страница	https://shop.example.com/
Заголовок	x-frame-options or CSP frame-ancestors
Затронуто страниц	1
Уверенность	точно

Шаги исправления

1. Зачем: Без запрета сайт можно встроить в чужую страницу и обманом заставить посетителя нажать вашу кнопку (кликджекинг).
2. Отдавайте X-Frame-Options: SAMEORIGIN или добавьте frame-ancestors 'self' в Content-Security-Policy.
3. nginx (server {}): add_header X-Frame-Options "SAMEORIGIN" always;
4. Apache (.htaccess): Header always set X-Frame-Options "SAMEORIGIN"
5. Как проверить: curl -sI https://shop.example.com/ | grep -i x-frame-options

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

13. Cookie без флагов httponly, samesite: woocommerce_session

Что найдено

Страница	https://shop.example.com/
Не хватает	httponly, samesite
Cookie	woocommerce_session
Cookie	woocommerce_session
Затронуто страниц	1
Уверенность	точно

Шаги исправления

1. Зачем: Без этих флагов cookie сессии может уйти по незащищённому соединению или быть прочитан внедрённым скриптом.
2. Для cookie woocommerce_session добавьте недостающие флаги: httponly, samesite.
3. Если cookie ставит сторонний сервис (аналитика, чат, CDN), флаги настраиваются у него — сверьте домен cookie в DevTools → Application → Cookies.
4. PHP: `session_set_cookie_params(['secure' => true, 'httponly' => true, 'samesite' => 'Lax']);`
5. nginx ≥ 1.19.3 (proxy): `proxy_cookie_flags ~ secure httponly samesite=lax;`
6. Как проверить: `curl -sI https://shop.example.com/ | grep -i set-cookie`

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Средний

14. Нет DMARC-записи для shop.example.com

Что найдено

Домен `_dmarc.shop.example.com`

Уверенность `точно`

Шаги исправления

1. Зачем: DMARC говорит почтовым сервисам, что делать с поддельными письмами от вашего имени, и присылает отчёты о них.
2. Опубликуйте запись с `p=none` и адресом для отчётов (пример ниже).
3. Через 2–4 недели, если в отчётах только ваши серверы, перейдите на `p=quarantine`.
4. `_dmarc.shop.example.com`. TXT "`v=DMARC1; p=none; rua=mailto:dmarc@shop.example.com`"
5. Как проверить: `dig +short TXT _dmarc.shop.example.com`

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Средний

15. Нет SPF-записи для shop.example.com

Что найдено

Домен `shop.example.com`

Уверенность `точно`

Шаги исправления

1. Зачем: Без SPF письма от имени вашего домена легко подделать, а настоящие письма чаще попадают в спам.
2. Узнайте у почтового провайдера его include (Яндекс 360: `_spf.yandex.net`, Google: `_spf.google.com`).
3. Добавьте TXT-запись в DNS (пример для Яндекс 360 ниже); запись SPF должна быть одна.
4. `shop.example.com`. TXT "`v=spf1 include:_spf.yandex.net ~all`"
5. Как проверить: `dig +short TXT shop.example.com`

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Средний

16. Известные уязвимости в jquery 1.12.4: CVE-2020-11022

Что найдено

Программа	jquery
Версия	1.12.4
Уверенность	точно
Главная уязвимость	CVE-2020-11022
Оценка опасности (CVSS)	6.1

Шаги исправления

1. Зачем: Для этой версии опубликованы уязвимости — автоматические сканеры ищут такие сайты в первую очередь.
2. Сверьте установленную версию jquery 1.12.4 с диапазоном затронутых версий (CVE-2020-11022).
3. Сделайте резервную копию и обновите jquery 1.12.4 до версии с исправлением.
4. Как проверить: <https://nvd.nist.gov/vuln/detail/CVE-2020-11022>

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Низкий

17. Заголовок Server раскрывает ПО и версию: «Apache/2.4.58 (Ubuntu)»

Что найдено

Заголовок	server
Значение	Apache/2.4.58 (Ubuntu)
Уверенность	точно

Шаги исправления

1. Зачем: Точная версия сервера помогает автоматическим сканерам подобрать известные уязвимости.
2. Скройте версию в ответах веб-сервера (пример ниже) и обновите сервер, если версия устарела.
3. nginx (http {}): server_tokens off;
4. Apache: ServerTokens Prod и ServerSignature Off
5. Как проверить: `curl -sI https://shop.example.com/ | grep -i '^server'`

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Низкий

18. Заголовок X-Powered-By раскрывает платформу: «PHP/8.4.26»

Что найдено

Заголовок x-powered-by

Значение PHP/8.4.26

Уверенность точно

Шаги исправления

1. Зачем: Название и версия платформы упрощают подбор известных уязвимостей.
2. Отключите заголовок в настройках платформы или скройте его на веб-сервере (примеры ниже).
3. PHP (php.ini): `expose_php = Off`
4. nginx: `fastcgi_hide_header X-Powered-By; proxy_hide_header X-Powered-By;`
5. Express: `app.disable('x-powered-by')`
6. Как проверить: `curl -sI https://shop.example.com/ | grep -i x-powered-by`

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Низкий

19. Нет файла /.well-known/security.txt

Что найдено

Ответ сервера HTTP 404

Уверенность точно

Шаги исправления

1. Зачем: Файл подсказывает исследователям, куда сообщить о найденной уязвимости, — вы узнаете о проблеме раньше злоумышленников.
2. Создайте файл `/.well-known/security.txt` с контактом и сроком действия (пример ниже).
3. Contact: `mailto:security@shop.example.com`
4. Expires: `2027-12-31T23:59:59.000Z`
5. Как проверить: `curl -s https://shop.example.com/.well-known/security.txt`

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Низкий

20. Нет CAA-записи для shop.example.com

Что найдено

Домен shop.example.com

Уверенность точно

Шаги исправления

1. Зачем: CAA разрешает выпуск сертификатов для домена только выбранным центрам сертификации.
2. Добавьте CAA-запись для центра, который выпускает ваш сертификат (пример для Let's Encrypt ниже).
3. shop.example.com. CAA 0 issue "letsencrypt.org"
4. Как проверить: dig +short CAA shop.example.com

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

Инфо

21. Публичный признак CMS

Что найдено

Страница https://shop.example.com/

Программа WordPress

Затронуты страниц 1

Уверенность точно

Шаги исправления

1. Подтвердите CMS и обновляйте ядро, темы и плагины официальным механизмом обновления.

Примерное время: 30–90 минут; уточните у исполнителя

► Технические детали

NVD

This product uses the NVD API but is not endorsed or certified by the NVD.

неинвазивная проверка по публичным данным. This product uses the NVD API but is not endorsed or certified by the NVD. Этот продукт использует NVD API, но не одобрен и не сертифицирован NVD.