Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

WAF на ModSecurity для корпоративного сайта

Связка libmodsecurity 3.0.15 + OWASP CRS 4 на Nginx: от DetectionOnly до боевой блокировки

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

Формы поиска, заказа и личные кабинеты корпоративного сайта круглосуточно сканирует автоматика: SQL-инъекции, XSS, перебор админок. Одна незакрытая форма — и таблица с клиентской базой уходит в дамп. Мы закрываем этот слой WAF-ом на ModSecurity v3 с OWASP CRS 4 прямо на сервере клиента: трафик и платёжные данные не покидают периметр, а атака блокируется до кода приложения.

Почему это важно бизнесу

- Утечка клиентской базы через одну форму = оборотные штрафы по 152-ФЗ, претензии клиентов и разбор с регулятором
- Взломанный сайт попадает в чёрные списки браузеров и поисковиков — трафик и заказы падают на недели
- Облачные WAF терминируют TLS у себя: платёжные и персональные данные уходят на чужую инфраструктуру
- Лицензии коммерческих WAF — сотни тысяч в год; ModSecurity закрывает те же классы атак ценой работы инженера

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

3.0.15

движок libmodsecurity; ниже этой версии не ставим — релиз закрывает DoS CVE-2026-30923 и CVE-2026-42268

ModSecurity v3, 28.04.2026

4.28.0

актуальный OWASP Core Rule Set; для консервативных прод-сред держим LTS-ветку 4.25

coreruleset.org

PL2

tx.blocking_paranoia_level для сайтов с личным кабинетом и платёжными данными (правило 900000)

crs-setup.conf, CRS 4

5 / 4

пороги inbound/outbound anomaly score (правило 900110): одно критичное правило даёт 5 баллов

crs-setup.conf, CRS 4

7 дней

выдержка в DetectionOnly до включения блокировки: за неделю снимаем ложные срабатывания

наш стандарт

10/300/3600

fail2ban: maxretry/findtime/bantime — бан IP на час после 10 блокированных запросов за 5 минут

наш стандарт

РЕАЛИЗАЦИЯ 1 ИЗ 3

Сборка WAF: libmodsecurity + коннектор в Nginx

Что настраиваем

Боевой веб-сервер Nginx 1.28.3 — WAF на том же хосте, без выноса трафика наружу

Как мы это делаем

1. Собираем libmodsecurity 3.0.15 и динамический модуль ModSecurity-nginx 1.0.4 строго под установленную версию Nginx; подключаем через load_module в nginx.conf
2. Базовый конфиг — из modsecurity.conf-recommended: SecRuleEngine DetectionOnly, SecRequestBodyLimit 13107200, SecAuditLogParts ABIJDEFHZ
3. Разворачиваем CRS 4: crs-setup.conf + Include rules/*.conf; свои правки — только в файлах REQUEST-900-EXCLUSION…BEFORE-CRS и RESPONSE-999-…AFTER-CRS
4. Включаем modsecurity on и modsecurity_rules_file на уровне server{}; проверяем nginx -t и контрольным запросом с тестовой сигнатурой

РЕЗУЛЬТАТ

WAF встраивается в конвейер обработки запросов Nginx без отдельного прокси и лишней точки отказа; весь трафик, включая платёжные данные, остаётся на сервере клиента.

КЛЮЧЕВОЙ НЮАНС

Коннектор — динамический модуль, бинарно привязанный к версии Nginx: любой апгрейд Nginx = пересборка ModSecurity-nginx, иначе сервис не стартует. Сборочный скрипт храним рядом с конфигом.

РЕАЛИЗАЦИЯ 2 ИЗ 3

Тюнинг CRS: от наблюдения к блокировке

Что настраиваем

Формы поиска и заказа, личный кабинет, админка, вебхуки эквайринга

Как мы это делаем

1. Неделю работаем в DetectionOnly: разбираем audit-лог, группируем срабатывания по id правила и URI, отделяем атаки от легитимного трафика
2. Ложные срабатывания снимаем адресно: ctl:ruleRemoveTargetById=942100;ARGS:q под конкретную форму, а не SecRuleRemoveById по всему сайту
3. Вебхуки платёжных сервисов выносим в отдельный location с allowlist по IP-диапазонам шлюза — их крупные JSON не гоняем через полный набор правил
4. Фиксируем tx.blocking_paranoia_level=2, пороги 5/4, включаем tx.early_blocking=1 (правило 900120) — блок до передачи запроса приложению
5. Переводим SecRuleEngine в On; первые три дня дежурный инженер следит за жалобами и логом блокировок

РЕЗУЛЬТАТ

Блокировка включается без потерь заказов: оценка по сумме баллов (правило 949110) отсекает атаки, а точечные исключения сохраняют работу поиска и корзины.

КЛЮЧЕВОЙ НЮАНС

CRS блокирует не по одному совпадению, а по сумме anomaly score, поэтому порог и paranoia level настраиваем парой: PL2 с порогом 5 строже PL1, но живёт без лавины ложных блокировок.

РЕАЛИЗАЦИЯ 3 ИЗ 3

Эксплуатация: fail2ban, логи, обновления

Что настраиваем

Тот же веб-узел + Telegram-канал дежурной смены

Как мы это делаем

1. fail2ban-jail по audit-логу ModSecurity: maxretry=10, findtime=300, bantime=3600; бан на уровне nftables — повторные запросы не доходят до анализа
2. logrotate на modsec_audit.log: daily, 14 ротаций, сжатие — при активном сканировании лог растёт на гигабайты в неделю
3. Обновляем CRS раз в месяц по тегам релизов с чтением changelog; новые правила неделю обкатываем через tx.detection_paranoia_level выше боевого
4. Критичные срабатывания и баны шлём в Telegram; раз в неделю инженер делает сводку топ-правил и топ-IP

РЕЗУЛЬТАТ

WAF остаётся боевым и через полгода: правила актуальны, диск не переполняется, атаки видит дежурная смена, а не аудитор после инцидента.

КЛЮЧЕВОЙ НЮАНС

Пара blocking/detection_paranoia_level — штатный механизм CRS 4: блокируем на PL2, наблюдаем на PL3 — видим атаки следующего уровня до включения их в блок.

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

✕ Блокировка с первого дня

SecRuleEngine On без обкатки ломает легитимные формы. Всегда стартуем в DetectionOnly и неделю разбираем audit-лог до включения блока.

✕ Правила отключаются целиком

SecRuleRemoveById глушит защиту на всём сайте. Ложные срабатывания снимаем через ruleRemoveTargetById под конкретный параметр и URI.

✕ Nginx обновили — WAF отвалился

Динамический модуль коннектора собран под конкретный бинарь Nginx; после apt upgrade пересобираем ModSecurity-nginx до перезапуска.

✕ Вебхуки эквайринга под общими правилами

Крупные JSON платёжных шлюзов набирают anomaly score и блокируются. Выносим их endpoint в отдельный location с allowlist по IP.

✕ Audit-лог съедает диск

SecAuditLogParts ABIJDEFHZ пишет тела запросов; при активном сканировании — гигабайты в неделю. Обязательны logrotate и алерт на место.

✕ CRS застыл на версии установки

Без обновлений набор правил слепнет к новым техникам атак. Ежемесячный апдейт по тегам плюс обкатка новых правил в detection-режиме.

✕ Сразу Paranoia Level 3–4

PL3+ даёт лавину ложных срабатываний и недели тюнинга. Для корпоративных сайтов и магазинов начинаем с PL1–PL2.

✕ WAF как индульгенция коду

WAF — дополнительный слой, а не замена prepared statements и валидации ввода. Уязвимости в коде закрываем параллельно.

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- libmodsecurity 3.0.15 + CRS 4 LTS на Nginx, старт в DetectionOnly
- Перевод в SecRuleEngine On после недели тюнинга, PL1, пороги 5/4
- fail2ban по audit-логу + logrotate + контроль места на диске

НОРМАЛЬНО

- PL2 (tx.blocking_paranoia_level=2) + early_blocking для ЛК и форм
- Адресные исключения в файлах BEFORE/AFTER-CRS, не удаление правил
- Telegram-алерты на критичные правила, еженедельный разбор лога
- Ежемесячное обновление CRS с чтением changelog релиза

ХОРОШО

- detection_paranoia_level=3 при блокировке на PL2 — видим атаки заранее
- Audit-лог в JSON → SIEM/ELK, корреляция с банами fail2ban
- Конфиги WAF в git, раскатка Ansible, nginx -t в CI перед деплоем
- Регулярный пентест форм и API + ревизия накопленных исключений

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] WAF реально блокирует (SecRuleEngine On), а не годами живёт в DetectionOnly?
- [ ] Движок обновлён до libmodsecurity 3.0.15 — закрыты DoS-уязвимости 2026 года?
- [ ] CRS обновлялся за последний месяц и кто-то читает changelog релизов?
- [ ] Ложные срабатывания сняты точечно по параметру, а не отключением правил?
- [ ] Вебхуки платёжных систем выделены в отдельный location и не блокируются?
- [ ] Настроена ротация audit-лога и алерт на заполнение диска?
- [ ] Повторные атаки с одного IP автоматически банятся fail2ban?
- [ ] Есть процедура: кто и как разбирает срабатывания и жалобы «не работает форма»?
- [ ] При апгрейде Nginx предусмотрена пересборка модуля коннектора?

Если хотя бы на два вопроса ответ «нет» или «не знаю» — тема требует внимания.

ЧТО МЫ ДЕЛАЕМ

Как поможет ITFresh

ITFresh — ИТ-аутсорсинг для юридических лиц до 50 рабочих мест в Москве и области. 15+ лет практики, собственная инфраструктура в дата-центре МТС (8 серверов Dell Xeon Platinum).

- Аудит сайта: карта форм и API, разбор логов Nginx на следы атак
- Развёртывание ModSecurity v3 + OWASP CRS 4 под ключ на вашем сервере
- Тюнинг без потерь заказов: недельный цикл DetectionOnly → блокировка
- Связка fail2ban + Telegram-алерты + ротация логов
- Абонентское сопровождение: обновление CRS, разбор срабатываний, отчёт раз в неделю

15+
лет в ИТ-поддержке

50
рабочих мест — наш профиль

МТС
дата-центр, Москва

КОНТАКТЫ

Обсудить вашу задачу

Сайт	**itfresh.ru**
Телефон	**+7 903 729-62-41**
Telegram	**@ITfresh_Boss**

Бесплатно посмотрим вашу инфраструктуру по этому чек-листу и скажем, где тонко — без обязательств.

itfresh.ru

ДОКУМЕНТАЦИЯ И НАШ СТЕК

Техническая база

01 modsecurity.conf-recommended — базовые директивы движка (github.com/owasp-modsecurity — 3.0.15)

02 crs-setup.conf.example: правила 900000–900990 (github.com/coreruleset — 4.28.0)

03 CRS Docs: Anomaly Scoring и Paranoia Levels (coreruleset.org — CRS 4)

04 ModSecurity-nginx Connector (динамический модуль) (github.com/owasp-modsecurity — 1.0.4)

05 ModSecurity Reference Manual v3 (SecRule, ctl:) (github.com/owasp-modsecurity — v3)

06 Наш шаблон: WAF-тюнинг, fail2ban jail, Telegram-алерты (itfresh.ru — 2026)

Основано на официальной документации продуктов и нашей практике внедрения.