

**Программа для ЭВМ
«UBIC CORE»**

Руководство пользователя

Оглавление

Введение	3
1. УПРАВЛЕНИЕ ЯДРОМ PFETCHER2.....	3
2. СЛУЖЕБНЫЙ RPC-ИНТЕРФЕЙС И КЛИЕНТ OSAT	3
3. УПРАВЛЕНИЕ ИДЕНТИФИКАЦИЕЙ АБОНЕНТОВ	4
4. УПРАВЛЕНИЕ URL-КАТАЛОГОМ	5
5. ЧТЕНИЕ И ПЕРЕНАПРАВЛЕНИЕ DNS (DNS-TRACER)	5
6. УЧЕТ АБОНЕНТОВ ПО RADIUS (RADIUS-TRACER).....	5
7. ПРОСМОТР ЭКСПЛУАТАЦИОННОЙ СТАТИСТИКИ (PSTATS).....	6
8. ФОРМАТ ВЫГРУЗКИ РЕЗУЛЬТАТОВ	6
Практический пример №1	7
Шаг 1. Идентификация абонентов	7
Шаг 2. Проверка абонентской базы	7
Шаг 3. Запуск ядра обработки и фильтрации трафика	7

Введение

Назначение документа

Руководство пользователя предназначено для следующих категорий специалистов:

- системных администраторов и инженеров, осуществляющих запуск, настройку и эксплуатацию компонентов программы для ЭВМ «UBIC CORE» – в части управления ядром DPI, каталогами, идентификацией абонентов, перенаправлением DNS и учетом по RADIUS;
- операторов, осуществляющих управление каталогами и правилами через служебный RPC-интерфейс, а также контроль эксплуатационной статистики;
- специалистов, осуществляющих проверку программы для ЭВМ «UBIC CORE» в рамках включения в «Единый реестр российских программ для электронных вычислительных машин и баз данных» (пример представлен в разделе «Практический пример №1»).

Основная функция ПО

Программа для ЭВМ «UBIC CORE» – это система анализа сетевых пакетов и умной маршрутизации, функционирующая на уровне сети провайдера или оператора связи.

Программа обрабатывает сетевые пакеты в режиме, близком к режиму реального времени, идентифицирует абонента, классифицирует запрашиваемый контент, при необходимости осуществляет перенаправление (маршрутизацию) пакетов, а также ведет учет и выгрузку статистики во внешний аналитический конвейер. Управление осуществляется через флаги командной строки компонентов, JSON-конфигурацию ядра и служебный RPC-интерфейс.

1. УПРАВЛЕНИЕ ЯДРОМ PFETCHER2

Ядро pfetcher2 запускается с указанием режима чтения, цели и конфигурационного файла. Основные флаги командной строки:

Флаг	Описание
--online / --offline / --inmemory	Режим чтения: сетевой интерфейс / рсар-дамп / из памяти
--config	Путь к JSON-конфигурации ядра
--target	Сетевой интерфейс или путь к дампу
--filter	Фильтр рсар для чтения пакетов
--debug_ip	Отладочное логирование по IP-адресу
--verbose	Расширенное логирование

Пример запуска ядра в режиме реального времени:

```
pfetcher2 --online --target ethX --config pfletcher2.json
```

2. СЛУЖЕБНЫЙ RPC-ИНТЕРФЕЙС И КЛИЕНТ 0SAT

Управление каталогами и правилами в реальном времени выполняется через служебный RPC-интерфейс («cowboy») по протоколу ZeroMQ (REQ/REP). Команды отправляются

CLI-клиентом `0cat` в формате JSON (читается из стандартного ввода); ответ печатается в стандартный вывод. Флаг `--h` конвертирует Unix-таймстемпы в дату и время.

Флаги клиента `0cat`:

Флаг	Описание
<code>--transport</code>	Адрес RPC (по умолчанию <code>ipc:///tmp/cowboy.ipc</code>)
<code>--timeout</code>	Таймаут ожидания ответа, мс
<code>--h</code>	Преобразование таймстемпов в дату/время

Команды RPC-интерфейса ядра `pfetcher2`:

Команда	Аргументы	Действие
<code>catalog_update</code>	<code>catalog, path</code>	Перезагрузить каталог/ДАКТ из файла
<code>catalog_find</code>	<code>catalog, url</code>	Найти URL в каталоге (для <code>rosenthal</code> – ДАКТ-паттерн)
<code>catalog_version</code>	<code>catalog</code>	Версия и таймстемпы сборки/загрузки каталога
<code>family_search_update</code>	<code>cpoint, key, value</code>	Обновить настройку управления интернет-потреблением
<code>pfetcher_version</code>	–	Версия модуля

Значение аргумента `catalog`: `main` / `local` / `personal` / `extension` (каталоги `linnaeus`) либо `rosenthal` (ДАКТ). Пример команды запроса версии основного каталога:

```
{ "name": "catalog_version", "args": { "catalog": "main" } }
```

3. УПРАВЛЕНИЕ ИДЕНТИФИКАЦИЕЙ АБОНЕНТОВ

Утилита `identification` создает и обслуживает сегмент разделяемой памяти `__SHID__`. Основные флаги:

Флаг	Описание
<code>--create</code> / <code>--destroy</code>	Создать/удалить сегмент
<code>--load</code>	Загрузить статические карты из MySQL
<code>--add</code> / <code>--identify</code> / <code>--show</code>	Добавить запись/проверить идентификацию / показать содержимое
<code>--ip</code> , <code>--vlan</code> , <code>--id</code> , <code>--hid</code> , <code>--bm</code> , <code>--wm</code> , <code>--fts</code>	Параметры записи о пользователе
<code>--static</code> / <code>--dynamic</code>	Работа со статической/динамической картой
<code>--segment_name</code> / <code>--segment_size</code>	Имя (<code>__SHID__</code>) и размер сегмента (по умолчанию порядка 8 ГБ)
<code>--db_host</code> / <code>--db_port</code> / <code>--db_user</code> / <code>--db_pswd</code> / <code>--db_name</code>	Реквизиты подключения к MySQL

4. УПРАВЛЕНИЕ URL-КАТАЛОГОМ

Утилита catalog2 строит дерево URL-каталога в разделяемой памяти из таблицы MySQL. Основные флаги:

Флаг	Описание
--create / --destroy / -load	Создать/удалить/загрузить каталог
--show	Показать содержимое каталога
--check <url>	Проверить категоризацию URL
--limit	Ограничение выборки
--catalog	Таблица-источник (по умолчанию dim_site_fast)
--segment_size	Размер сегмента (по умолчанию 50 МБ)
--db_*	Реквизиты подключения к MySQL

Ядро pfletcher2 при работе с файловыми каталогами linnaeus/rosenthal (формат .dic) сразу перезагружает их командой catalog_update служебного RPC-интерфейса. Это независимый от catalog2 механизм представления каталога.

5. ЧТЕНИЕ И ПЕРЕНАПРАВЛЕНИЕ DNS (DNS-TRACER)

Компонент dns-tracer перенаправляет DNS-ответы абонентам с установленными параметрами интернет-потребления по CSV-каталогу. Основные флаги:

Флаг	Описание
--offline / --verbose	Обработка дампа/расширенное логирование
--in / --out	Входной и выходной интерфейсы
--dnscsv	Путь к CSV-каталогу DNS
--touchfile	Файл сигнала живости для мониторинга
--transport	Адрес RPC (по умолчанию tcp://*:6666)

CSV-каталог можно перезагрузить без остановки компонента RPC-командой dnscsv_update. Вспомогательный скрипт ipv6_resolver.sh предвычисляет IPv6-адреса.

6. УЧЕТ АБОНЕНТОВ ПО RADIUS (RADIUS-TRACER)

Компонент radius-tracer наполняет таблицу __SHID__ динамически по данным RADIUS-аккаунтинга. Основные флаги:

Флаг	Описание
--offline / --verbose	Обработка дампа/расширенное логирование
--target	Сетевой интерфейс
--id_segment_name	Имя сегмента идентификации (по умолчанию __SHID__)
--debug_logins	Выборочное логирование по MD5 логина
--transport	Адрес RPC (по умолчанию tcp://*:6667)

7. ПРОСМОТР ЭКСПЛУАТАЦИОННОЙ СТАТИСТИКИ (PSTATS)

Утилита `pstats` предназначена для управления сегментом разделяемой памяти `__PSTATS__` и просмотра хранимых в нем счетчиков. Основные флаги:

Флаг	Описание
<code>--create / --destroy / --reset</code>	Создать/удалить/обнулить сегмент
<code>--show</code>	Показать счетчики (вывод в формате JSON)
<code>--segment_name</code>	Имя сегмента (по умолчанию <code>__PSTATS__</code>)

В числе отображаемых счетчиков: количество прочитанных и обработанных пакетов, HTTP-запросов, перенаправленных поисковых запросов, выгруженных фактов, добавленных и удаленных сессий, успешных и неуспешных попыток реассемблинга. Также поддерживается расчет скорости обработки пакетов в Мбит/с.

8. ФОРМАТ ВЫГРУЗКИ РЕЗУЛЬТАТОВ

Результаты классификации сериализуются в формате Protocol Buffers (сообщение Click) и передаются во внешнюю систему аналитической обработки данных с использованием протокола ZeroMQ (режим PUSH). Сообщение содержит версию формата, метку времени и тип записи (запись распознанного абонента или запись нераспознанного абонента). Для распознанного абонента передаются идентификатор пользователя, URL-адрес, значение поля Referer, значение cookie, идентификатор сайта, присвоенные категории, результат проверки по спискам запрещенных ресурсов, IP-адрес и значение User-Agent. Записи, содержащие служебные (не подлежащие выгрузке) категории, при выгрузке отбрасываются.

Практический пример №1

Ниже приведен типовой сценарий развертывания и эксплуатации Программы.

Шаг 1. Идентификация абонентов

Обеспечить прием AAA-трафика (RADIUS) на стенд, запустить RADIUS-трейсер с логированием и подать AAA-пакет тестового абонента; убедиться в появлении записи «IP – ID абонента» в кэше командой `updater --identify --ip=<id>`.

Для отладочной имитации идентификационного сигнала без RADIUS-трафика используется `updater --add --id=<id> --hid=<hid> --bm=<black> --wm=<white> --fts=<fts> --dynamic` (либо `--static`).

Шаг 2. Проверка абонентской базы

Выгрузить базу абонентов и убедиться в ее наполнении: `updater --show > /tmp/subscribers.dump && wc -l /tmp/subscribers.dump`.

Шаг 3. Запуск ядра обработки и фильтрации трафика

Подать на стенд трафик (пассивная копия либо активный) и зафиксировать состояние счетчиков командой `pstats --show` до и после выдержанного интервала.

Критерий работоспособности – прирост счетчика принятых/обработанных пакетов при отсутствии роста счетчика `drops/errors`.