Настройка rsyslog для передачи сообщений внешнему сервису посредством unix-сокета

1. Создайте скрипт, который будет открывать на прослушивание сокет и выполнять действия при поступлении сообщений (если копируете код, то помните, что имеет значение количество пробелов в начале строки):

```
# cat /usr/local/sbin/mylogparser.py
#!/usr/bin/python3

import socket
import os
import subprocess
import re

# Местоположение сокета
SOCKET_PATH = '/var/run/mysyslog.socket'

# Удаляем сокет, если он существует
if os.path.exists(SOCKET_PATH):
    os.remove(SOCKET_PATH)

# Создаем сокет
server_socket = socket.socket(socket.AF_UNIX, socket.SOCK_DGRAM)

# Подключаемся к созданному сокету
server_socket.bind(SOCKET_PATH)
os.chmod(SOCKET_PATH, 0o600)

# Стоку ниже и далее с функцией print можно раскомметировать для отладки
#print(f"Unix DGRAM server listening on {SOCKET_PATH}...")

# Регулярные выражения для обработки сообщений
pattern = r"^.*sudo:\s*(\S+)\s*: user NOT in sudoers.*$"
replace = r"\1"

try:
    while True:
        # Получаем данные
        data, addr = server_socket.recvfrom(1024)
        if not data:
            break
        #print(f"Received from {addr}: {data.decode()}")
        # Вычисляем с помощью регулярного выражения имя пользователя
        buser = re.sub(pattern, replace, data.decode())
        #print(f"Bad user is {buser}")
        # Запускаем команду для блокировки пользователя
        result = subprocess.run(["/bin/passwd", "-l", f"{buser}"],
capture_output=True, text=True)
        #print(f"Result of locking {buser}: {result}")
        # Запускаем команду для завершения всех сеансов пользователя
        result = subprocess.run(["/bin/loginctl", "terminate-user", f"{buser}"],
capture_output=True, text=True)
        #print(f"Result of processes terminating {buser}: {result}")

except KeyboardInterrupt:
    print("Server stopped.")
finally:
    server_socket.close()
    if os.path.exists(SOCKET_PATH):
        os.remove(SOCKET_PATH)
```

2. Создайте службу, которая открывает сокет:

```
# cat /etc/systemd/system/mylogparser.service
[Unit]
Description=Run socket parser for messages
Before=rsyslog.service

[Service]
Type=exec
ExecStart=/usr/local/sbin/mylogparser.py
Restart=always

[Install]
WantedBy=multi-user.target
```

3. Обновите конфигурацию systemd и включите сервис:

```
# systemctl daemon-reload

# systemctl enable --now mylogparser.service
Created symlink
'/etc/systemd/system/multi-user.target.wants/mylogparser.service' →
'/etc/systemd/system/mylogparser.service'.

# systemctl status mylogparser.service
● mylogparser.service - Run parser for message recieved from socket
     Loaded: loaded (/etc/systemd/system/mylogparser.service; enabled; preset: >
     Active: active (running) since Wed 2026-02-18 19:28:44 +05; 5s ago
```

4. Настройте rsyslog на передачу сообщений внешней службе:

```
# cat /etc/rsyslog.d/myparser.conf

# Строки ниже показывают альтернативные способы передачи сообщений
# Это через именованный канал (pipe)
#:msg, contains, "user NOT in sudoers"  |/var/run/mysyslog.pipe
# Это через UDP порт
#:msg, contains, "user NOT in sudoers"  @127.0.0.1:33333

$ModLoad omuxsock

# Сначала задаем параметры для следующего за ними действия
$OMUxSockSocket /var/run/mysyslog.socket

# Затем само условие и вызов модуля через двоеточия
:msg, contains, "user NOT in sudoers" :omuxsock:
```

5. Перезапустите службу rsyslog и проверьте корректность работы сервиса:

```
# systemctl restart rsyslog.service

# systemctl status rsyslog.service
● rsyslog.service - System Logging Service
     Loaded: loaded (/usr/lib/systemd/system/rsyslog.service; enabled; preset: >
     Active: active (running) since Wed 2026-02-18 19:36:14 +05; 9s ago

# ssh user@::1
user@::1's password:
<...>
user@srv1:~$ sudo -i
[sudo] пароль для user:
user отсутствует в файле sudoers.
Администратор был оповещён об этом событии.
user@srv1:~$ Connection to ::1 closed by remote host.
Connection to ::1 closed.
```

Применение: Для переключения в режим UDP блок:

```
# Местоположение сокета
SOCKET_PATH = '/var/run/mysyslog.socket'

# Удаляем сокет, если он существует
if os.path.exists(SOCKET_PATH):
    os.remove(SOCKET_PATH)

# Создаем сокет
server_socket = socket.socket(socket.AF_UNIX, socket.SOCK_DGRAM)

# Подключаемся к созданному сокету
server_socket.bind(SOCKET_PATH)
```

Надо заменить на:

```
# Местоположение сокета
SOCKET_HOST = '127.0.0.1'
SOCKET_PORT = 33333

# Создаем сокет
server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

# Подключаемся к созданному сокету
server_socket.bind((SOCKET_HOST,SOCKET_PORT))
```

Для использования именованных каналов нужно будет создать этот канал и прочитать из него данные:

```
import os, time

pipe_path = "/var/run/mysyslog.pipe" # Specify a path

if not os.path.exists(pipe_path):
    os.mkfifo(pipe_path) # Create the named pipe

<...>

# Читаем данные из канала:
with open(pipe_path, "r") as pipe_reader:
    data = pipe_reader.read()
<...>
# Clean up the named pipe file (usually done after communication is complete)
os.remove(pipe_path)
```