Мониторинг серверного оборудования через утилиту IpmiTools

Назначение

Передача данных мониторинга он в внешних систем в протокол событий СПО ИНДИГИРКА посредством формирования и отправки JSON-файлов. В данной рекомендации в качестве источника информации используется серверная утилита Ipmitool

Используемое оборудование

Название	Дополнительная информация
ИД-ССР-РВ	Серверная станция ИСБ ИНДИГИРКА с поддержкой утилиты Ipmitools
СПО ИНДИГИРКА	ЛИЦ-ИД-СПО-ПА-СРВ – Сервер связи с оборудованием, ЛИЦ-ИД-СПО-ПА-СБД – сервер протокола событий.

Описание

В процессе работы системы сбора и передачи информации принимают участие несколько скриптов bash. Ниже приведены описания скриптов и порядок их работы. Тексты скриптов приведены полностью в данном документе для их самостоятельно создания и размещения. В данном примере описана передача состояний блоков питания от сервера ИД-ССР-РВ.

1) Создать файл скрипта.

Создать файл скрипта **ipmi_tcp** и расположить его в каталоге с установленной СПО ИНДИГИРКА **/opt/Indigirkalnstall/bin/Config/**.

Далее приведен текст скрипта, который необходимо скопировать в созданный файл. Синим цветом в документе выделены комментарии по работе скрипта, красным — поля в скрипте для самостоятельного заполнения на основе исходных данных.:

```
#!/bin/bash
# Настройка IP_IPMI это IP-адрес, по которому скрипт получает данные от внешней утилиты
#мониторинга, в данном случае от ipmitool:
IP_IPMI="192.168.0.1"
# Скрипт запрашивает у утилиты ipmitool информацию о состоянии блоков питания:
POWER_STATUS=$(sudo ipmitool sdr type "Power Supply" | awk -F'|' -v ip_ipmi="$IP_IPMI" '
BEGIN {
   print "{"
   print "  \"Signature\": \"ID_HARDWARE_MONITORING\","
   # Заполняются названия состояний, которые будут переданы в протокол событий СПО
#ИНДИГИРКА. ST_FALURE_LIST – список тревожных состояний (в протоколе событий они будут
#выводится в строке оранжевого цвета, что соответствует событию неисправности),
#ST_NORME_LIST – список нормальных состояний, которые не выделяются цветом в протоколе
#событий.
   ST_FAILURE_LIST = "\"Неисправность блока питания1\",\"Неисправность блока питания2\""
   ST_NORME_LIST ="\"Блоки питания в исправном состоянии\""
   # Соответствия вышеуказанных состояний неисправности с выводом от внешней утилиты сбора
#данных:
   error_map["PS1"]["Power Supply AC lost"] = "Неисправность блока питания1"
   error_map["PS2"]["Power Supply AC lost"] = "Неисправность блока питания2"
   # Инициализация полученных данных от внешней утилиты:
   error_messages = ""
```

```
    error_count = 0
}
{
    gsub(/^[ \t]+|[ \t]+$/, "", $1);
    gsub(/^[ \t]+|[ \t]+$/, "", $5);
    # Определение типа источника питания, на основе информации, присланной от внешней
#утилиты мониторинга:
    ps_type = "PS1"
    if (match($1, /PS2|2/)) ps_type = "PS2"
    # Проверка на ошибки:
    for (error in error_map[ps_type]) {
        if (index($5, error) > 0) {
            if (error_count++ > 0) error_messages = error_messages "; "
            error_messages = error_messages error_map[ps_type][error]
        }
    }
}
END{

# Формирование состояния, для нормы, когда неисправности отсутствуют. Название состояния
#должно совпадать с названием, добавленным в ST_NORME_LIST:
    state = "Блоки питания в исправном состоянии"
    if (error_count > 0) {
    state = error_messages
}
    # Вывод результата, в строке HW_OBJECT задать название оборудования, от которого пришло
#сообщение, именно так оно будет показано в протоколе событий в столбце «Объект»:
    print "  \"HW_OBJECT\": \"CCR SRV-1 источник питания материнской платы\","
    print "  \"ST_FAILURE\": [" ST_FAILURE_LIST "],"
    print "  \"ST_NORME\":[" ST_NORME_LIST "],"
    print "  \"STATE\": \"" state "\","
    print "  \"PARAM\": \""ip_ipmi "\""
    print "}\xf8\xef"
}')
# Формирование полного вывода:
FULL_OUTPUT="\n$POWER_STATUS"

# Вывод результата в терминал для проверки работы скрипта:
echo -e "Сформированные данные:"
echo -e "$POWER_STATUS"
# Отправка сообщения, для этого необходимо задать ip — адрес и порт сервера связи
#СПО ИНДИГИРКА):
TCP_SERVER="192.168.0.2"
TCP_PORT="8081"
if ! exec 3<>/dev/tcp/$TCP_SERVER/$TCP_PORT; then
    echo "Ошибка подключения к серверу $TCP_SERVER:$TCP_PORT"
    exit 1
fi
echo "$FULL_OUTPUT" >&3
read -t 5 -r response <&3
[ -n "$response" ] && echo "Ответ сервера: $response"
exec 3<&-
exec 3>&-
echo "Данные отправлены на сервер $TCP_SERVER:$TCP_PORT"
```

2) Создать файл для запуска скрипта.

Для удобства обращения создать скрипт **start_ipmi** в общей папке **/opt/**. Далее приведен текст скрипта, который необходимо скопировать в созданный файл. Синим цветом в документе выделены комментарии по работе скрипта, красным — поля в скрипте для самостоятельного заполнения на основе исходных данных.:

```
#!/bin/bash
while true; do
# Указать путь расположения основного скрипта ipmi_tcp:
/opt/IndigirkaInstall/bin/Config/ipmi_tcp
# Указать периодичность запуска скрипта, в секундах, от этого зависит как часто скрипты будут
#запрашивать и проверять состояние оборудования:
        sleep 60
done
```

3) Настройка службы CRON для автоматического запуска скриптов.

Открыть терминал сочетанием клавиш ALT+T и ввести команду:
sudo crontab -e

В открывшемся файле настройки CRON (crontab) добавить задание на запуск скрипта при перезагрузки рабочей станции:
@reboot /opt/start_ipmi

4) Для подтверждения изменений перезапустить рабочую станцию.