

WCS Core

- [Настройки логирования](#)
 - [Настройки логирования flashphoner.properties](#)
 - [Настройки логирования log4j.properties](#)
 - [Описание настроек](#)
 - [Горячая замена настроек логирования](#)
 - [Трассировка Websocket сообщений](#)
- [Клиентские логи](#)
 - [Включение, отключение и управление уровнем логирования](#)
 - [Управление уровнем логирования "на лету"](#)
 - [REST-методы и статусы ответа](#)
 - [Параметры](#)
 - [Включение отладочных логов для всех клиентских сессий](#)
 - [Использование самописца \(flight recorder\)](#)
 - [Структура и содержимое клиентских логов](#)
 - [Лог flashphoner.log](#)
 - [Лог client-report](#)
 - [Дампы медиатрафика](#)
 - [Логflight_recorder.log](#)
- [Серверные логи](#)
- [CDR-логи](#)
- [MDR-логи](#)
- [SDR-логи](#)
- [CONNDR-логи](#)
- [GC-логи](#)
- [Лог статистики медиасессий](#)
- [УязвимостьCVE-2021-44228](#)
 - [Пояснения: почему WCS не подвержен уязвимости](#)

Настройки логирования

За логирование WCS Core отвечает файл настроек[log4j.properties](#) и ряд настроек файла[flashphoner.properties](#):

Настройки логирования flashphoner.properties

Настройка	Значение по умолчанию
client_log_level	INFO
client_dump_level	0
enable_extended_logging	true

Логи пишутся в каталог `/usr/local/FlashphonerWebCallServer/logs`

- `client_logs` - логи, которые пишутся на стороне сервера и относятся к сессии клиента с WCS-сервером (Клиентские логи).
- `server_logs` - общие логи, которые пишутся на стороне сервера.

Настройки логирования log4j.properties

Это стандартный конфиг формата [log4j](#).

```
mc - root@localhost:usr/local/FlashphonerWebCallServer-3.0.1011/conf
log4j.properties [---] 0 L:[ 1+ 0 1/ 40] * (0 /2269b)= 1 108 0x6C
log4j.rootLogger=info, stdout, fAppender

log4j.logger.incoming.Publication=info, incoming_publication
log4j.logger.outgoing.Publication=info, outgoing_publication
log4j.logger.pushLogs.FlashphonerHandler=info, clientLog
log4j.additivity.incoming.Publication=false
log4j.additivity.outgoing.Publication=false
log4j.additivity.pushLogs.FlashphonerHandler=false

log4j.logger.sipMessages=DEBUG
#log4j.logger.send.SentMessageControl=DEBUG
#log4j.logger.send.BurstAvoidanceController=DEBUG
#log4j.logger.send.FlowWriter=DEBUG
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%d{HH:mm:ss,SSS} %-5p %20.20c{1} - %m%n

log4j.appender.fAppender=org.apache.log4j.DailyRollingFileAppender
log4j.appender.fAppender.DatePattern='.'yyyy-MM-dd-HH
log4j.appender.fAppender.layout=org.apache.log4j.PatternLayout
log4j.appender.fAppender.layout.ConversionPattern=%d{HH:mm:ss,SSS} %-5p %20.20c{1} - %t %m%n
log4j.appender.fAppender.File=${com.flashphoner.fms.AppHome}/logs/server_logs/flashphoner.log

log4j.appender.incoming_publication=org.apache.log4j.DailyRollingFileAppender
log4j.appender.incoming_publication.DatePattern='.'yyyy-MM-dd-HH
log4j.appender.incoming_publication.layout=org.apache.log4j.PatternLayout
log4j.appender.incoming_publication.layout.ConversionPattern=%m%n
log4j.appender.incoming_publication.File=${com.flashphoner.fms.AppHome}/logs/stats/flashphoner-incoming-publications.log

log4j.appender.outgoing_publication=org.apache.log4j.DailyRollingFileAppender
log4j.appender.outgoing_publication.DatePattern='.'yyyy-MM-dd-HH
log4j.appender.outgoing_publication.layout=org.apache.log4j.PatternLayout
log4j.appender.outgoing_publication.layout.ConversionPattern=%m%n
log4j.appender.outgoing_publication.File=${com.flashphoner.fms.AppHome}/logs/stats/flashphoner-outgoing-publications.log

log4j.appender.clientLog=org.apache.log4j.DailyRollingFileAppender
log4j.appender.clientLog.DatePattern='.'yyyy-MM-dd-HH
log4j.appender.clientLog.layout=org.apache.log4j.PatternLayout
log4j.appender.clientLog.layout.ConversionPattern=%d{HH:mm:ss,SSS} %m%n
log4j.appender.clientLog.File=${com.flashphoner.fms.AppHome}/logs/client_logs/flashphoner-client-logs.log

1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn 10Quit
```

Описание настроек

Атрибут	Значение	Описание
log4j.rootLogger	info, stdout, fAppender	Корневой логгер. info - Уровень логгирования INFO. Доступны другие более подробные уровни, например DEBUG и TRACE и менее подробные, например ERROR. stdout, fAppender - определяют как и куда будут выводиться логи.
log4j.logger.incoming.Publication	info, incoming_publication	Логгер статистики RTMFP-SIP звонков для входящего с SIP сервера трафика. info - уровень логгирования incoming_publication - определяют как и куда будут выводиться логи.
log4j.logger.outgoing.Publication	info, outgoing_publication	Логгер статистики RTMFP-SIP звонков для исходящего на SIP сервер трафика. info - уровень логгирования outgoing_publication - определяют как и куда будут выводиться логи.
log4j.logger.pushLogs.FlashphonerHandler	Не используется	Не используется
log4j.additivity.incoming.Publication	false	Не дублировать данные логи в общий лог, а писать в отдельные
log4j.additivity.outgoing.Publication	false	Не дублировать данные логи в общий лог, а писать в отдельные
log4j.logger.sipMessages	debug	Выводить входящие и исходящие SIP сообщения в лог

log4j.logger.WSServerHandler	trace	Выводить исходящие Websocket сообщения в лог
log4j.logger.WSClient	debug	Выводить входящие Websocket сообщения в лог
log4j.appender.stdout	org.apache.log4j.ConsoleAppender	Вывод логов в stdout
log4j.appender.fAppender	org.apache.log4j.DailyRollingFileAppender	Вывод логов в fAppender
log4j.appender.incoming_publication	org.apache.log4j.DailyRollingFileAppender	Вывод статистики RTMFP в incoming_publication
log4j.appender.outgoing_publication	org.apache.log4j.DailyRollingFileAppender	Вывод статистики RTMFP в outgoing_publication
log4j.appender.clientLog	org.apache.log4j.DailyRollingFileAppender	Не используется

Горячая замена настроек логгирования

WCS автоматически подхватывает изменения, сделанные в файле log4j.properties. Это удобно для целей отладки и получения дополнительных логов без перезагрузки сервера. Например, в том случае если требуется включить более подробные логи или изменить формат вывода логов. Однако для большей надежности в production все же рекомендуется выполнить перезагрузку WCS-сервера.

Трассировка Websocket сообщений

В целях отладки или разработки собственного API, может быть включена трассировка всех Websocket сообщений, кроме транспортных. Для того, чтобы логировать все входящие/исходящие Websocket сообщения в файл websocket.log в каталоге /usr/local/FlashphonerWebCallServer/logs/server_logs, необходимо добавить в файл log4j.properties следующие строки:

```
log4j.logger.WSServerHandler=trace, wsAppender
log4j.logger.WSClient=debug, wsAppender
log4j.appender.wsAppender=org.apache.log4j.DailyRollingFileAppender
log4j.appender.wsAppender.DatePattern='.'yyyy-MM-dd-HH
log4j.appender.wsAppender.layout=org.apache.log4j.PatternLayout
log4j.appender.wsAppender.layout.ConversionPattern=%d{HH:mm:ss,SSS} %-5p %20.20c{1} - %t %m%n
log4j.appender.wsAppender.File=${com.flashphoner.fms.AppHome}/logs/server_logs/websocket.log
```

Клиентские логи

Включение, отключение и управление уровнем логгирования

Клиентские логи - это логи на сервере, которые относятся к сессии web-клиента. Клиентские логи в client_logs пишутся только тогда, когда включена настройка (по умолчанию)

```
enable_extended_logging=true
```

Для отключения клиентских логов необходимо установить в файле [flashphoner.properties](#)

```
enable_extended_logging=false
```

Управлять уровнем логгирования можно настройкой client_log_level, которая может принимать значения ERROR, INFO, DEBUG, TRACE. По умолчанию

```
client_log_level=INFO
```

Для периодической очистки клиентских логов рекомендуется использовать cron в сочетании с find. Например, задание для проверки на устаревшие логи каждые 24 часа и удаление логов старше 30 дней будет выглядеть следующим образом

```
0 0 * * * find /usr/local/FlashphonerWebCallServer/logs/client_logs/ -type d -mtime +30 | xargs rm -rf
```

Управление уровнем логирования "на лету"

Уровень логирования для определенной сессии можно менять на ходу, без перезапуска сервера. Для этого используются REST-запросы

REST-запрос должен быть HTTP/HTTPS POST запросом в таком виде:

- HTTP:http://test.flashphoner.com:8081/rest-api/logger/enable_client_log
- HTTPS:https://test.flashphoner.com:8444/rest-api/logger/enable_client_log

Здесь:

- test.flashphoner.com- адрес WCS-сервера
- 8081 - стандартный REST / HTTP порт WCS-сервера
- 8444 - стандартный HTTPS порт
- rest-api - обязательная часть URL
- /logger/enable_client_log- используемый REST-метод

REST-методы и статусы ответа

REST-метод	Пример тела REST-запроса	Пример тела REST-ответа	Статусы ответа	Описание
/logger/enable_client_log	<pre>{ "sessionId": "/127.0.0.1:57539/192.168.1.101:8443", "logLevel": "DEBUG" }</pre>		200 - Уровень изменен 404 - Сессия не найдена	Установить указанный уровень логирования в заданной сессии
/logger/disable_client_log	<pre>{ "sessionId": "/127.0.0.1:57539/192.168.1.101:8443" }</pre>		200 - Логирование отключено 404 - Сессия не найдена	Полностью отключить логирование в заданной сессии

Параметры

Имя параметра	Описание	Пример
sessionId	Идентификатор сессии	/127.0.0.1:57539/192.168.1.101:8443
logLevel	Устанавливаемый уровень логирования	DEBUG

Таким образом, при возникновении проблем с потоком, опубликованным на сервере (например, поток опубликован, но не воспроизводится), необходимо отправить серверу REST-запрос и переключить уровень логирования в DEBUG, а затем, когда проблема воспроизведена и данные собраны, переключить уровень логирования обратно в INFO. Возможно также полностью отключить логирование для определенной сессии.

Изменение уровня логирования при помощи REST-запросов влияет только на заданную сессию, но не на другие сессии на сервере, в том числе на сессии, создаваемые позднее.

Включение отладочных логов для всех клиентских сессий

В некоторых случаях для диагностирования проблемы необходимо включить логирование для всех вновь подключившихся клиентских сессий, чтобы зафиксировать в логах процесс установки соединения и начало публикации потока. Эта возможность, начиная со сборки [5.2.512](#), включается настройкой

```
client_log_force_debug=true
```

Для всех вновь подключившихся клиентов будут записаны отладочные логи в течение времени в секундах, заданного настройкой

```
client_log_force_debug_timeout=60
```

По умолчанию, логи будут записываться в течение 60 секунд с начала сессии.

Эти настройки могут быть изменены через [интерфейс командной строки](#) и применены без перезапуска сервера.

Использование самописца (flight recorder)

Самописец (flight recorder) позволяет циклически записывать для публикуемого потока последние несколько событий. Эта информация поможет диагностировать проблемы с публикацией потока, не включая полные отладочные логи клиента. Самописец включается настройкой в файле [flashphoner.properties](#)

```
enable_flight_recorder=true
```

Необходимо также указать категорию событий, которые будут записываться (определяется разработчиком)

```
flight_recorder_categories=WCS1438
```

События записываются для публикующего клиента в файл `flight_recorder.log` в случае, если диагностируется остановка публикации потока, или поток портится каким-либо образом.

Для того, чтобы протестировать работу самописца, необходимо установить параметр

```
enable_flight_recorder_test=true
```

без перезапуска сервера. Это запишет события для всех подключенных публикующих клиентов.



Параметр `enable_flight_recorder_test` не предназначен для промышленной эксплуатации

Структура и содержимое клиентских логов

Структура клиентских логов:

```
client_logs
---- 2018-05-16
----- 84gij60a6u3ni7docsr1di115b-15-06-59
----- flashphoner.log
----- client-84gij60a6u3ni7docsr1di115b-2018.05.16.15.07.26-1526458046646.report
----- MediaDump-85d65b00-639e-4a7e.31002-31004-31006-31008.pcap
```

Лог flashphoner.log

Клиентские логи `client_logs` пишутся по датам. Под каждую дату создается директория с именем в формате ГГГГ-ММ-ДД, например 2018-05-16. Когда web-клиент устанавливает соединение с сервером, внутри директории с датой создается каталог для этой сессии клиента, например 84gij60a6u3ni7docsr1di115b-15-06-59, где 84gij60a6u3ni7docsr1di115b - идентификатор сессии, 15 - час, 06 - минута, 59 - секунда. В директорию пишется `flashphoner.log`, который содержит только те события на сервере, которые непосредственно относятся к этой сессии клиента. Таким образом мы видим когда клиент соединился с сервером, и какие логи были записаны для сессии этого клиента.

Лог client-report

Дополнительный клиентский лог. Web-клиент имеет специальную функцию WCS JavaScript API `'pushLog'`. Эта функция отправляет на WCS-сервер логи, которые ведутся на стороне браузера. Все логи, полученные от web-клиента по `pushLog`, будут сохраняться на сервере. Когда web-клиент завершит сессию с WCS-сервером, полученные логи будут записаны в файл `client-84gij60a6u3ni7docsr1di115b-2018.05.16.15.07.26-1526458046646.report`, где 84gij60a6u3ni7docsr1di115b - идентификатор сессии, 2018 - год, 05 - месяц, 26 - день, 15 - час, 07 - минута, 26 - секунда, 1526458046646 - миллисекунды.

Дампы медиатрафика

Если в файле настроек [flashphoner.properties](#) указано ненулевое значение `client_dump_level`, для клиента дополнительно пишется дамп сессии:

- если `client_dump_level=1`, записывается только SIP трафик;
- если `client_dump_level=2`, записывается весь медиатрафик.

Трафик записывается при помощи `tcpdump`, если данная утилита установлена в системе.

Лог flight_recorder.log

В данный файл записываются последние зафиксированные события для публикуемого потока.

Серверные логи

WCS Core пишет общие логи сервера в logs/server_logs

```
server_logs
---- flashphoner.log
---- flashphoner.log.2018-05-17-16
```

В этих логах можно отследить запуск и стартовые настройки сервера:

```
tail -f flashphoner.log
```

Запуск сервера

```
17:35:21.682 INFO Config - main Patches NOT installed
17:35:21.683 INFO Config - main NODE_ID: Op8P1bTHDacuaVfAELoJgcOFiWDVY6NL@0.0.0.0
17:35:21.693 INFO SettingsLoader - main Flashphoner config has been validated success
17:35:21.693 INFO SettingsLoader - main Server properties have been loaded:
(media_port=32000, wss.port=8443, burst_avoidance_count=100, wss.cert.password=password, get_callee_url=/usr/local/FlashphonerWebCall
17:35:21.953 INFO SettingsLoader - main Override setting media_port_to: from 32000 to 32000
17:35:21.985 INFO SettingsLoader - main Override setting wss.port: from 8443 to 8443
17:35:21.985 INFO SettingsLoader - main Override setting wss.cert.password: from password to password
17:35:21.985 INFO SettingsLoader - main Override setting burst_avoidance_count: from null to 100
17:35:21.986 INFO SettingsLoader - main Override setting get_callee_url: from null to /usr/local/FlashphonerWebCallServer/conf/ca
17:35:21.986 WARN Settings - main Setting 'log_level' is not found. Please check setting.
17:35:21.986 INFO SettingsLoader - main Override setting flush_video_interval: from 80 to 0
17:35:21.986 INFO SettingsLoader - main Override setting audio_frames_per_packet: from -1 to 6
17:35:21.986 INFO SettingsLoader - main Override setting call_record_listener: from null to com.flashphoner.server.client.Default
17:35:21.986 WARN Settings - main Setting 'waiting_answer' is not found. Please check setting.
17:35:21.986 INFO SettingsLoader - main Override setting on_record_hook_script: from null to on_record_hook.sh
17:35:21.987 INFO SettingsLoader - main Override setting keep_alive_peer_interval: from 2000 to 2000
17:35:21.987 WARN Settings - main Setting 'enable_context_logs' is not found. Please check setting.
17:35:21.987 INFO SettingsLoader - main Override setting keep_alive_server_interval: from 5000 to 5000
17:35:21.987 INFO SettingsLoader - main Override setting ip_local: from 0.0.0.0 to 95.191.131.64
17:35:21.987 INFO SettingsLoader - main Override setting codecs_exclude_streaming: from null to flv,telephone-event
17:35:21.987 INFO SettingsLoader - main Override setting balance_header: from null to balance
17:35:21.987 INFO SettingsLoader - main Override setting domain: from null to
17:35:21.988 INFO SettingsLoader - main Override setting audio_reliable: from partial to partial
17:35:21.988 INFO SettingsLoader - main Override setting codecs_exclude_sip_rtmp: from null to opus,g729,g722,mppeg4-generic,vp8,m
17:35:21.988 INFO SettingsLoader - main Override setting user_agent: from Flashphoner/1.0 to Flashphoner/1.0
17:35:21.988 INFO SettingsLoader - main Override setting rtmp_transponder_stream_name_prefix: from null to rtmp_
17:35:21.988 INFO SettingsLoader - main Override setting video_reliable: from partial to partial
17:35:21.988 INFO SettingsLoader - main Override setting codecs: from null to opus,alaw,ulaw,g729,speex16,g722,mppeg4-generic,tele
```

Остановка сервера

```
17:34:37.209 INFO ShutdownHandler - Thread-15 Shutting down RTMP Connections
17:34:37.211 INFO ShutdownHandler - Thread-21 Shutting down Rtp sessions
17:34:37.210 INFO activeShutdownHandler - Thread-6 Shutting down native libs
17:34:37.214 INFO ShutdownHandler - Thread-18 Shutting down RTMP Connections
17:34:37.214 INFO Sessions - Thread-18 shutdown
17:34:37.214 INFO ShutdownHandler - Thread-18 RTMP connections closed
17:34:37.219 INFO ShutdownHandler - Thread-15 Rtmp connections closed
17:34:37.219 INFO ShutdownHandler - Thread-21 Rtp sessions closed
17:34:37.221 INFO ShutdownHandler - Thread-20 Shutting down WebSocket connections
17:34:37.222 INFO ShutdownHandler - Thread-20 WebSocket connections closed
17:34:37.222 INFO ShutdownHandler - Thread-19 Shutting down WebSocket connections
17:34:37.223 INFO ShutdownHandler - Thread-19 WebSocket connections closed
17:34:37.236 INFO activeShutdownHandler - Thread-6 Done
```

Информация о лицензии:

```
17:35:22.722 INFO SipUserAgentListener - main License details
Activation date: 2018.04.09
Expiration date: 2017.10.22
Name: *****
Company: Flashphoner
Product name: Web Call Server 5
Features: lwcs_rtmp2rtmp_broadcasting, wcs_sip_as_rtmp, rtc2sip_vp8, flash2sip_h264, flash2sip_h263, wcs_webrtc_screen_sharing, rtc_au
LicenseNumber: *****-*****-*****
LicenseType: Subscription
LicenseSc: -1
HardwareId: 25349A0AF0B4E6EEB9EA9168BEED41DE83E47A190FF571AF38D0157DAA7D3FB45559F70ACB8B7BB40D5B4B9FBB6B72494204DBFF495B798C28D6D4237E5C
Support: Monthly subscription basic support
```

Кроме того, в серверных логах отображается информация о вызовах REST-методов

```
08:01:06,649 INFO          RestClient - API-ASYNC-pool-8-thread-2 SEND REST OBJECT ==>
URL:http://localhost:8081/EchoApp/StreamStatusEvent
OBJECT:
{
  "nodeId" : "rR3YA7yKB11iIIID4XkYveTF8ePhezMU@0.0.0.0",
  "appKey" : "defaultApp",
  "sessionId" : "/5.44.168.45:58541/95.191.131.65:8443",
  "mediaSessionId" : "58488550-99dd-11e8-bf13-9b5947c0a0f5",
  "name" : "569a",
  "published" : true,
  "hasVideo" : true,
  "hasAudio" : true,
  "status" : "PUBLISHING",
  "audioCodec" : "opus",
  "videoCodec" : "H264",
  "info" : "Unknown",
  "record" : false,
  "width" : 0,
  "height" : 0,
  "bitrate" : 0,
  "minBitrate" : 0,
  "maxBitrate" : 0,
  "quality" : 0,
  "timeShift" : -1,
  "createDate" : 1533603665644,
  "mediaProvider" : "WebRTC",
  "history" : false,
  "origin" : "https://test.flashphoner.com:8888"
}
```

Таким образом, серверные логи предоставляют общую информацию о работе сервера.

CDR-логи

Call Detail Record - это журнал SIP-звонков.

CDR записи пишутся в лог файл находящийся в logs/cdr/cdr.log. Новый лог файл создается на основе суточного интервала. Записи формируются в CSV-файле, что позволяет их удобно обрабатывать.

Названия полей в файл не пишутся.

Формат записи:

```
src;dst,cid,start,answer,end,billsec,disposition
```

Пример записи :

```
3000;3001;f294f6116bf2cc4c725f20457ed76e5b@192.168.56.2;2014-11-21 15:01:37; 2014-11-21 15:01:41; 2014-11-21 15:02:45;64;ANSWERED
```

Поле	Описание
src	Инициатор звонка
dst	Принимающий звонок
cid	Идентификатор звонка
start	Начало звонка (дата и время).
answer	Дата и время ответа абонента или SIP стороны.
end	Дата и время завершения звонка

billsec	Время в секундах между 'answer' и 'end'.
disposition	Результат звонка:ANSWERED, NO_ANSWER, BUSY, FAILED.

MDR-логи

Message Detail Record - это журнал SIP-сообщений.

MDR записи пишутся в лог файл находящийся в logs/cdr/mdr.log. Новый лог файл создается на основе суточного интервала. Записи формируются в CSV-файле, что позволяет их удобно обрабатывать.

Названия полей в файл не пишутся.

Формат записи:

```
date,msgId,from,to,disposition
```

Пример записи :

```
Fri Dec 26 15:26:16 NOV 2014,null,A006,A005,RECEIVED
```

Поле	Описание
date	Дата и время сообщения
msgId	Идентификатор сообщения. Будет представлен только в message/cpim сообщениях, при isImdnRequired=true (см. документацию Web Call Server - Call Flow, где описаны параметры передаваемых сообщений в методsendMessage).
from	SIP from
to	SIP to
disposition	Результат сообщения:RECEIVED, SENT, FAILED. RECEIVED- сообщение получено. SENT- сообщение отправлено. FAILED- во время отправки сообщения произошла ошибка.

Вы также можете собирать любую необходимую статистику по сообщениям и их статусам, используя WCS REST API. См. документацию Web Call Server - Call Flow, где перечислены все методы и наборы данных, которые WCS отправляет через REST при обработке сообщений.

SDR-логи

Stream Detail Record - это журнал сессий публикации и воспроизведения потоков.

SDR записи пишутся в лог-файл sdr.log, находящийся в директории logs/cdr. Новый лог-файл создается на основе часового интервала. Записи сохраняются в CSV-формате, что упрощает их обработку. Названия полей в файл не пишутся.

Формат записи:

```
start;mediaProvider;name;mediaSessionId;duration;disposition;info;type;subscribers;
```

Пример записи:

```
2015-11-11 08:36:13;Flash;stream-Bob;5c2d75c0-7d87-421d-aa93-2732c48d8eaa;00:00:48;UNPUBLISHED;;PUBLISH;3;
```

Поле	Описание
start	Дата и время установки сессии
mediaProvider	Используемое медиа на WCS JavaScript API:WebRTC, Flash

name	Имя публикуемого / воспроизводимого потока
mediaSessionId	Идентификатор медиа-сессии
duration	Продолжительность сессии
disposition	Как сессия была завершена:UNPUBLISHED, STOPPED, FAILED UNPUBLISHED- публикация потока была остановлена STOPPED- воспроизведение потока было остановлено FAILED- некорректное завершение сессии
info	Если disposition==FAILED, содержит описание причины
type	PUBLISHв случае публикации потока SUBSCRIBEв случае воспроизведения потока
subscribers	Количество подписчиков в случае публикации потока;0в случае воспроизведения потока

CONNDR-логи

Connection Detail Record - это журнал WebSocket-сессий.

CONNDR записи пишутся в лог-файл `sdr.log`, находящийся в директории `logs/cdr`. Новый лог-файл создается на основе часового интервала. Записи сохраняются в CSV-формате, что упрощает их обработку. Названия полей в файл не пишутся.

Формат записи:

```
start;mediaSessionId;disposition;info;duration;
```

Пример записи:

```
2018-04-25 19:29:08;/5.44.168.45:52199/95.191.131.64:8443;DISCONNECTED;Normal disconnect;17;
```

Поле	Описание
start	Дата и время установки сессии
mediaSessionId	Идентификатор медиа-сессии
disposition	Как сессия была завершена:DISCONNECTED, FAILED DISCONNECTED- завершение сессии по инициативе клиента FAILED- некорректное завершение сессии
info	Содержит описание завершения сессии
duration	Продолжительность сессии

GC-логи

По умолчанию логи сборщика мусора находятся в директории `/usr/local/FlashphonerWebCallServer/logs`.

```
logs
---- gc-core-2018-12-18_20-02.log
---- gc-core-2018-12-18_19-56.log
```

Расположение и префикс имени лога можно изменить в файле настроек [wcs-core.properties](#).

Для осуществления ротации логов средствами JVM в [wcs-core.properties](#) могут быть добавлены следующие настройки:

```
-XX:+UseGCLogFileRotation
-XX:NumberOfGCLogFiles=10
-XX:GCLogFileSize=2M
```

Тогда имена файлов будут такими

```
logs
---- gc-core.log2018-12-14_18-57.log.0
---- gc-core.log2018-12-14_18-57.log.1
---- gc-core.log2018-12-14_18-57.log.2.current
```

Суффикс 'current' обозначает файл, в который ведется запись.

Чтобы убрать время создания из имени файла, нужно убрать проставление даты из переменной GC_SUFFIX в bin/setenv.sh:

```
GC_SUFFIX=".log"
```

Тогда имена файлов будут такими

```
logs
---- gc-core.log.0
---- gc-core.log.1
---- gc-core.log.2.current
```

Лог статистики медиасессий

В сборке [5.2.1883](#) добавлено отображение [текущей статистики по медиасессиям](#). Для того, чтобы эта статистика сохранялась в файл, была добавлена возможность ее логирования по завершении медиасессии. Кроме того, в сборке [5.2.1975](#) добавлена настройка, которая включает эту возможность

```
media_session_connection_stats_log=true
```

Статистика записывается в файл `/usr/local/FlashphonerWebCallServer/logs/stats/media-session-connection-stats.log` в формате CSV

```
{mediaSessionId}; {channels_not_writable}; {decodable_drops_old}; {incomplete_drops_old};
{decodable_drops_reset}; {incomplete_drops_reset}; {decodable_drops_pli}; {incomplete_drops_pli};
{data_packets_with_empty_payload}; {missed_h264_units}; {dropped_audio_data_packets}
```

Здесь

- `mediaSessionId` - идентификатор медиа сессии
- `channels_not_writable` - количество событий, в результате которых не удалось записать данные на отправку в TCP сокет
- `decodable_drops_old` - количество сброшенных H264 фреймов, собранных из пакетов трафика
- `incomplete_drops_old` - количество сброшенных H264 фреймов, не полностью собранных из пакетов трафика
- `decodable_drops_reset` - количество сброшенных H264 фреймов до новой точки декодирования, собранных из пакетов трафика
- `incomplete_drops_reset` - количество сброшенных H264 фреймов до новой точки декодирования, не полностью собранных из пакетов трафика
- `decodable_drops_pli` - количество сбросов всех H264 фреймов, собранных из пакетов трафика, при приходе ключевого фрейма
- `incomplete_drops_pli` - количество сбросов всех H264 фреймов, не полностью собранных из пакетов трафика, при приходе ключевого фрейма
- `data_packets_with_empty_payload` - количество пакетов с пустым содержимым, высылаются браузером для оценки канала публикации при включенном TWCC
- `missed_h264_units` - количество потерянных H264 элементов
- `dropped_audio_data_packets` - количество аудио пакетов, отброшенных на этапе передачи в движок сервера

Пример записи

```
f49f8cb0-dc52-11ee-81df-51ad589334c0; 0; 0; 7; 0; 0; 0; 10; 0; 443; 0
```

Запись статистики в файл настраивается в `log4j.properties` следующим образом

```
log4j.logger.MediaSessionConnectionStats=error, mediaSessionConnectionStatsAppender
log4j.additivity.MediaSessionConnectionStats=false
log4j.appender.mediaSessionConnectionStatsAppender=com.flashphoner.common.logging.NewLogForEachRunFileAppender
log4j.appender.mediaSessionConnectionStatsAppender.DatePattern='.'yyyy-MM-dd-HH
log4j.appender.mediaSessionConnectionStatsAppender.layout=org.apache.log4j.PatternLayout
log4j.appender.mediaSessionConnectionStatsAppender.layout.ConversionPattern=%m%n
log4j.appender.mediaSessionConnectionStatsAppender.File=${com.flashphoner.fms.AppHome}/logs/stats/media-session-connection-stats.log
```

Если запись статистики медиасессии в файл включена, но в `log4j.properties` нет соответствующей настройки, статистика будет выводиться в серверный лог на ERROR уровне:

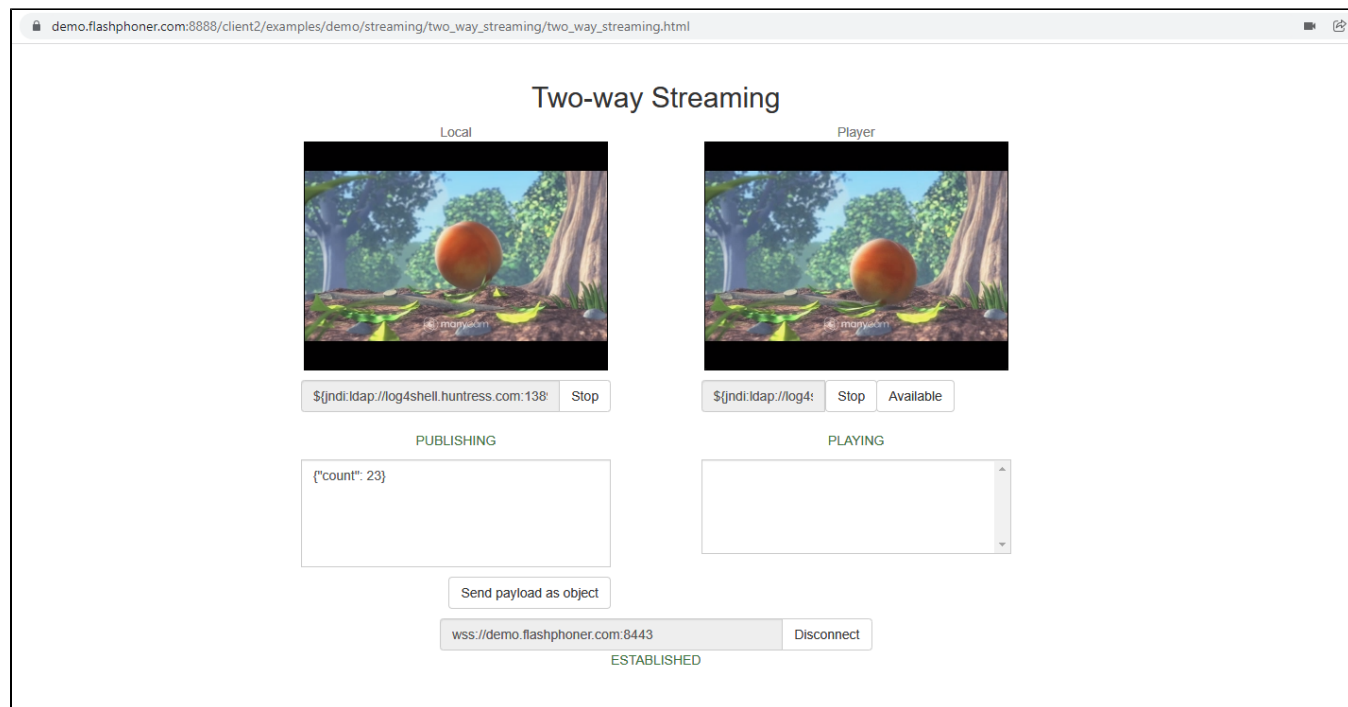
```
ERROR ssionConnectionStats - API-ASYNC-pool-7-thread-3 359943b0-fc64-11ee-bd5a-1dcdbd3939090; 0; 0; 0; 0; 0; 0; 0; 0; 40; 0; 0
```

Уязвимость CVE-2021-44228

Уязвимость [CVE-2021-44228](#) в библиотеке Apache log4j не может эксплуатироваться на WCS сервере. Конфигурирование логгера осуществляется только через файл `log4j.properties`, поэтому злоумышленник должен получить доступ к файловой системе сервера. Через поля ввода уязвимость не может быть проэксплуатирована:

1. Для проверки используем адрес <https://log4shell.huntress.com/>. По этому адресу генерируется уникальная ссылка, которую необходимо вставить в поля ввода.

2. Открываем пример Two Way Streaming на демо сервере https://demo.flashphoner.com:8888/client2/examples/demo/streaming/two_way_streaming/two_way_streaming.html, нажимаем Connect и вставляем ссылку в поля для имени потока. Публикуем и играем поток:



3. Открываем ссылку для просмотра результатов. В колонках IP address и Date/Time должны выводиться обращения от нашего сервера, если уязвимость сработала

log4shell.huntress.com/view/14d2fb6d-06f2-4809-973f-2a59627ca0f8

☆

100

Huntress Log4Shell Vulnerability Results

Any time a server reaches out to our LDAP server with your unique identifier, it will be logged here. You can use the payload you received on the home page to test various services in your network and check back here for any results. Your payload is:

```
{jndi:ldap://log4shell.huntress.com:1389/14d2fb6d-06f2-4809-973f-2a59627ca0f8}
```

⚠

The entries below are only cached for up to 30 minutes. If you need this data, you should copy it to a safe place.

ℹ

Looking for JSON results? You can download them from [here!](#)

IP Address	Date/Time
------------	-----------

Как видно, уязвимость через поля ввода не может быть проэксплуатирована в сборке WCS 5.2.1109

Пояснения: почему WCS не подвержен уязвимости

В составе WCS используется версия библиотеки Apache log4j 1.2.17. В данной версии нет поддержки JDNI, которая была добавлена начиная с log4j 2.0-beta9. Поэтому CVE-2021-44228 не может эксплуатироваться в WCS.