6 Мониторинг файлов журналов

Обзор

Zabbix можно использовать для централизованного мониторинга и анализа файлов журналов с/без поддержки ротации журналов.

Можно использовать оповещения для предупреждения пользователей, когда файл журнала содержит конкретные строки или шаблоны строк.

Для наблюдения за файлом журнала у вас должно быть:

- Работающий Zabbix агент на узле сети
- Настроенный элемент данных для мониторинга журнала

Максимальный размер наблюдаемого файла журнала зависит от поддержки файлов большого объема.

Настройка

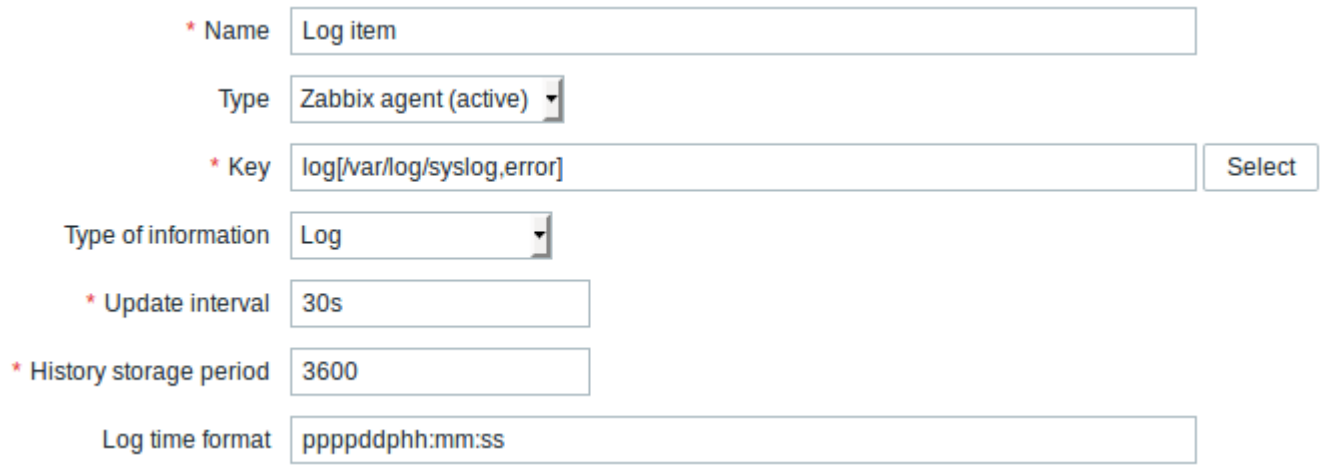
Проверка параметров агента

Убедитесь, что в файле конфигурации агента:

- Параметр 'Hostname' совпадает с именем узла сети в веб-интерфейсе
- Указаны сервера в параметре 'ServerActive' для обработки активных проверок

Настройка элемента данных

Настройте элемент данных для мониторинга журнала.



Все обязательные поля ввода отмечены красной звёздочкой.

Специально для элементов данных наблюдения за журналами вы должны указать:

Тип	Здесь выберите **Zabbix агент (активный)**.

Ключ	Укажите: **log**[/путь/к/файлу/имя_файла,<регулярное выражение>,<кодировка>,<макс. кол-во строк>,<режим>,<вывод>,<максзадержка>] или **logrt**[/путь/к/файлу/регулярное_выражение_описывающее_шаблон_имени_файла,<регулярное выражение>,<кодировка>,<макс. кол-во строк>,<режим>,<вывод>,<максзадержка>] Zabbix агент фильтрует записи из файла журнала по регулярному выражению, если оно указано. Если требуется только количество совпадающих строк укажите: **log.count**[/путь/к/файлу/имя_файла,<регулярное выражение>,<кодировка>,<макс. кол-во строк>,<режим>,< максзадержка >] или **logrt.count**[/путь/к/файлу/регулярное_выражение_описывающее_шаблон_имени_файла,<регулярное выражение>,<кодировка>,<макс. кол-во строк>,<режим>,< максзадержка >]. Убедитесь, что у файла имеются права на чтение для пользователя 'zabbix', в противном случае состояние элемента данных будет 'unsupported'. Для получения более подробных сведений смотрите информацию о ключах **log**, **log.count**, **logrt** и **logrt.count** в разделе поддерживаемых ключей элементов данных Zabbix агентом.
Тип информации	Выберите здесь Журнал (лог) для элементов данных **log** и **logrt** или Числовой (целое положительное) для элементов данных **log.count** и **logrt.count**. Если используется опциональный параметр вывод, вы можете выбрать подходящий тип информации, отличный от “Журнал (лог)”. Обратите внимание, что выбор не журнального типа информации приведет к потере локального штампа времени.
Интервал обновления (в сек)	Этот параметр задает как часто Zabbix агент будет проверять наличие любых изменений в файле журнала. Указав этот параметр равным 1 секунде, вы можете быть уверенными, что получите новые записи как можно скорее.
Формат времени журнала	В этом поле вы можете опционально задать шаблон для анализа штампа времени строки журнала. Если оставить пустым, штамп времени не будет анализироваться. Поддерживаемые значения: * **y**: *Год (0001-9999)* * **M**: *Месяц (01-12)* * **d**: *День (01-31)* * **h**: *Час (00-23)* * **m**: *Минута (00-59)* * **s**: *Секунда (00-59)* Например, рассмотрим следующую строку из файла журнала Zabbix агента: “ 23480:20100328:154718.045 Zabbix agent started. Zabbix 1.8.2 (revision 11211).” Она начинается шестью символами обозначающими PID, далее следует дата, время, и остальная часть строки. Форматом времени журнала для этой строки является “ppppp:yyyyMMdd:hhmmss”. Обратите внимание, что символы “p” и “:” являются лишь заменителями и могут быть чем угодно, за исключением “yMdhms”.

Важные замечания

- Сервер и агент следят за размером наблюдаемого журнала и временем последней модификации (для logrt) двумя счетчиками. Дополнительно:
  - Также агент использует номера inode (на UNIX/GNU/Linux), индексы файлов (на Microsoft Windows) и MD5 суммы первых 512 байт файла журнала для улучшения выбора в случае когда файлы журнала усекаются и ротируются.
  - На системах UNIX/GNU/Linux предполагается, что файловые системы где хранятся файлы журналов, сообщают числа inode, которые могут быть использованы для слежения за состоянием файлов.
  - На системах Microsoft Windows Zabbix агент определяет тип файловой системе на которой находятся файлы журналов:
    - На файловой системе NTFS 64-битные файловые индексы.
    - На файловых системах ReFS (только Microsoft Windows Server 2012) 128-битные файловые ID.
    - На файловых системах где файловые индексы меняются (т.е. FAT32, exFAT)

используется запасной алгоритм для получения разумного подхода в неопределенных условиях, когда сжатие файла журнала приводит в результате к множеству файлов журналов с одинаковым временем изменения.

- Номера inode, индексы файлов и суммы MD5 собираются Zabbix агентом. Они не передаются Zabbix серверу и теряются в случае остановки Zabbix агента.
- Не меняйте время последней модификации файлов журналов, используя утилиту 'touch', не копируйте файл журнала с последующим восстановлением его имени (это изменит идентификатор иноды файла). В обоих случаях файл будет рассматриваться как другой и будет проанализирован с самого начала, что может привести к дубликатам оповещений.
- Если есть несколько совпадающих файлов журналов для элемента данных `logrt[]` и Zabbix агент следит за наиболее новым из них и этот более новый файл журнал удаляется, предупрежающиее сообщение будет записано “`there are no files matching ”<regexp mask>“ in ”<directory>“`. Zabbix агент игнорирует файлы журналы с временем изменения меньше чем последнее время модификации полученное агентом во время проверки элемента данных `logrt[]`.

- Агент начинает читать файл журнала с той позиции, на которой он остановился последний раз.
- Количество байт уже проанализированное (счётчик размера) и время последней модификации (счетчик времени) хранятся в базе данных Zabbix и отправляются агенту, для уверенности, что агент начнет читать файл журнала с этой позиции в случаях, когда агент только что был запущен или агент получил элементы данных, которые были ранее деактивированы или не поддерживались. Однако, если агент получает ненулевой размер счётчика от сервера, но элементы данных logrt[] или logrt.count[] не найдены и не удается найти соответствующие файлы, счётчик размера сбрасывается в 0, чтобы начать анализ сначала, если файлы появятся позже.
- Всякий раз, когда файл журнала становится меньше, чем значение счетчика размера известное агенту, счетчик обнуляется и агент начинает читать файл журнала с самого начала, принимая во внимание счетчик времени.
- Если есть несколько файлов журналов, с одинаковым последним временем модификации файла в соответствующей папке, агент пытается корректно проанализировать все файлы журналы с одинаковым временем модификации и избежать пропущенных данных или проанализировать данные дважны, несмотря на это невозможно охватить все возможные ситуации. Агент не предполагает какую либо определенную схему ротации файлов журналов, либо определяет ее. Когда есть несколько фалов журналов с одинаковым последним временем изменения, агент будет обрабатывать их лексикографически в порядке убывания. Таким образом, для некоторых схем ротации файлы журналы будут проанализированы в их оригинальном порядке. Для других же схем ротации журналов первоначальный порядок файла журнала не будет соблюдаться, что может привести к получению найденных по шаблону строк файла журнала в измененном порядке (проблема не случится, если файлы журнала имеют разное время последнего изменения).
- Zabbix агент обрабатывает новые записи файла журнала один раз за *Период обновления* секунд.
- Zabbix агент отправляет не более чем **макс. кол-во строк** записей из файла журнала за секунду. Это ограничение предотвращает перегрузку сети и ресурсов процессора и переопределяет значение по умолчанию предусмотренное параметром **MaxLinesPerSecond** в файле конфигурации агента.
- Для поиска необходимой строки Zabbix обрабатывает в 10 раза больше строк, чем указано в параметре MaxLinesPerSecond. Таким образом, например, если элемент данных

`log[]` или `logrt[]` имеет *Интервал обновления* 1 секунда, по умолчанию агент будет анализировать не более чем 400 строк файла журнала и будет отправлять не более чем 200 совпавших записей Zabbix серверу за одну проверку. Увеличением параметра **MaxLinesPerSecond** в файле конфигурации агента или указанием параметра **макс. кол-во строк** в ключе элемента данных, лимит можно увеличить вплоть до 10000 проанализированных записей в журнале и 1000 совпадающих записей для отправки Zabbix серверу за одну проверку. Если *Интервал обновления* указан значением в 2 секунды, лимиты для одной проверки могут быть увеличены в два раза больше, чем для *Интервала обновления* в 1 секунду.

- Кроме того, данные из файлов журналов всегда ограничены 50% размера буфера отправки у агента, даже если в буфере нет значений не связанных с данными из файлов журналов. Таким образом, значения **макс. кол-во строк** будут отправлены за одно соединение (а не в нескольких соединений), параметр BufferSize агента должен быть по крайней мере равен макс. кол-во строк x 2.
- При отсутствии данных для элементов данных журналов весь размер буфера используется для значений не связанных с данными из журналов. Когда появляются значения от файлов журналов они заменяют устаревшие данные не связанные с файлами журналов, если требуется, до максимального уровня 50%.
- Для записей в файле журнала длиннее 256КБ, только первые 256КБ сопоставляются с регулярным выражением, остальная часть игнорируется. Однако, если Zabbix агент был остановлен в процессе обработки длинной строчки, внутреннее состояние агента теряется и длинная строчка может быть проанализирована иначе после запуск агента.
- Специальное примечание для разделителей пути “\”: если формат файла представлен как “file\.log”, тогда там не должно быть папки “file”, поскольку невозможно однозначно определить, экранируется ли это символ ”.“ или это первый символ в имени файла.
- Регулярные выражения для **logrt** поддерживаются только в именах файлов, совпадение регулярного выражения с папкой не поддерживается.
- В UNIX элементы данных `logrt[]` становится НЕПОДДЕРЖИВАЕМЫМ, в случае если папка не существует где файл журнала должен был бы находиться.
- В Microsoft Windows, если папка не существует элемент данных не переводится в состояние НЕПОДДЕРЖИВАЕТСЯ (например, если в ключе элемента данных папка указана с ошибкой)
- Отсутствие файла журнала для элемента данных `logrt[]` не переводит его в состояние НЕПОДДЕРЖИВАЕТСЯ.
- Ошибки чтения файлов журналов для элемента данных `logrt[]` записываются в журнал агента как предупреждения, но не переводят элемент данных в состояние НЕПОДДЕРЖИВАЕТСЯ.
- Журнал Zabbix агента может быть очень полезен для поиска причин почему элементы данных `log[]` или `logrt[]` становятся НЕПОДДЕРЖИВАЕМЫМИ. Zabbix может мониторить свой файл журнала, за исключением случая когда он в режиме DebugLevel=4.

Извлечение совпадающей части регулярного выражения

Иногда мы можем захотеть извлечь только интересующие значения из требуемого файла вместо того, чтобы получать всю строку, в случае когда найдено совпадение с регулярным выражением.

Начиная с Zabbix 2.2.0, элементы данных файлов журналов расширены возможностью получения извлечения требуемых значений из строк файла. Добавился дополнительный параметр **вывод** у элементов данных `log` и `logrt`.

Использование параметра 'вывод' позволяет обозначить подгруппу совпадения в которой мы можем быть заинтересованы.

И так, например

```
log[/path/to/the/file,"large result buffer allocation.*Entries:
([0-9]+)",,,,\1]
```

должно позволить получить количество записей со следующего содержания:

```
Fr Feb 07 2014 11:07:36.6690 */ Thread Id 1400 (GLEWF) large result
buffer allocation - /Length: 437136/Entries: 5948/Client Ver: >=10/RPC
ID: 41726453/User: AUser/Form: CFG:ServiceLevelAgreement
```

Причина, почему Zabbix вернет только одно число, потому что параметр 'вывод' здесь определен как **\1** ссылка только на первую интересующую подгруппу: **([0-9]+)**

Вместе с возможностью извлечения и получения числа, значение можно использовать в определениях триггеров.

Использование параметра максзадержка

Параметр 'максзадержка' в элементах данных журналов позволяет игнорировать более старые строки с целью получения наиболее новых строк проанализированных в течении “максзадержка” секунд.

Параметр 'maxdelay' > 0, может привести к **игнорированию важных записей в файлах журналов и пропуску оповещений**. Используйте этот параметр осторожно и на свой страх и риск, только в случае необходимости.

По умолчанию элементы данных мониторинга журналов забирают все новые строки появляющиеся в файлах журналов. Однако, имеются приложения, которые в некоторых ситуациях начинают записывать огромное количество сообщений в свои файлы журналов. Например, если база данных или DNS сервер недоступны, то такие приложения могут флудить файлы журналов тысячами практически идентичных сообщений об ошибке до тех пор пока не восстановится нормальный режим работы. По умолчанию, все эти сообщения добросовестно анализируются и совпадающие строки оправляются на сервер, как настроено в элементах данных `log` и `logrt`.

Встроенная защита от перегрузов состоит из настраиваемого параметра 'макс. кол-во строк' (защищающий сервер от слишком большого количества приходящих совпадающих строк в журнале) и ограничения в 4*'макс. кол-во строк' (защищает CPU и I/O хоста от перегрузки агентам одной проверкой). Тем не менее имеется 2 проблемы со встроенным механизмом защиты. Первая, на сервер будет отправлено большое количество потенциально не так информативных сообщений, которые займут место в базе данных. Вторая, по причине ограниченного количества строк анализируемых в секунду агент может отставать на часы от

самых новых записей в журнале. Вполне вероятно, что вы захотите как можно быстрее быть информированным о текущей ситуации в файлах журналов вместо ковыряния часами старых записей.

Решение этих двух проблем является использование параметра 'максзадержка'. Если параметр 'maxdelay' > 0, во время каждой проверки измеряются количество обработанных байт, количество оставшихся байт и время обработки. Отталкиваясь от этих значений, агент вычисляет оценочную задержку - как много секунд может потребоваться, чтобы проанализировать все оставшиеся записи в файле журнала.

Если задержка не превышает 'максзадержка', тогда агент поступает с анализом файла журнала как обычно.

Если задержка больше чем 'максзадержка', тогда агент **игнорирует часть файла журнала, “перепрыгивая” эту часть** к новой оценочной позиции таким образом, чтобы оставшиеся строки можно было проанализировать за 'максзадержка' секунд.

Обратите внимание, что агент даже не читает проигнорированные строки в буфер, но вычисляет приблизительную позицию для прыжка в файле.

Сам факт пропуска строк в файле журнала записывается в файл журнала агента, примерно следующим образом:

```
14287:20160602:174344.206
item:"logrt["/home/zabbix32/test[0-9].log",ERROR,,1000,,,120.0]"
logfile:"/home/zabbix32/test1.log" skipping 679858 bytes
(from byte 75653115 to byte 76332973) to meet maxdelay
```

Количество “to byte” является оценочным, потому что после “прыжка” агент скорректирует позицию в файл к началу строки в журнале, которая может быть в файле чуть дальше или раньше.

В зависимости от того как скорость роста соотносится к скорости анализа файла журнала, вы можете не увидеть “прыжков”, а можете увидеть редкие или частые “прыжки”, большие или маленькие “прыжки”, или даже маленькие “прыжки” каждую проверку. Колебания загрузки системы и сетевые задержки также влияют на вычисления задержки и, следовательно, “прыжки” вперед чтобы не отставать от параметра “максзадержка”.

Не рекомендуется указывать 'максзадержка' < 'интервал обновления' (это может привести к частым маленьким “прыжкам”).

Заметки по обработке ротации 'copytruncate' файлов журналов

`logrt` с опцией `copytruncate` подразумевает, что разные файлы журналов имеют разные записи (по крайней мере штампы времени в них отличаются), поэтому MD5 суммы начальных блоков (до первых 512 байт) будут отличаться. Два файла с одинаковыми MD5 суммами начальных блоков означают, что один из них оригинал, а второй - копия.

`logrt` с опцией `copytruncate` делает попытку правильной обработки копий файлов журналов

без дублирующих сообщений. Тем не менее, такие варианты как создание нескольких копий файлов журналов с одинаковыми штампами времени, ротация файлов журналов чаще чем интервал обновления logrt[] элемента данных, частый перезапуск агента не рекомендуются. Агент пытается справиться со всеми этими ситуациями, но хорошие результаты не гарантируются при всех обстоятельствах.

Действия, если произошла ошибка связи между агентом и сервером

Каждая совпадающая строка с элементов данных `log[]` и `logrt[]` и результат проверки каждого элемента данных `log.count[]` и `logrt.count[]` требует свободный слот в выделенной 50% области буфера отправки в агенте. Элементы буфера регулярно отправляются серверу (или прокси) и слоты буфера становятся снова пустыми.

Пока имеются свободные слоты в выделенной области для журналов в буфере отправки в агенте и связь между агентом и сервером (или прокси) нарушена, результаты мониторинга журналов накапливаются в буфере отправки. Такое поведение позволяет смягчить кратковременные нарушения связи.

Во время длительных нарушений свящи все слоты журналов становятся занятыми и выполняются следующие действия:

- Проверки элементов данных `log[]` и `logrt[]` останавливаются. Когда связь восстановится и появятся свободные слоты, проверки вернутся к предыдущей позиции. Не совпадающие строки потеряются. Совпадающие строки не будут потеряны, они просто отправятся позже.
- Проверки `log.count[]` и `logrt.count[]` останавливаются, если `maxdelay = 0` (по умолчанию). Поведение похоже на элементы данных `log[]` и `logrt[]`, описанное выше. Обратите внимание, что потеря связи может повлиять на результаты `log.count[]` и `logrt.count[]`: например, одна проверка насчитает 100 совпадающих строк в файле журнала, но по причине отсутствия свободных слотом в буфере проверка будет остановлена. Когда связь восстановится агент насчитает те же 100 совпадающих строк, а также 70 новых совпадающих строк. После чего агент отправит количество = 170, так как они найдены за одну проверку.
- Проверки `log.count[]` и `logrt.count[]` при `maxdelay > 0`: если не было “прыжка” во время проверки, тогда поведение аналогично описанному выше. Если всё же был “прыжок” через строки файла журнала, тогда позиция после “прыжка” сохранится и подсчитанный результат будет отброшен. Таким образом, агент пытается не отставать от увеличивающегося файла журнала, даже в случае проблем со связью.