Java Agent v2.2.0

배포일: 2022-11-01

New Feature (새 기능)

- virtual 애플리케이션에서 logsink 데이터를 생성하도록 기능 추가
- logsink의 stdout 로그도 txid 정보를 연결할 수 있도록 기능 추가
- 2초 데이터 수집 기능 추가

  whatap.conf

  ```
  fast_perf_enabled=false
  fast_perf_interval=2000
  fast_perf_gc_enabled=true
  fast_perf_os_enabled=true
  debug_fast_perf_enabled=false
  ```

  수집데이터

  - 경과시간
  - 액티브 트랜잭션
  - gc 정보: gc건수, gc경과시간, old generation gc건수, old generation gc경과시간
  - OS 정보

- 액티브 트랜잭션 건수에 따라 알림을 보내도록 기능 추가

  whatap.conf

  ```
  # 엑티브 트랜잭션 알림 설정 여부
  too_many_actx_alert_enabled=false

  # 엑티브 트랜잭션 알림 무시 시간 (ms)
  too_many_actx_alert_silent_time=30000

  # 엑티브 트랜잭션 알림 기준 건수
  too_many_actx_alert_count=50

  # 엑티브 트랜잭션 알림 메시지 표시 건수
  too_many_actx_alert_msg_url_count=5
  ```

```
# 엑티브 트랜잭션 알림 기준 시간
too_many_actx_alert_over_time=30000
```

  - 엑티브 트랜잭션 기준 건수는 `too_many_actx_alert_count >= (too_many_actx_alert_over_time / 5000)`으로 결정
  - 엑티브 트랜잭션 기준 건수와 같거나 넘어가는 경우 `too_many_actx_alert_msg_url_count` 만큼의 url 정보를 알림 메시지에 추가해 알림 전송

- 프로파일 디버그 옵션: 디버그 대상 패턴을 등록한 경우 프로파일과 에이전트 로그에서 스텍을 확인할 수 있는 기능 추가

whatap.conf

```
hook_method_debug_enabled=true
hook_method_debug_patterns=
```

- 플러그인 활용 클래스 추가
  - {AGENT_HOME}/plugin/{...}.x 파일에서 활용할 수 있도록 `HashTable` 클래스를 추가하였습니다.
  - `whatap.agent.api.Ref.h`를 사용하면 `HashTable`을 활용할 수 있습니다.

Example

```
Class hashTable = (Class) whatap.agent.api.Ref.h;
```

- TTA BMT를 위한 전용 클래스들 추가: 자바 에이전트 시연을 위해 활용 가능

Java Class

```
# map
bmt.m.MyMap

# datasource
bmt.ds.MyDataSource
bmt.ds.MyConnection

# jdbc
bmt.jdbc.Config
bmt.jdbc.FakeBlob
bmt.jdbc.FakeClob
bmt.jdbc.FakeConnection
bmt.jdbc.FakeDatabaseMetaData
```

```
bmt.jdbc.FakeDriver
bmt.jdbc.FakeCallableStatement
bmt.jdbc.FakePreparedStatement
bmt.jdbc.FakeResultSet
bmt.jdbc.FakeStatement
bmt.jdbc.FakeSocket
bmt.jdbc.Lock
```

Tracing

```
// Connection Pool 추적
// ConnectionPoolASM
target.put("bmt/ds/MyDataSource", MyDataSource.className);

// Connection Pool 추적
// DBConCountCollectorThread
add(new MyDataSource());

// JDBC Connection 추적
// JDBCConnectionOpenASM
AsmUtil.add(reserved, "bmt/ds/MyDataSource", "getConnection()Ljava/sql/Connection;");

// JDBC Statement 추적
// JDBCStatementASM
target.add("bmt/jdbc/FakeStatement");

// JDBC PreparedStatement 추적
// JDBCPreparedStatementASM
target.add("bmt/jdbc/FakePreparedStatement");

// JDBC ResultSet 추적
// JDBCResultSetASM
target.add("bmt/jdbc/FakeResultSet");
```

- 에러 스텝
  - 프로파일에 에러 스텝을 추가할 수 있도록 기능 추가
  - 설정 추가해 프로파일에 에러 관련 정보를 표시하는 스텝 추가

whatap.conf

```
profile_error_step_enabled=false
```

- jmx 설정 추가: 임의의 jmx 정보를 추가로 조회할 수 있는 기능 추가

whatap.conf

```
perfx_tomcat_enabled=false
perfx_jmx_value_limit=100
perfx_jmx_tag_from_objectname_enabled=false
```

tomcat의 jmx 정보를 수집하여 메트릭스 조회에서 test1, test2로 확인하기 위한 설정은 다음과 같습니다.

whatap.conf

```
perfx_jmx_enabled=true
perfx_jmx@test1=Catalina:type=Connector,*
perfx_jmx@test2=java.lang:type=GarbageCollector,*
perfx_jmx_ignore@test2=LastGcInfo
# perfx_debug_enabled=true
# perfx_jmx_tag_from_objectname_enabled=false
```

- oshi 라이브러리 활용, 프로세스의 proc_mem_rss 데이터를 추적 기능 추가

whatap.conf

```
oshi_proc_mem_enabled=false
```

- `org.edb.jdbc.PgCallableStatement`를 jdbc statement 추적 기능 추가
- 오라클 프로시저에서 SQL 파라미터 추적 기능
  - 오라클 프로시저에서 SQL 파라미터를 추적할 수 있는 기능을 추가하였습니다.
  - CallableStatement에서 SQL 파라미터를 추적할 수 있도록 기능을 추가하였습니다.
- Weblogic에서 `HttpURLConnection`으로 http 호출을 하는 경우 `weblogic.net.http.HttpURLConnection`을 사용하는 것을 추적하는 기능 추가

whatap.conf

```
HttpURLConnection_weblogic=true
```

- sigar 라이브러리를 활용, 프로세스의 네이티브 메모리 데이터(proc_mem_rss)를 수집하는 기능 추가
- 기존 30초 평균으로 수집한 tps 데이터를 5초 평균으로 수집하는 기능 추가

whatap.conf

```
service_metrics_spike_enabled=false
```

- 프로파일 데이터 압축 전송 기능 추가
  - 프로파일 데이터가 `profile_zip_max_buffer_size` 크기를 넘는 경우 프로파일 데이터를 전송
  - 큐에 데이터가 쌓인지 `profile_zip_max_wait_time` 값을 넘은 경우 프로파일 데이터를 전송
  - 프로파일 데이터를 전송할 때 큐에 있는 프로파일 데이터가 `profile_zip_min_size` 크기 미만인 경우를 제외하고는 프로파일 데이터를 압축 전송

whatap.conf

```
# 프로파일 데이터 압축 여부
profile_zip_enabled=true

# 프로파일 데이터들을 가지고 있을 큐 사이즈
# ZipProfileThread.java가 사용하는 RequestQueue 사이즈를 결정
profile_zip_queue_size=1000

# 프로파일 데이터를 보내는 대기 시간
# 1000ms 지난 경우 프로파일 데이터 전송
profile_zip_max_wait_time=1000

# 프로파일 데이터 버퍼 사이즈
# 프로파일 데이터가 1MB가 넘는 경우 데이터를 전송
profile_zip_max_buffer_size=1024*1024

# 프로파일 데이터 압축 여부 결정 최소 사이즈
# 프로파일 데이터가 100byte 미만인 경우 압축하지 않음
profile_zip_min_size=100

# 프로파일 데이터 압축 디버그 여부
# 프로파일 데이터 전송 관련 정보(ZipPack의 status, recordCount, buffer 사이즈, 프로파일 데이터 큐 사이즈)를 에이전트 로그에 출력
debug_profile_zip_enabled=false

# 프로파일 데이터 압축 디버그 로그 출력 간격
# 프로피알 데이터 압축 디버그 옵션이 켜져 있는 경우 5000ms 간격으로 프로파일 데이터 전송 관련 정보를 에이전트 로그에 출력
debug_profile_zip_interval=5000
```

Change (업데이트)

- logsink 데이터 전송 주기인 `max_wait_time`을 `logsink_max_wait_time`으로 옵션 이름 수정
- 플러그인에서 수집하는 CustomCounter 기능을 기존 CustomCounterTask 대신에 CustomTaskLoader로 대체

whatap.conf

```
custom_task_enabled=true
custom_task_jarfile={full_path}
```

- 에이전트-proxy간 서버시간 차이가 10초 이상인 경우 로그를 출력하도록 기능 추가
- 경과시간 측정
  - 트랜잭션과 프로파일 경과시간 측정 시 시스템 시간이 아닌 JVM 시간을 사용하도록 수정
  - 경과시간을 측정하는 경우 기존 `System.currentTimeMillis()`에서 `System.nanoTime()`을 사용하도록 수정
- 에이전트-proxy의 서버시간 차이가 있는 경우 에이전트 로그에 서버간 시간 차이를 표시하도록 수정

Agent log

```
# 30초 이상 차이가 나는 경우
"fatal warning time-reverse (시간 차이) ms"

# 30초 미만 차이가 나는 경우
"warning time-reverse (시간 차이) ms"
```

- 사용자 카운팅을 위한 옵션들을 wclient_xxx 형태로 수정, 이전 옵션 이름도 사용할 수 있지만, v2.2.0부터는 새로운 옵션 이름 사용 권장

whatap.conf

```
# 교체
trace_user_method=IP
wclient_trace_type=IP

# 교체
trace_user_jsession_key=JSESSIONID
wclient_jsession_key=JSESSIONID

# 교체
trace_user_header_key=token
wclient_header_key=token

# 교체
trace_user_cookie_limit=4000
wclient_cookie_limit=4000

# 교체
trace_user_cookie_domain=
```

```
wclient_cookie_domain=

# 교체
realtime_user_thinktime_max=300000
wclient_thinktime_max=300000

# 교체
trace_user_using_ip=true
wclient_using_ip=true

# 교체
user_header_ticket=
wclient_header_ticket=

# 추가
wclient_max_count=70000
```

- 사용자 수 관련 카운팅 설정 반영해 내장 weaving 컴포넌트 재컴파일
- CounterPack 경과시간 계산에 `System.nanoTime()`을 사용하도록 수정
- 로그 출력 로고에 릴리즈 버전 포함

Logo

```
  _     ____  ______
| |    / / /_/_  __/___ _____  ___ ___
| | /| / / __ \/ / / __ `/ __ \| _ \|_  )
| |/ |/ / / / / / / /_/ / /_/ / |  / / /
|__/|__/_/ /_/_/  \__,_/ .___/  |_|_\/___|
                      /_/
The Best Observability in Java
```

- 자바에이전트의 통계 수집 기본값을 기존 5000에서 10000으로 수정, 통계 데이터 전송 시 2초마다 체크

whatap.conf

```
stat_tx_max_count=10000
```

- 성능지표를 수집하는 oshi 라이브러리 버전을 기존 oshi-core-5.4.1에서 oshi-core-6.1.6으로 업데이트
- 성능지표를 수집하는 {AGENT_HOME}의 라이브러리 폴더 이름 변경
  - sigar 라이브러리 폴더 이름 변경 : lib -> lib3
  - oshi 라이브러리 폴더 이름 변경 : lib2 -> lib4

Fixed (버그 수정)

- 자바에이전트의 내장 플러그인 Webflux 위빙플러그인의 read-only인 HttpHeaders 항목에 멀티트랜잭션 관련 헤더(header)를 추가하는 경우 발생하는 에러 수정

  webflux-5.1, webflux-5.2, webflux-5.3 버전에 전부 적용하였습니다.

- 트랜잭션과 스텝의 경과시간이 0보다 작은 경우 발생하는 버그 수정
- 트랜잭션과 스텝의 경과시간이 0보다 작은 경우 경과시간을 0으로 표시하는 버그 수정.
- `blocking_detect_count` 건수 처리 버그를 수정