Go Agent v0.1.8

배포일: 2022-01-19

베타 - 업데이트

동시접속 사용자 수집 설정 추가

동시접속 사용자 지표는 5분전 부터 현재까지의 고유한 사용자의 수를 표시합니다. 5분간 합산된 사용자 수가 표시 됩니다. 고유한 사용자 정보를 HyperLogLog 알고리즘으로 처리합니다.

고유한 사용자 정보는 초기에 IP로 설정됩니다. 추가로 HTTP Header , Cookie 항목의 값으로 사용자 식별 정보를 설정할 수 있습니다.

설정

- `trace_user_header_ticket`

  #Default : ""#

  #Type : string#

  설정된 HTTP Header의 값을 고유한 사용자 정보로 선택합니다.

- `trace_user_cookie_keys`

  #Default : ""#

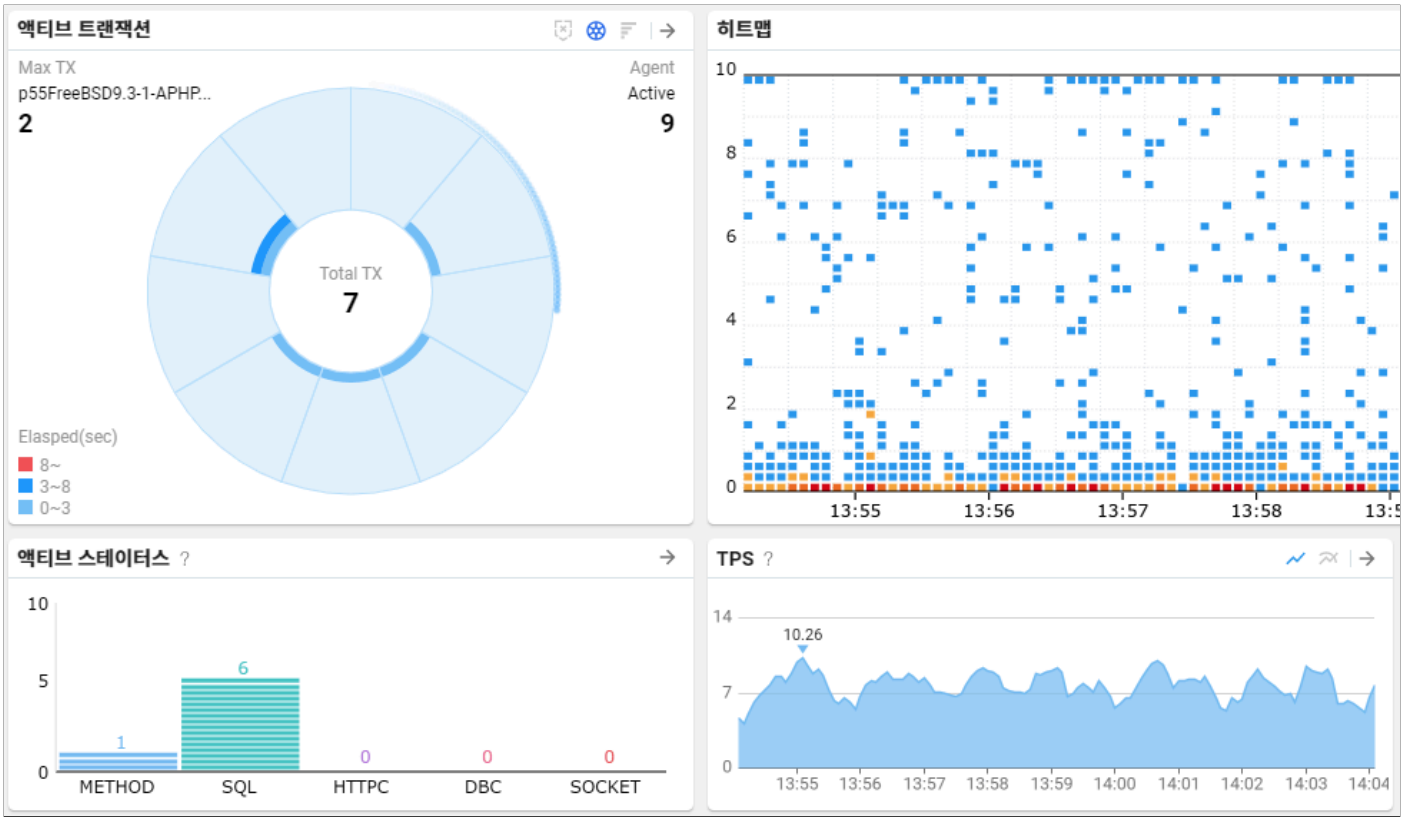
  #Type : string#

  설정된 Cookie의 값을 고유한 사용자 정보로 선택합니다. 콤마(,) 구분으로 여러개의 이름을 설정할 수 있습니다. 설정된 순서대로 검색합니다.

액티브 스테이터스

액티브 스테이터스 기능을 추가합니다.

대시보드 화면에 액티브 스테이터스 그래프가 표기됩니다. 상태는 일반 함수, SQL, DB connection, 외부 HTTP 연결, 소켓 연결이 있으며 각 상태별로 진행중인 요청 개수를 표시합니다.



내부 지표 수집

별도의 goroutine을 실행하여 5초 간격으로 지표를 수집합니다.

runtime 지표 수집

분석 > 메트릭스 차트에서 카테고리 `go_runtime`를 선택해서 볼 수 있습니다.

수집 지표

지표 이름	설명
NumCpu	현재 프로세스에서 사용할 수있는 논리 CPU 수
NumCgoCall	현재 프로세스에서 수행 한 cgo 호출 수
NumGoroutine	현재 존재하는 고 루틴의 수
Alloc	할당 된 힙 개체의 바이트

지표 이름	설명
TotalAlloc	힙 개체에 할당 된 누적 바이트
Sys	OS에서 얻은 총 메모리 바이트
Lookups	-
Mallocs	할당 된 힙 개체의 누적 개수
Frees	해제 된 힙 개체의 누적 개수
HeapAlloc	할당 된 힙 개체의 바이트
HeapSys	OS에서 얻은 힙 메모리의 바이트
HeapIdel	사용되지 않는 바이트
HeapInuse	사용중인 바이트
HeapReleased	OS에 반환 된 물리적 메모리의 바이트
HeapObjects	할당 된 힙 개체의 수
StackInuse	사용중인 스택 바이트
StackSys	OS에서 얻은 스택 메모리의 바이트
MSpanInuse	할당 된 mspan 바이트
MSpanSys	mspan에 대해 OS에서 얻은 메모리 바이트
MCacheInuse	할당 된 mcache 바이트
MCacheSys	OS에서 얻은 메모리의 mcache 바이트
BuckHashSys	버킷 해시 테이블을 프로파일링하는 메모리의 바이트

지표 이름	설명
GCSys	가비지 컬렉션 메타 데이터의 메모리 바이트
OtherSys	기타 off-heap 메모리 바이트 (런타임 할당)
NextGC	다음 GC주기의 대상 힙 크기
LastGC	마지막 가비지 수집이 완료된 시간 (unixstamp nanosecond)
PauseTotalNs	GC의 누적 나노초
NumGC	완료된 GC주기의 수
NumForcedGC	강제 된 GC주기의 수

net/http 라이브러리의 RoundTripper 지원

http transport에 RoundTripper를 사용할 수 있도록 지원합니다. 패키지의 `sql.Open` 함수 대신 `whatapsql.OpenContext` 함수를 사용합니다. `PrepareContext`, `QueryContext`, `ExecContext` 등 context를 전달하는 함수를 사용하기를 권장합니다.

전달하는 context는 `trace.Start()`를 통해서 whatap TraceCtx 정보가 있어야합니다.

Install guide

```
import (
  "github.com/whatap/go-api/instrumentation/net/http/whataphttp"
)

func main() {
  config := make(map[string]string)
  trace.Init(config)
  defer trace.Shutdown()

  ctx, _ := trace.Start(context.Background(), "Http call")
  defer trace.End(ctx, nil)

  callUrl := "http://aaa.com/xxx"
  client := http.DefaultClient
```

```
    // Use WhaTap RoundTripper. whatap의 TraceCtx가 있는 context 를 전달합니다.
    client.Transport = whataphttp.NewRoundTrip(ctx, http.DefaultTransport)

    resp, err := client.Get(callUrl)
    if err != nil {
        ...
    }
    defer resp.Body.Close()

    ...
}
```

참조 예제