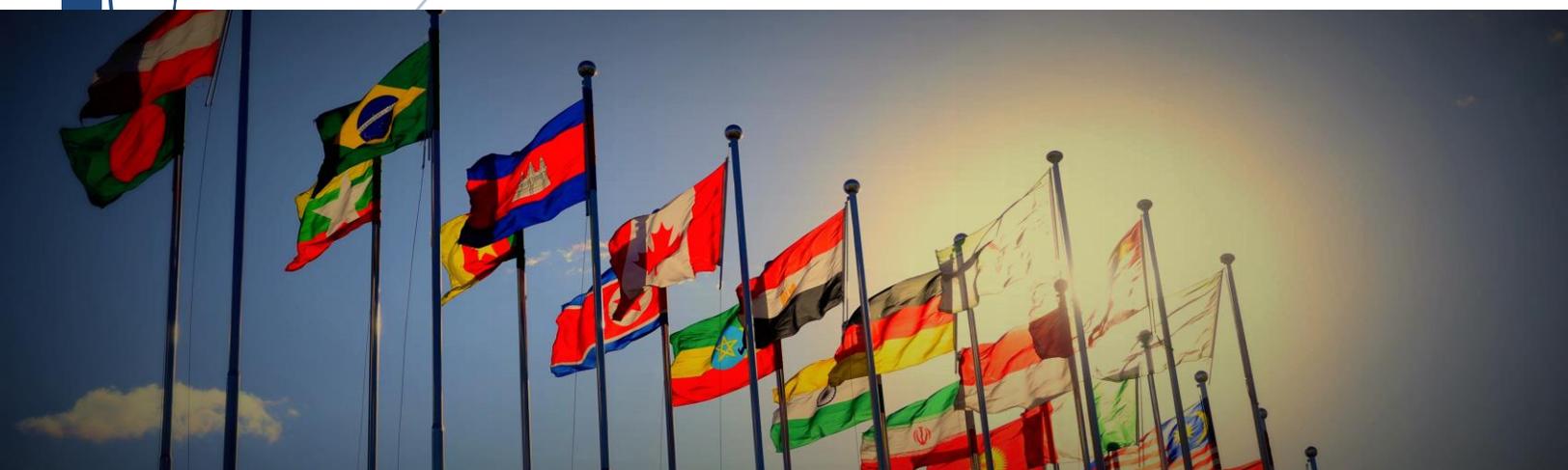


Heimdall 数据访问 平台安装与设置



目 录

1	HEIMDALL 数据访问平台概述	4
1.1	HEIMDALL 的名称	4
1.2	主要功能.....	5
1.3	工作原理.....	8
1.4	功能列表.....	9
2	安装	11
2.1	系统需求.....	11
2.2	免费版与付费版	11
2.3	简明安装过程	12
2.4	WINDOWS 版本安装	12
2.5	非 WINDOWS 版本安装.....	16
2.6	修改默认端口	17
3	使用示例程序 TRAFFIC GENERATOR.....	18
3.1	下启动 TRAFFIC GENERATOR.....	18
3.2	测试数据库连接	19
3.3	监视数据库访问动态	20
4	应用接入 HEIMDALL 平台.....	22
4.1	简明安装步骤	22
4.2	安装 HEIMDALLDRIVER.JAR.....	23

Heimdall 数据访问平台 的安装与设置



4.3	安装 HAZELCAST-3.6.JAR.....	23
4.4	创建 DATA SOURCE , 配置 JDBC.....	24
4.5	创建 VIRTUAL DATABASES , 计算 HDAP 参数.....	26
4.6	将 HDAP 参数填写到应用配置文件中.....	26
4.7	重启应用.....	29
4.8	通过 ANALYTICS 分析数据库访问性能.....	30
4.9	通过设置 RULES 调整缓存策略.....	32
4.10	通过 DASHBOARD 查看性能.....	33
5	性能分析.....	36
5.1	基准性能测试.....	40
5.2	HEIMDALL 性能基准测试步骤.....	40
5.3	成功进行性能测试的几点提示 :	42
6	利用 HDAP 实现 HA	43
7	HDAP 规则说明	44
8	技术支持.....	46

1 Heimdall 数据访问平台概述

Heimdall Data Access Platform (Heimdall 数据访问平台，简称为 HDAP) 为应用数据访问提供性能优化，高可用和安全管理，是一个轻量级全方位的数据访问平台。

1.1 Heimdall 的名称

Heimdall 来自于北欧神话中的神，他具有三种神力：能明察秋毫的视力与听力，引领众神的号角，以及战无不胜的长剑。这三种神力可以用来比喻 HDAP 的特点：全方位的监控、及时的预警、高效地解决问题。

企业 IT 系统管理最重要的目标是可用性和灵活性。当前的趋势是解决以下两类问题：1) 管理的复杂度；2) 企业应用的性能瓶颈。这两类问题严重影响了企业应用的开发进度，增加了劳动成本，并有可能影响到企业的利润。

Heimdall Data 的目标是为企业应用和数据库管理人员提供简单易用的操作方式：1) 发现系统性能瓶颈；2) 无需修改系统代码的前提下，将系统性能提升 10 倍以上。

无论是初创公司还是成熟企业使用都适合使



Heimdall 数据访问平台 的安装与设置



用 HDAP。超高的性价比，高效的解决方案，安全和稳定的性能是产品的突出特点。HDAP 基于模块化架构，可根据实际需求选取和扩展所需功能。更具实际价值的是 HDAP 可以兼容绝大多数主流数据库产品，并可在 5 分钟内完成安装和部署。HDAP 是解决企业应用性能优化问题的利器。简单、准确、高效。

1.2 主要功能

HDAP 的功能包括三个主要方面：数据库高可用(HA)、数据库访问性能、以及数据安全。



数据库高可用

- 实现应用可感知的数据库故障转移 (application-aware database failover)
- 实现 0 停机维护 (例如频繁地进行版本迭代更新)

数据库性能优化

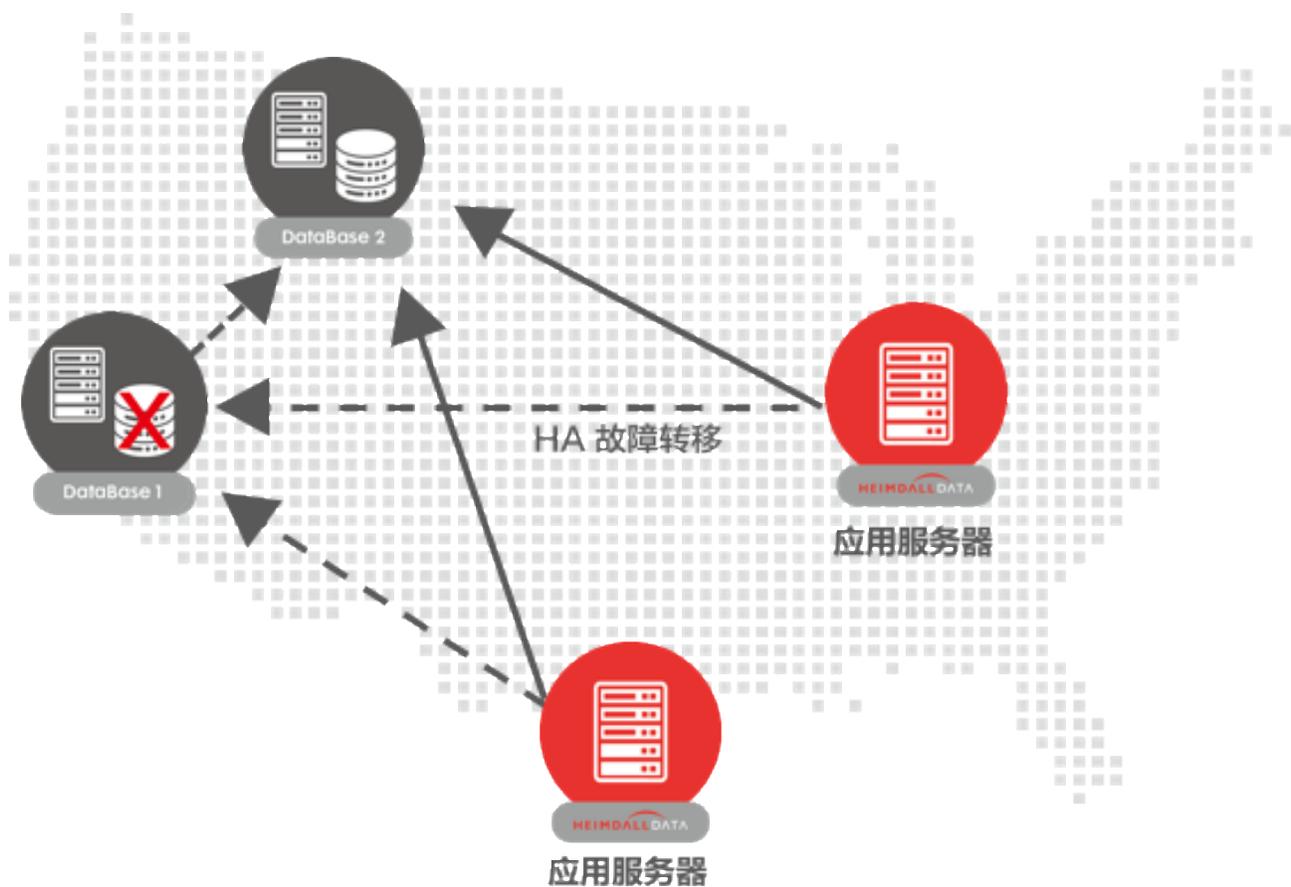
- 降低慢 SQL 带来的性能影响
- 减少应用的网络延迟

Heimdall 数据访问平台 的安装与设置

数据库安全

- 防御一般性的数据攻击（例如 SQL 注入）
- 强制数据库执行严格的管理访问
- 提供可追溯的数据访问管理

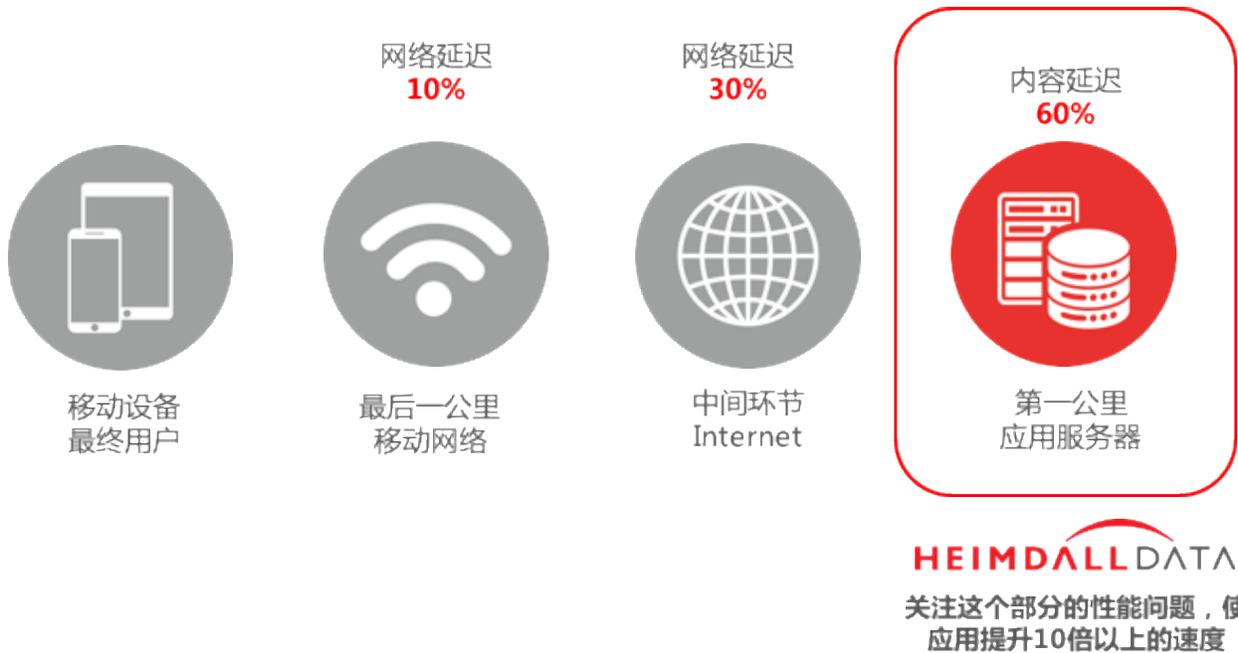
Heimdall 提供了一套开箱即用的方案，即能帮助企业快速实现 HA，还能支持各种数据库产品，彻底实现了 HA 方案与底层数据库产品的分离，能够为企业节省大量的产品购买成本和系统维护成本。



Heimdall 数据访问平台 的安装与设置

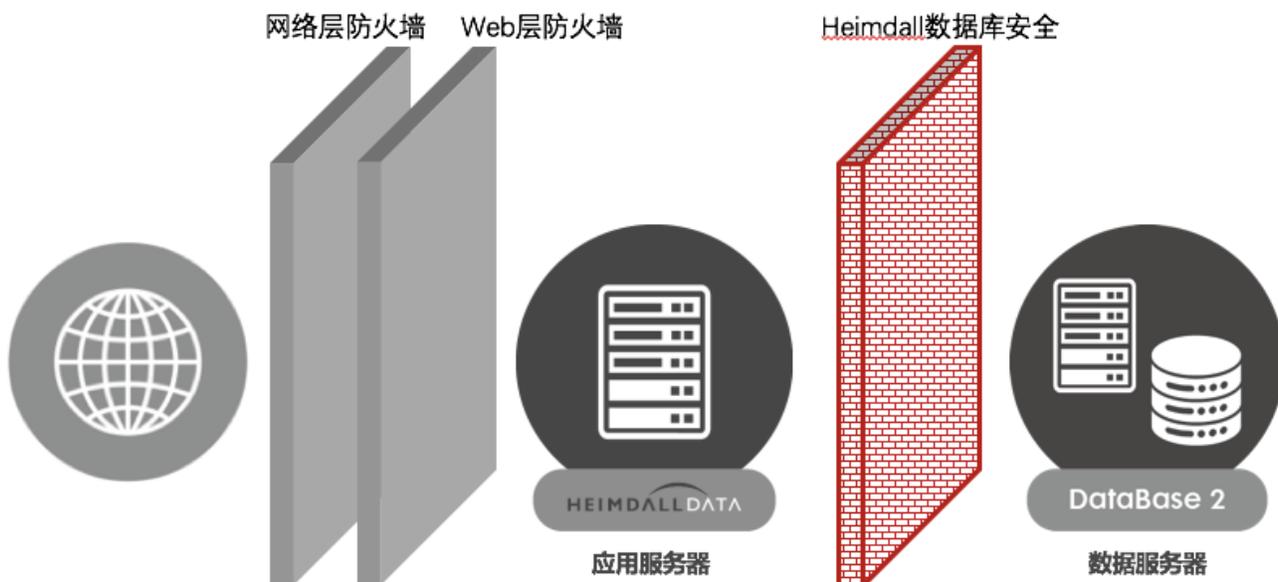


Heimdall 关注数据库与应用之间的数据传输上。在移动应用以及云端应用场景下，这部分的延迟甚至能够占据总体网络延迟的 60%左右（具体详情请访问 <http://www.heimdalldata.cn>，技术白皮书）。Heimdall 通过智能缓存的方式帮助应用将数据预先装在到应用端，极大地节省了应用到数据库之间的网络延迟，使应用提升 10 倍以上的速度。



安全性是所有数据库系统都需要面对的重要问题。围绕网络层、web 层、应用层等解决方案能够帮助企业防御一般性的攻击行为，如 SQL 注入、暴力破解密码等，但是很难防御从内部发起的攻击（如感染员工机器从内部越权读取数据，执行破坏性操作等）。HDAP 的数据分析引擎能够识别出恶意用户访问数据库的行为模式，并根据行为模式采取适当的防御手段。

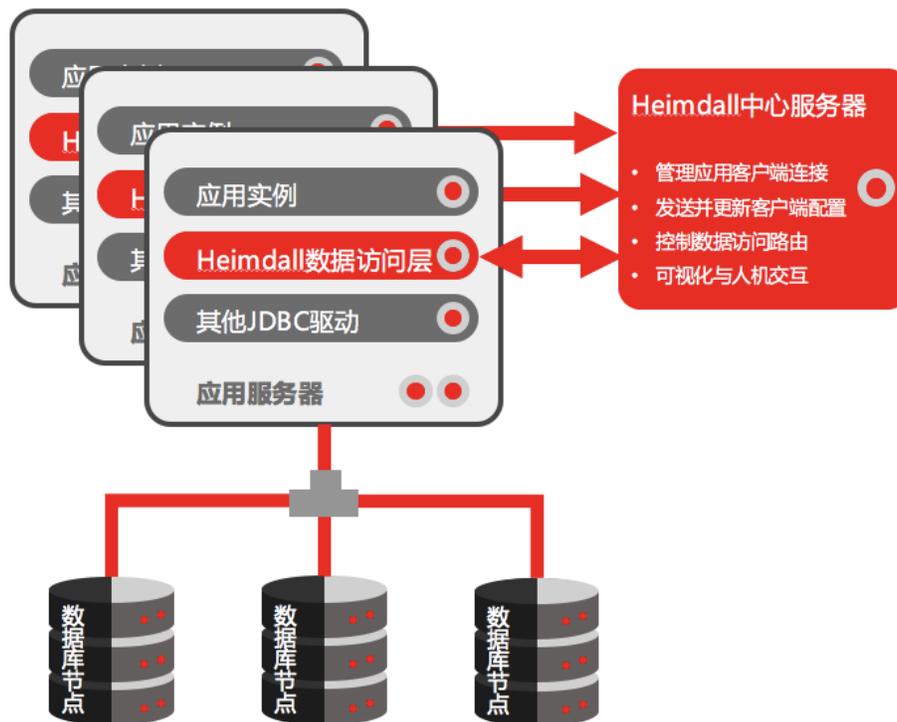
Heimdall 数据访问平台 的安装与设置



1.3 工作原理

Heimdall 数据访问平台包含两个部分：HDAP 的 JDBC driver 和 HDAP 中心服务器。

HDAP driver 安装在应用服务器上，利用应用服务器的计算资源实现分布式缓存、HA 与故障恢复逻辑、与中心服务器通信等功能。中心服务器可安装在任何能够与 HDAP driver 通信的主机上，使用 TCP 协议和 8087 端口进行通信。



从不同的角度来看，HDAP driver 既可以被归为第三类 driver（中间件），又可以被归为第四类 driver（原生 Java）。关于 JDBC driver 分类请参考 <http://www.jdbc-tutorial.com/jdbc-driver-types.html>。

与大部分第三类 driver 直接将数据请求进行转换或者传递到下一个设备的工作方式不同，HDAP driver 利用已有的 JDBC driver 来执行数据库访问，并且 HDAP 在相同的系统上执行时均能提供一致的可扩展性。

HDAP 中心服务器组件为应用服务器提供了一个简单易用的操作面板，用户可执行配置，管理，数据记录，故障转移逻辑设置等操作。

1.4 功能列表

- 所有功能均独立于数据库厂商和协议

Heimdall 数据访问平台 的安装与设置



- 支持多 JVM，并可以与已有的各主流分布式缓存系统集成（支持 Memcached, Redis, EHCache, Hazelcast），默认使用 Hazelcast
- 支持正则表达式和数据表级的缓存及简便灵活的配置
- HA 和故障转移策略配置，支持多种拓扑结构（active, standby, read-only 等）
- 支持读写分离的数据库
- 基于表达式的优化和分片实现了根据策略重定向的能力
- 为未使用连接池或连接池效率不高的应用提供高效连接池
- 自动将低效写入语句进行变换，提高效率
- 应用维护时可直接修改数据源配置而无需重启
- 可以选择性地记录下 SQL 语句和 JDBC 方法并用于性能分析和故障分析
- 基于最优缓存策略的 SQL 语句效率分析和一键缓存
- 用户访问控制，支持基于子网和主机的配置

2 安装

2.1 系统需求

HDAP 服务器组件需要安装 JRE 或者 JDK1.6 或者更高版本，以及 1GB-2GB 的空闲系统内存。HDAP driver 所需的缓存可自行配置，另外还需要约 200MB 的堆空间。因为安装过程可能会修改系统配置，强烈推荐在安装过程中关闭杀毒软件。特别是 Windows 系统下安装的过程中，杀毒软件和系统安全软件会阻止 HDAP 获取管理员权限，或者阻止 HDAP 与数据库的连接。

2.2 免费版与付费版

HDAP 的免费试用版包含了所有的功能但有如下使用限制：

- 中心服务器只能同时管理 2 个客户端 (driver)
- 只能支持 2 个数据库服务器
- 30 天的使用期限

如果您想升级试用版的 HDAP，请发邮件至 support@heimdalldata.com

2.3 简明安装过程

- 1 拷贝 heimdalldriver.jar 和 hazelcast-3.6.jar 到应用 class path
- 2 在 HDAP 控制台中的 Data Source 中配置 JDBC 路径、HA 节点
- 3 配置 Virtual Database , 得到 HDAP 参数
- 4 将 HDAP 参数复制到应用 JDBC 配置文件中
- 5 重启应用
- 6 通过 Analytics 分析数据库访问的动态情况
- 7 在 Rules 页面中定义 HA 规则、缓存规则、安全规则等
- 8 在 Dashboard 中查看性能的变化

2.4 Windows 版本安装

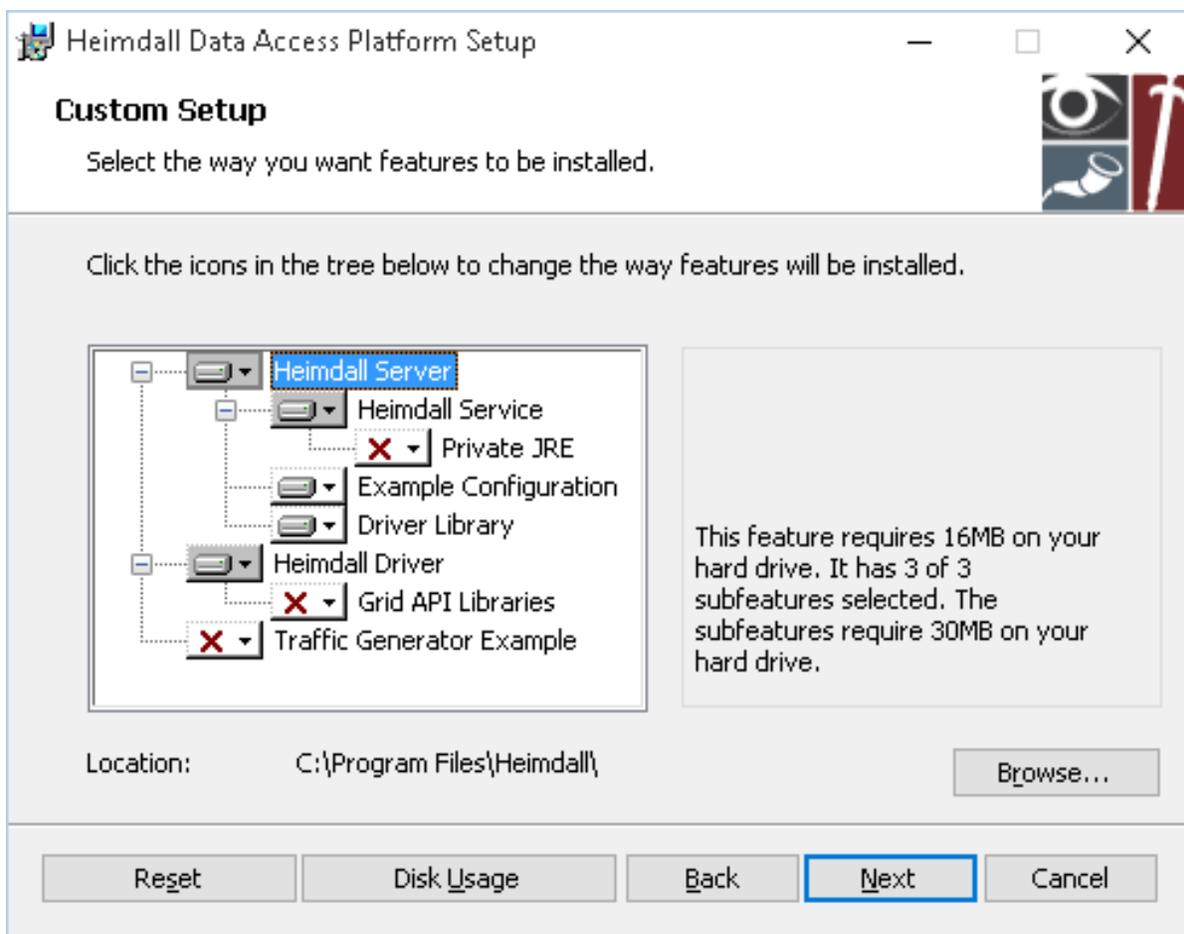
HDAP 可以在 5 分钟内完成安装 , 测试和配置。以下是 Windows 下 Heimdall 的安装步骤 :

- 1) 下载。首先下载 HDAP Windows 安装包 (仅支持 64 位操作系统) 。 <http://www.heimdalldata.cn/downloads>

Heimdall 数据访问平台 的安装与设置



- 2) 安装。HDAP 安装包的必选组件包括 HDAP 服务器和 HDAP driver。可选的组件包括私有 JRE，支持外接分布式缓存的 grid API 所需的库文件，以及一个简单的 demo 程序，名为 Traffic Generator。



其中每个选项的描述如下：

- **Heimdall Server**

HDAP 的中心服务器组件，用于管理各个 HDAP driver。如果你的系统是分布式的，它可以安装在任意一个主机上，即可以通过浏览器从其他主机访问。如果你的系统是单主机的，那么它就可以安装在此主机上。

- **Heimdall Service**

如果你希望将 Heimdall Server 注册为 Windows 系统的一个服务，可以勾选此选项。安装

Heimdall 数据访问平台 的安装与设置



成功之后 Heimdall Server 将会在 Windows 启动之后自动启动。

- Private JRE

如果你没有可用的系统级 JRE 时，你可以选择安装一个私有化的 JRE。安装程序会自动检测你的系统环境变量，如果没有发现可用的 JRE，将会自动勾选这个选项。如果你没有管理员权限，无法安装系统级 JRE，你也可以选择这个选项，反之，你可以先安装最新版的 JRE 或者 JDK 之后再安装 HDAP。

- Example Configuration

安装成功之后，Heimdall Server 组件将会读取一组预定义好的配置文件。对于新用户来说，我们建议勾选此选项。这些配置均可在安装之后随时修改。

- Driver Library

JDBC driver 库包括了各主流数据库官方发布的 JDBC driver。可以随 Example Configuration 一起安装，也可以独立安装。我们推荐勾选此选项，因为它能够帮助用户简化数据访问操作的过程。安装完成之后也可以根据需求随时增加或者删除。

- Heimdall Driver

如果你的这台主机同时也作为应用服务器，那就应该选择安装。Heimdall Driver 安装路径可以按需修改。

- Grid API Libraries

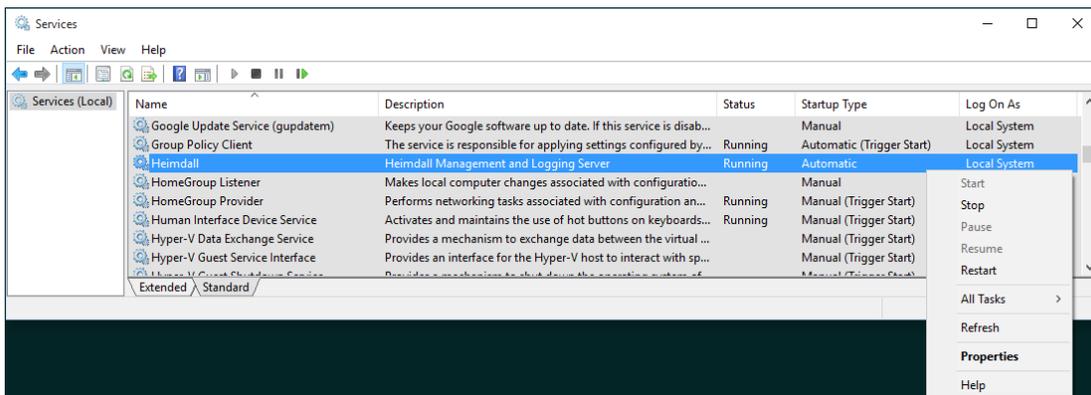
安装完成之后，会在 Heimdall Driver 的安装目录下生成一个 lib 文件夹，里面放置了 grid API 库，用于支持外接的分布式缓存（Memcached, Redis, EHcache, Hazelcast）。通常情况下建议安装。

- Traffic Generator Example

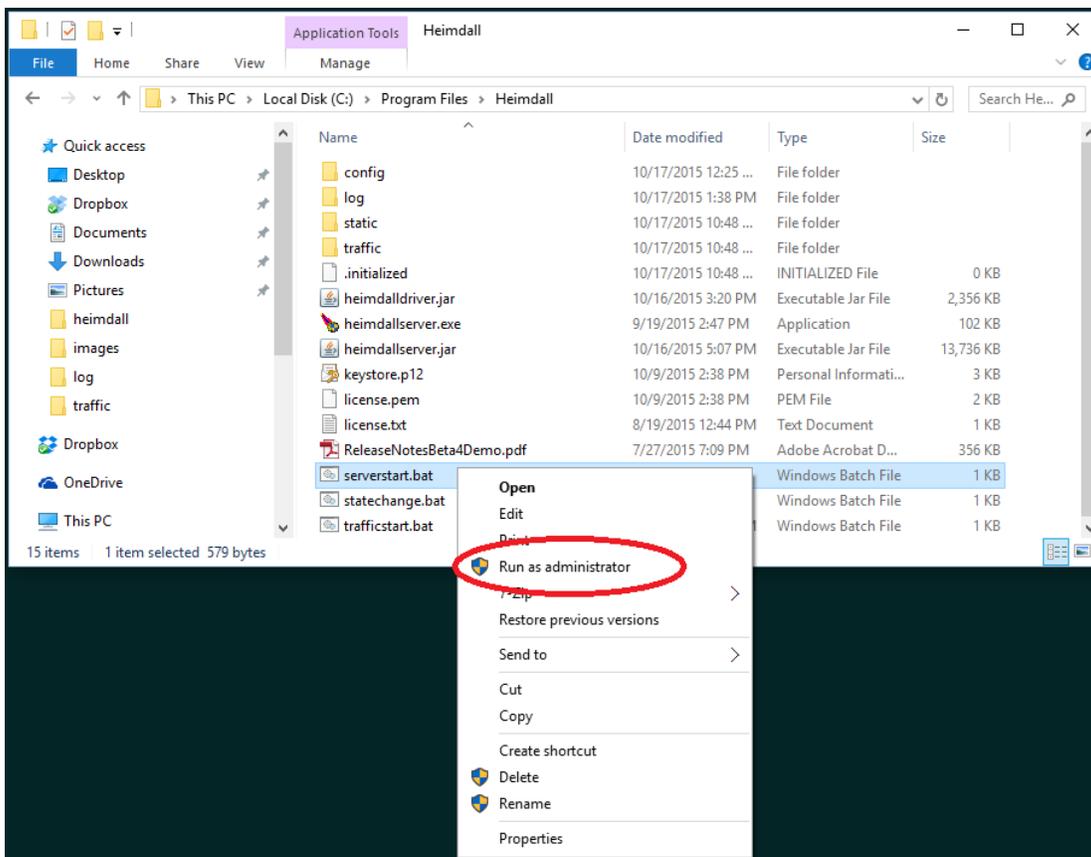
这是一个测试应用程序。它不断地向远程数据库发起访问请求，并通过控制台打印输出。默认情况下的安装位置在 Heimdall 安装目录下的 traffic 目录中。运行 traffic 需要 JDK1.6 及其以上版本。关于 Traffic Generator 的详细情况请参见第 3 章

3) Heimdall 服务启动。如果安装过程中选择了 Heimdall Service 选项，Heimdall 将被注册为一个 Windows 服务随系统自动启动。要重启或者停止 Heimdall 服务，可以使用 Service Manager。

Heimdall 数据访问平台 的安装与设置



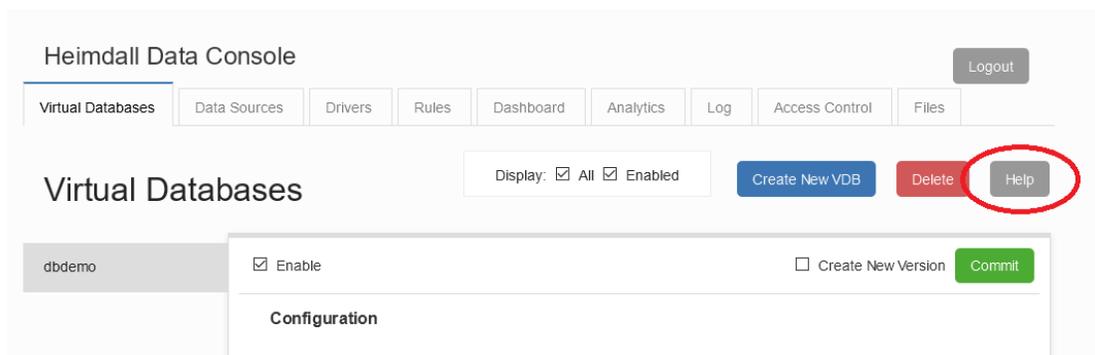
- 4) 手动启动。如果安装过程中没有选择 Heimdall Service，你需要手动启动 Heimdall 服务器。在 Heimdall 安装目录下（默认是 C:\Program Files\Heimdall）有一个“serverstart.bat”的批处理文件。请使用管理员权限来运行（右键单击，选择“用管理员权限运行”）。



- 5) 访问中心服务器管理控制台。你可以使用浏览器访问 <http://localhost:8087> 直接从本

机访问控制台，也可以通过 <http://yourdomain-or-yourip:8087> 来远程访问。

- 6) 获取帮助。管理控制台的每个页面上都有一个 Help 按钮，你可以随时访问在线文档以获取帮助。



- 7) 卸载。Heimdall 可以使用 Windows 标准的 MSI 安装包方式进行安装和卸载。只需要再次运行安装包，选择“卸载”即可。也可以通过 Windows 控制面板的“添加或删除程序”来进行卸载。

2.5 非 Windows 版本安装

Heimdall 支持所有可以运行 JVM 环境的操作系统，兼容 JRE 1.6 及以上版本。安装步骤如下：

- 1) 下载。首先访问以下 URL 来下载独立安装包：<http://www.heimdalldata.cn/downloads>
- 2) 安装。HDAP 安装包包含 HDAP driver、中心服务器，外接分布式缓存 Grid API 支持库，以及一个 Traffic Generator 示例程序。直接将安装包解压到所需的目录下即可。
- 3) 服务启动。因为不同的操作系统启动条件不同，所以我们提供了一个简单的启动脚本文件 `heimdallserver.sh`，你可以在你自己的启动脚本中（例如 `rc.local`）调用它来实现 Heimdall 服务的自动启动。Heimdall 的启动脚本 `heimdallserver.sh` 需要接收一个命令参数来启动或停止 Heimdall 服务。启动命令“`./heimdallserver.sh start`”，停止命令“`./heimdallserve`

r.sh stop”。

2.6 修改默认端口

Heimdall 服务器默认的端口是 8087。如果要修改为其他的端口，只需要在安装目录下创建一个名为“application.properties”的文件，并添加以下内容：

```
Server.port = X
```

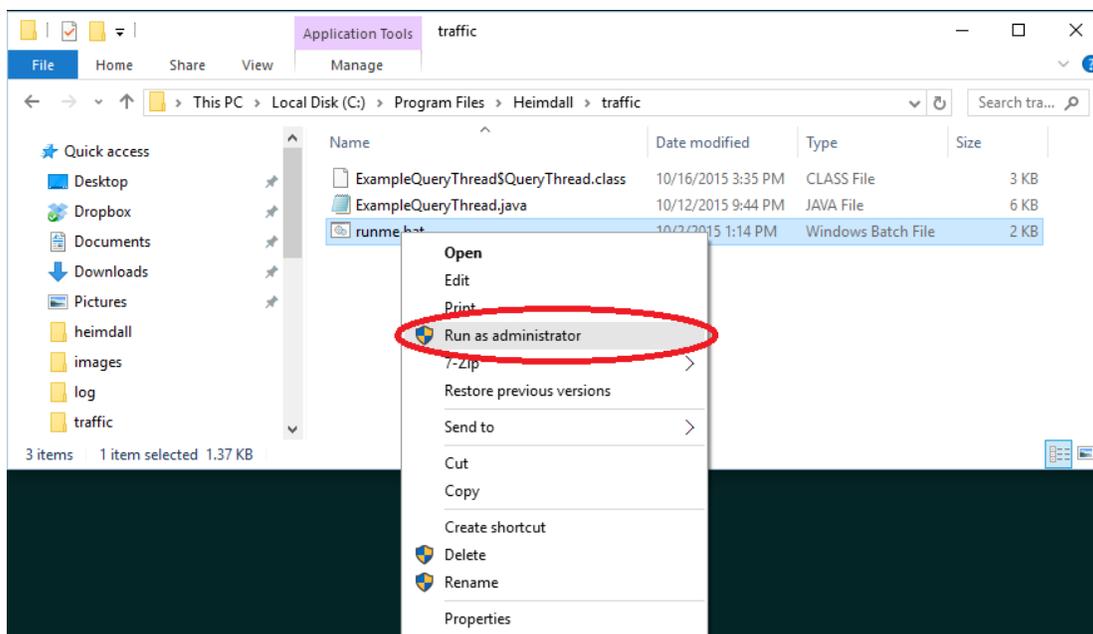
(这里 x 即使你所希望使用的端口号)

3 使用示例程序 Traffic Generator

3.1 下启动 Traffic Generator

1) Windows 下启动

示例程序 Traffic Generator 的默认安装路径 C:\Program Files\Heimdall\traffic，目录下有一个 runme.bat 批处理文件，使用管理员身份运行即可启动示例程序 Traffic Generator。初次启动时，将会调用本地的 javac 来编译源代码。安装包里只提供了 JRE，并没有 JDK。所以如果系统中没有 JDK，则需要到 Oracle 的官网上下载。



2) Linux/Unix 下启动

Heimdall 数据访问平台 的安装与设置



在非 Windows 系统中启动 Traffic Generator，可运行 traffic 目录下的 runme.sh。初次启动时，将会调用本地的 javac 来编译源代码。安装包里只提供了 JRE，并没有 JDK。所以如果系统中没有 JDK，则需要到 Oracle 的官网上下载。

运行 runme.sh 后控制台会输出 Traffic 打印出的信息。正常情况下会显示程序的编译进度、连接进度，最后输出数据库查询的情况。

```
四月 23, 2016 11:41:02 下午 com.hazelcast.cluster.impl.MulticastJoiner
信息: [10.0.0.6]:5701 [dev] [3.6]

Members [1] {
  Member [10.0.0.6]:5701 this
}

四月 23, 2016 11:41:02 下午 com.hazelcast.core.LifecycleService
信息: [10.0.0.6]:5701 [dev] [3.6] Address[10.0.0.6]:5701 is STARTED
Config{groupConfig=GroupConfig [name=dev, password=*****], properties={}, networkConfig=NetworkConfig{publicAddress='null', port=5701, portCount=100, portAutoIncrement=true, join=JoinConfig{multicastConfig=MulticastConfig [enabled=true, multicastGroup=224.2.2.3, multicastPort=54327, multicastTimeToLive=32, multicastTimeoutSeconds=2, trustedInterfaces=[], loopbackModeEnabled=false], tcpIpConfig=TcpIpConfig [enabled=false, connectionTimeoutSeconds=5, members=[], requiredMember=null], awsConfig=AwsConfig{enabled=false, region='us-east-1', securityGroupName='null', tagKey='null', tagValue='null', hostHeader='ec2.amazonaws.com', iamRole='null', connectionTimeoutSeconds=5}, discoveryProvidersConfig=com.hazelcast.config.DiscoveryConfig@69721379}, interfaces=InterfacesConfig{enabled=false, interfaces=[]}, sslConfig=null, socketInterceptorConfig=null, symmetricEncryptionConfig=null}, mapConfigs={heimdall=MapConfig{name='heimdall', inMemoryFormat=BINARY, backupCount=1, asyncBackupCount=0, timeToLiveSeconds=0, maxIdleSeconds=0, evictionPolicy='NONE', evictionPercentage=25, minEvictionCheckMillis=100, maxSizeConfig=MaxSizeConfig{maxSizePolicy='USED_NATIVE_MEMORY_SIZE', size=476}, readBackupData=false, hotRestart=HotRestartConfig{enabled=false, fsync=false}, nearCacheConfig=null, mapStoreConfig=MapStoreConfig{enabled=false, className='null', factoryClassName='null', writeDelaySeconds=0, writeBatchSize=1, implementation=null, factoryImplementation=null, properties={}, readOnly=null, initialLoadMode=LAZY, writeCoalescing=true}, mergePolicyConfig='com.hazelcast.map.merge.PutIfAbsentMapMergePolicy', wanReplicationRef=null, entryListenerConfigs=[], mapIndexConfigs=[], mapAttributeConfigs=[], quorumName=null, queryCacheConfigs=[], cacheDeserializedValues=INDEX_ONLY}, topicConfigs={}, reliableTopicConfigs={}, queueConfigs={}, multiMapConfigs={}, executorConfigs={default=ExecutorConfig{name='default', poolSize=16, queueCapacity=2147483647}}, semaphoreConfigs={}, ringbufferConfigs={}, wanReplicationConfigs={}, listenerConfigs=[], partitionGroupConfig=PartitionGroupConfig{enabled=false, groupType=PER_MEMBER, memberGroupConfigs=[]}, managementCenterConfig=ManagementCenterConfig{enabled=false, url='null', updateInterval=3}, securityConfig=SecurityConfig{enabled=false, memberCredentialsConfig=CredentialsFactoryConfig{className='null', implementation=null, properties={}}, memberLoginModuleConfigs=[], clientLoginModuleConfigs=[], clientPolicyConfig=PermissionPolicyConfig{className='null', implementation=null, properties={}}, clientPermissionConfigs=[]}, liteMember=false}
ClusterService{address=Address[10.0.0.6]:5701}
四月 23, 2016 11:41:02 下午 com.hazelcast.partition.InternalPartitionService
信息: [10.0.0.6]:5701 [dev] [3.6] Initializing cluster partition table arrangement...
Queries: 1000 failures: 0
Queries: 2000 failures: 0
Queries: 3000 failures: 0
Queries: 4000 failures: 0
Queries: 5000 failures: 0
Queries: 6000 failures: 0
Queries: 7000 failures: 0
Queries: 8000 failures: 0
Queries: 9000 failures: 0
Queries: 10000 failures: 0
```

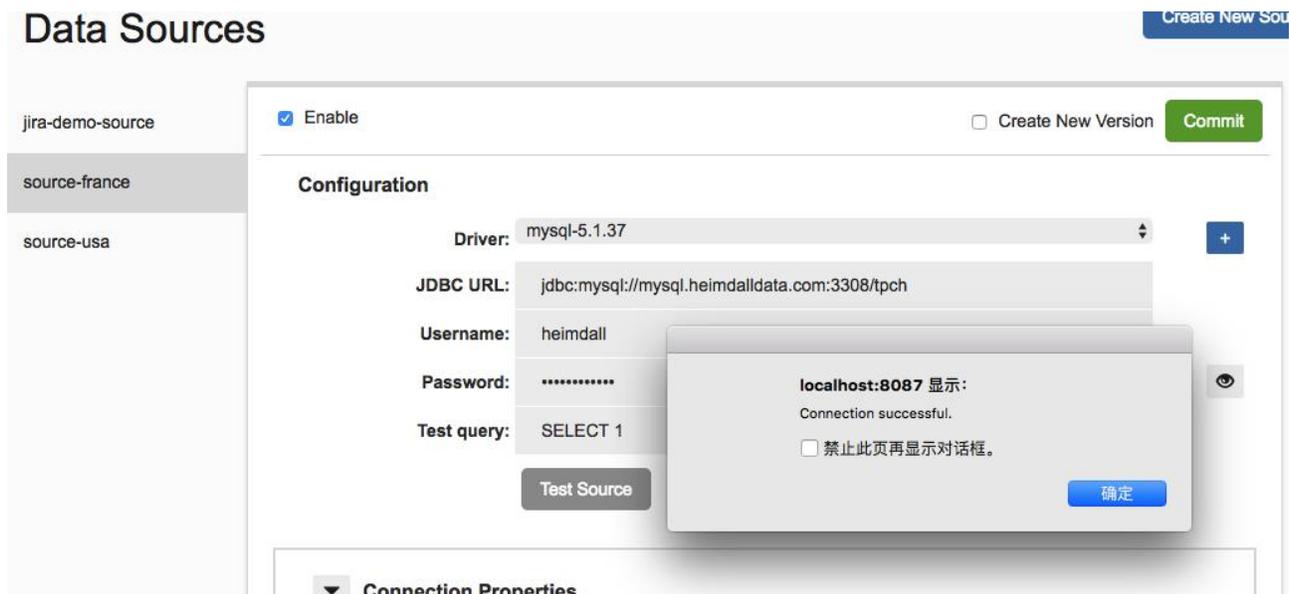
3.2 测试数据库连接

默认情况下 HDAP 已经为 Traffic Generator 配置好了参数，可以直接进行数据库连接测试。Traffic Generator 连接的是 Heimdall 公司的数据库服务器，因此要确保测试用的主机是连接在

Heimdall 数据访问平台 的安装与设置



互联网上的。通过 <http://localhost:8087> 访问 HDAP 控制台，首先进入 Data Source 页面，有三个默认的数据源。其中 source-france 和 source-usa 是 Traffic 的两个分别位于法国和美国的数据源。其中 source-usa 数据源配置了两个数据节点，互为主备，是 Active-Active 的结构。选择数据源后点击 Test Source 按钮，查看是否数据库连接成功。如果成功，会弹出消息显示 Connection Successful。



3.3 监视数据库访问动态

切换到 Dashboard 页面，这里有 9 个动态曲线，分别从不同的角度展示当前数据库的访问情况和缓存的命中情况。包括数据库节点是否正常、Active-Active 双主机各自分配的数据流量、每个数据库的访问性能等。正常情况下应该能看到所有曲线都出现动态的波动，且缓存命中率均值在 40%-80%之间（具体数据依赖于主机的内存、网络速度等因素）。

Heimdall 数据访问平台 的安装与设置



图中 9 个图形的含义参见第 4 章。

4 JVM 应用接入 Heimdall 平台

应用只需要通过简单的几步操作即可与 Heimdall 平台集成。无需代码修改，也无需重新部署整个应用。但是集成的前提是首先要确认应用处于正常的状态下，可以无错误地启动。另外安装过程需要重启应用，因此对于关键性业务系统，需要谨慎地选择重启时机。

4.1 简明安装步骤

- 1 拷贝 heimdalldriver.jar 和 hazelcast-3.6.jar 到应用 class path
- 2 在 HDAP 控制台中的 Data Source 中配置 JDBC 路径、HA 节点
- 3 配置 Virtual Database，得到 HDAP 参数
- 4 将 HDAP 参数复制到应用 JDBC 配置文件中
- 5 重启应用
- 6 通过 Analytics 分析数据库访问的动态情况
- 7 在 Rules 页面中定义 HA 规则、缓存规则、安全规则等
- 8 在 Dashboard 中查看性能的变化

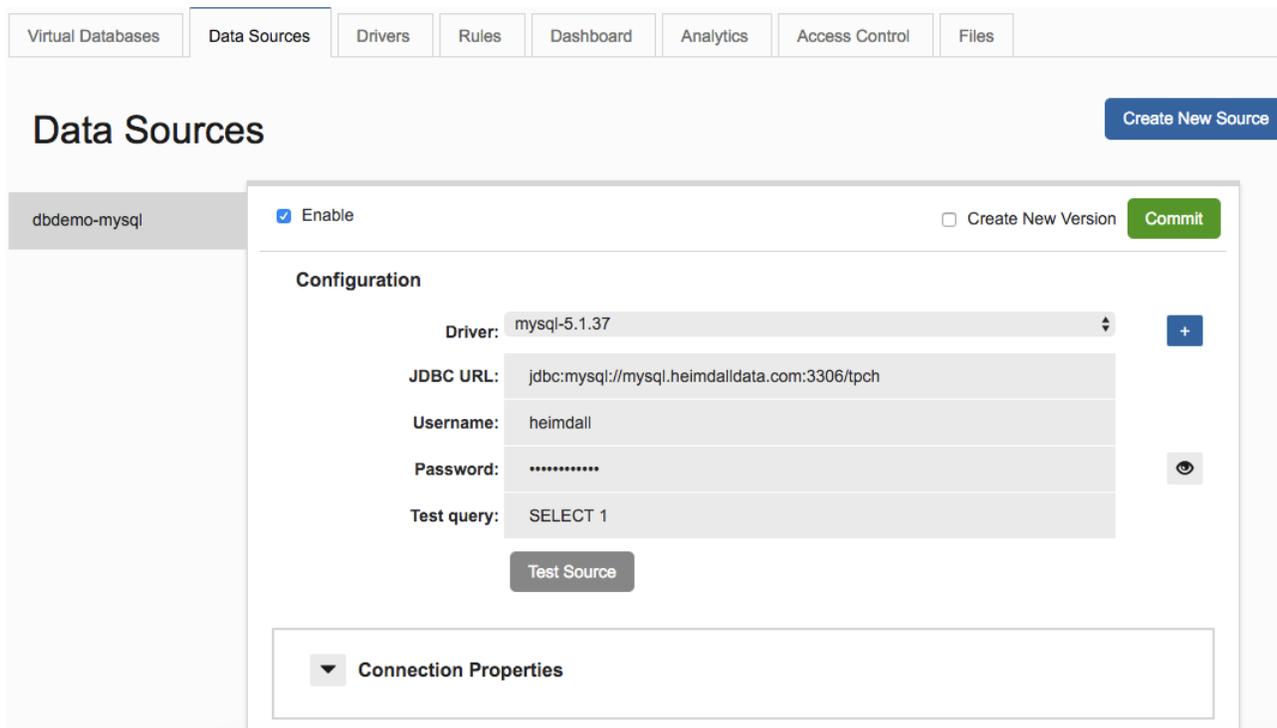
4.2 安装 heimdalldriver.jar

首先要安装 HDAP driver。它的文件名是 heimdalldriver.jar。从安装包中找到这个文件，并将其复制到应用的 class path 中，即完成了安装过程。如果你不确定应用的 class path，可以先找到应用正在使用的 JDBC driver 目录，再将 heimdalldriver.jar 复制到这个目录中即可。

4.3 安装 hazelcast-3.6.jar

默认情况下 Heimdall 使用的 hazelcast 作为缓存，因此需要将安装包里的 hazelcast-3.6.jar 复制到应用的 class path。Hazelcast-3.6.jar 在 /heimdall/libs/ 目录下。

4.4 创建 Data Source , 配置 JDBC



从浏览器访问 <http://localhost:8087> 进入 HDAP 控制台，进入 Data Source 页面。首先通过 Create New Source 按钮创建一个新的 Data Source。输入名称后在 Drive 栏选择你的应用正在使用的 JDBC driver。列表里列出了大多数常用的 JDBC driver，如果你使用的 driver 不在其中，可以切换到 Drivers 页面中上传一个。如下图。

Heimdall 数据访问平台 的安装与设置



Virtual Databases | Data Sources | Drivers | Rules | Dashboard | Analytics | Access Control | Files

Driver Management

[Add New Driver](#)

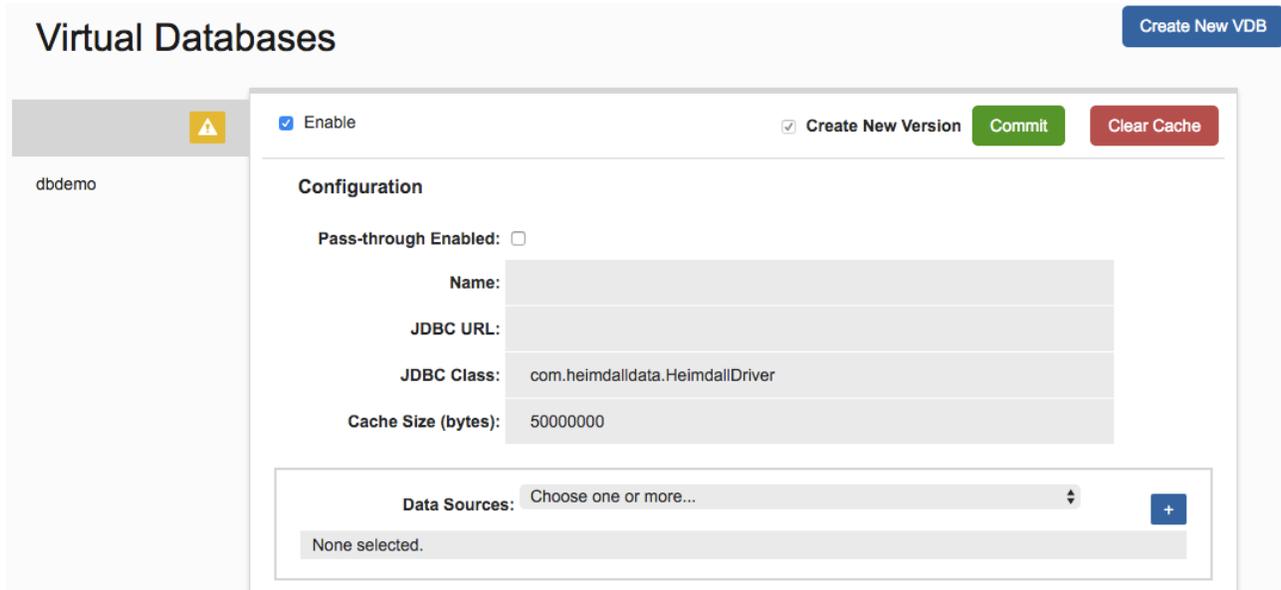
<input checked="" type="checkbox"/> Enable	<input type="checkbox"/> Create New Version	Commit
Configuration		
Version:	5.1.37	
Example Uri:	jdbc:mysql://<hostname>[<failoverhost>][<:3306>]/<dbname>[?<param1>=	
Website Uri:	http://dev.mysql.com	
JDBC Class:	com.mysql.jdbc.Driver	
Data Source Class:	com.mysql.jdbc.jdbc2.optional.MysqlDataSource	
XA Data Source Class:	com.mysql.jdbc.jdbc2.optional.MysqlXADataSource	
Notes:	Java 6+	
Driver Upload:	选择文件 未选择任何文件	
Current files:	mysql-5.1.37.jar ×	

选择正确的 JDBC driver 后，在 JDBC URL 中输入应用中正在使用的 JDBC 配置，然后输入数据库的用户名和密码，最后可以通过 Test Source 按钮测试数据库是否联通。如果测试失败，可能是由于以下原因：

- 当前主机无法连接数据库
- 当前主机的 IP 被数据库拒绝
- JRE 或者 JDK 的版本过低
- 用户名和密码错误
- JDBC driver 版本不匹配

Data Source 创建完成之后点击 Commit 按钮保存。

4.5 创建 Virtual Databases , 计算 HDAP 参数



切换到 Virtual Database 页面，点击 Create New DB 按钮创建一个新的 Virtual Database。

- 1) 输入名称；
- 2) JDBC URL、JDBC Class 框不能输入，由系统自动计算出来；
- 3) Cache Size 栏默认是 50M，可以调整为期望的数值；
- 4) 选择 Data Source。从列表中选择上一个步骤中创建的 Data Source；
- 5) 点击 Commit，JDBC URL 和 JDBC Class 将会自动计算出来。

4.6 将 HDAP 参数填写到应用配置文件中

这个步骤的目的是将应用先有的 JDBC 路径替换为 HDAP 计算出来的 JDBC 路径。举例说明：

Heimdall 数据访问平台 的安装与设置



例如在 Atlassian Bitbucket 服务器中，配置文件的路径一般是：`/var/atlassian/application-data/bitbucket/shared/bitbucket.properties`：

其中关于 JDBC 的配置部分为：

```
jdbc.driver=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/bitbucket?
characterEncoding=utf8&useUnicode=true&sessionVariables=storage_engine %3DInnoDB
jdbc.user=user
jdbc.password=password
```

这里就应该将 `jdbc.driver` 和 `jdbc.url` 替换成 HDAP 控制台上 Virtual Database 自动计算出来的值。URL 通常会以固定的 `jdbc:heimdall://` 开头，最后的参数是 Virtual Database 的名称。此例中是 `dbdemo`。

*注意：HDAP 计算出来的 URL 只是一个参考形式，最终的 URL 还应该考虑实际情况。如果应用服务器使用的是公网 IP（如 111.222.3.4），而数据库服务器使用的内网 IP（如 192.168.1.2），在 HDAP 控制台上计算出来的 URL 会使用公网 IP（`jdbc:heimdall://111.222.3.4:8087/dbdemo`）。此时配置文件中就不应该使用公网 IP，而使用内网 IP（`jdbc:heimdall://192.168.1.2:8087/dbdemo`）。如下所示：

```
#jdbc.driver=com.mysql.jdbc.Driver
jdbc.driver= com.heimdalldata.HeimdallDriver
#jdbc.url=jdbc:mysql://localhost:3306/bitbucket?//原有的这行写到 Datasource 页面的 URL 栏里
jdbc.url= jdbc:heimdall://192.168.1.2:8087/dbdemo
characterEncoding=utf8&useUnicode=true&sessionVariables=storage_engine %3DInnoDB
jdbc.user=user
jdbc.password=password
```

再举个例子：在 Atlassian Jira 的配置中，配置文件是：`/var/atlassian/application-data/jira/db`

Heimdall 数据访问平台 的安装与设置



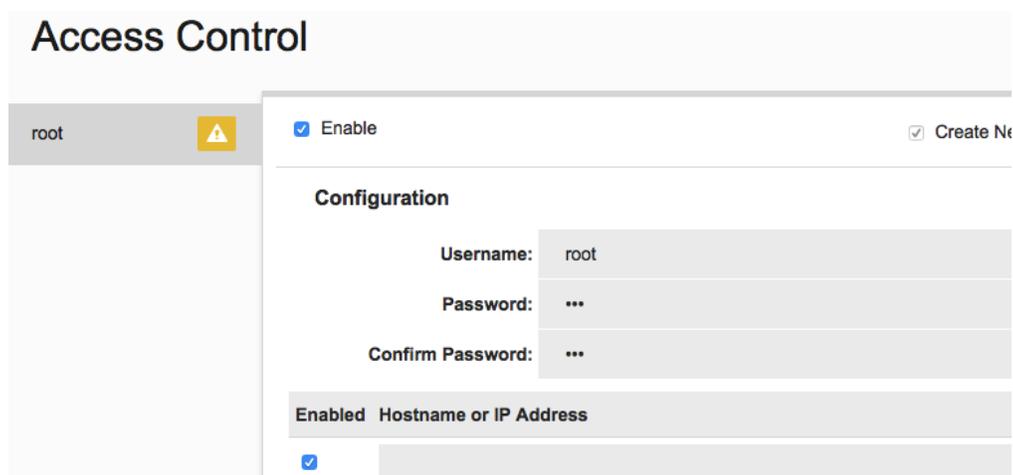
config.xml。其中关于 JDBC 的配置部分为：

```
<url>jdbc:mysql://localhost:3306/jira?useUnicode=true&characterEncoding=UTF8&sessionVariables=storage_engine=InnoDB</url>  
<driver-class>com.mysql.jdbc.Driver</driver-class>  
<username>root</username>  
<password>a1k81ucuTr</password>
```

同样这里将<url>标签内的内容替换为 HDAP 计算出来的 JDBC URL，<driver-class>标签中的内容替换为 JDBC driver 即可。

```
<!-- <url>jdbc:mysql://localhost:3306/jira?useUnicode=true&characterEncoding=UTF8&sessionVariables=storage_engine=InnoDB</url>原有的这行配置写到 Datasource 的 URL 栏里 -->  
<url>jdbc:heidmall://192.168.1.2:8087/dbdemo</url>  
<!-- <driver-class>com.mysql.jdbc.Driver</driver-class> -->  
<driver-class> com.heidmalldata.HeimdallDriver </driver-class>  
<username>root</username>  
<password>a1k81ucuTr</password>
```

*注意：如果在 HDAP 的 Access Control 页面创建了用户账户和密码（例如用户名 root，密码 123）。



则需要将用户名和密码通过 URL 参数的方式加到 JDBC URL 后面。其中用户名参数为 “hduser”，密码参数为 “hdpassword”。上述的例子中就是：

```
jdbc:heimdall://192.168.1.2:8087/dbdemo?hduser=root&hdpassword=123
```

4.7 重启应用

应用重启后配置文件才能生效。重启后应当立即对应用进行功能性测试，以确保配置文件修改正确，未影响其他部分的功能。这一步骤至关重要，有必要慎重周密地测试。应该检查应用输出的 log。如果出现了类似下面的信息，则表示 HDAP driver 已经被正确安装并启动了。

```
[2015-10-20 06:06:02.344] Heimdall: Retrieving updated configuration for URL:  
http://localhost:8087/dbdemo
```

```
[2015-10-20 06:06:02.345] Heimdall: getting config from: http://localhost :8087/api/config/  
vdb/dbdemo
```

```
[2015-10-20 06:06:02.545] Heimdall: heimdall: Startup performed:
```

```
com.heimdalldata.log.EventLogger@7de87192
```

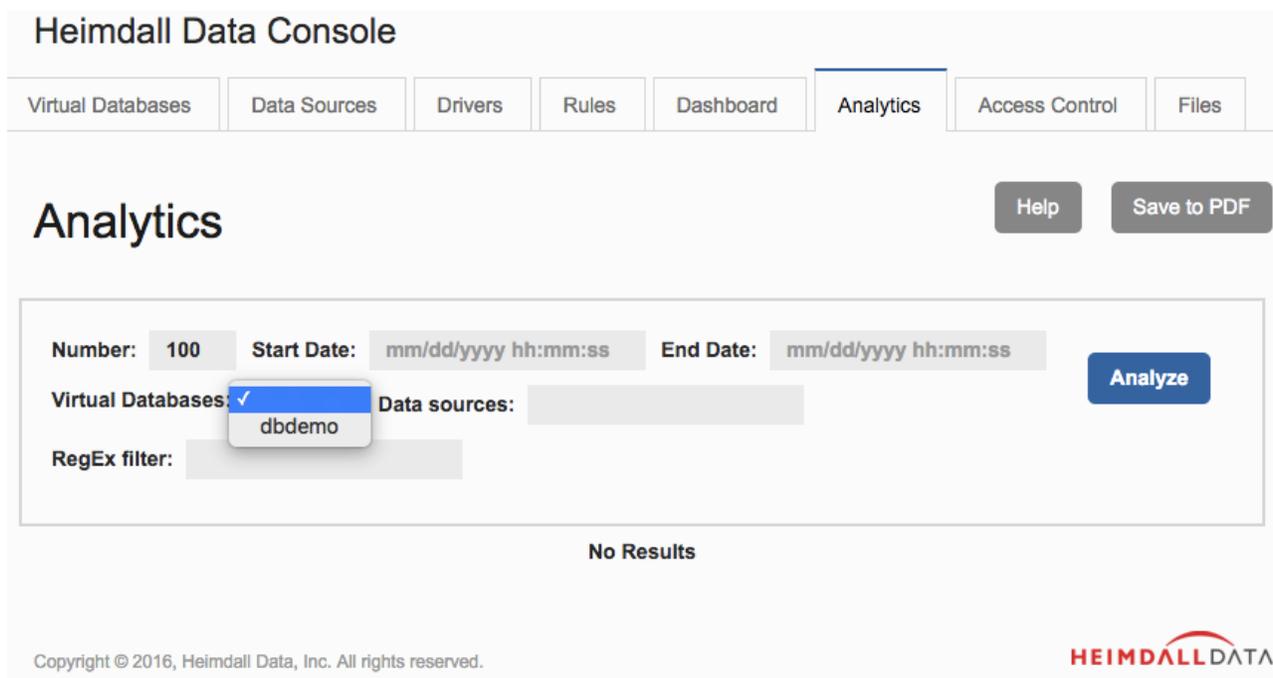
```
[2015-10-20 06:06:02.546] Heimdall: Initializing logger:  
com.heimdalldata.log.EventLogger@7de87192
```

所有由 Heimdall driver 输出的信息都从[YYYY-MM-DD HH:mm:ss.SSS] Heimdall: message 开头。如果在应用的 log 里找不到，有可能是 HDAP driver 没有被应用初始化，或者是应用其他的错误导致。

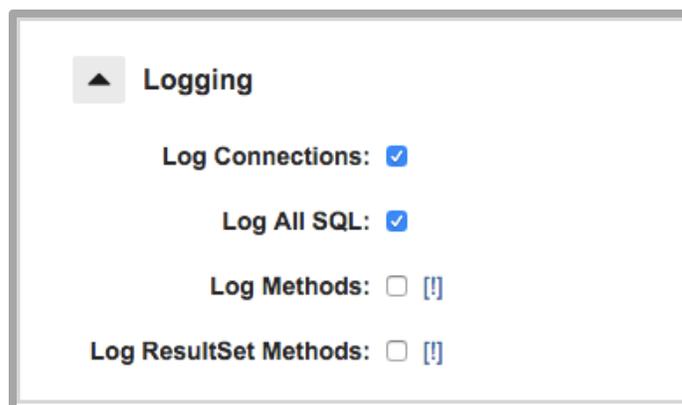
配置常见错误包括：

- HDAP driver 没有放在应用的 class 路径下，应用找不到
- JDBC URL 或者 classname 没有正确配置
- 8087 端口没有开放

4.8 通过 Analytics 分析数据库访问性能



Analytics 页面将显示应用 SQL 执行的性能情况。首先在页面上选择 Virtual Databases 后点击 Analyze 按钮。如果没有出现分析结果，说明分析系统没有获取到足够的数据库。可以通过在 Virtual Database 页面中开启 logging 功能来让分析系统更快地收集到跟多数据。



Heimdall 数据访问平台 的安装与设置



需要注意的是，开启了 logging 功能之后性能会有略微的下降（约 5~10%）。而分析系统所需的 log 只需要覆盖了用户的基本场景之后即可。因此无需长时间地开启 logging 功能。

Rank	One-Click Cache	Query	Server Time (%)	Duplicate Query & Response (%)	Cache Safety Rating	Cache Time (s)	Cache Hit (%)	Count	Query Response Time (µs)	Result Retrieval Time (µs)	Total KBs
1	Caching	SELECT * FROM Part where p_partkey = ?	49	26	0	86400	0.0	4351	444853.5	445055.7	4351
2	Caching	SELECT * FROM Customer where c_custkey = ?	49	26	0	86400	0.0	4355	440225.0	440323.9	4355
3	Caching	SELECT * FROM Nation INNER JOIN Region ON n_regionkey = r_regionkey	0	100	100	86400	99.7	4410	6249.7	6348.1	30870
4	Caching	SELECT * FROM Nation	0	100	100	86400	99.8	8820	1687.0	1729.2	35280
5	Caching	SELECT * FROM Orders where o_orderkey = ?	0	93	10	86400	99.5	4398	2424.4	2437.9	4398
6	Caching	SELECT * FROM Region	0	100	100	86400	99.8	8820	1088.8	1102.6	8820
7	Caching	SELECT avg(o_totalprice) FROM Orders	0	100	90	86400	99.7	4398	1605.0	1615.4	4398

上图是示例程序 Traffic Generator 的分析页面。各列的意义如下：

- 1) Rank：“推荐缓存”的 SQL 排名。HDAP 通过智能算法对监控的 SQL 执行效率进行计算，将最适合缓存的 SQL 排在最前，以此类推。
- 2) One-Click Cache：点击 Cache 按钮可以实现“一键缓存”。点击之后按钮将变成“Caching”，表示此 SQL 正在被系统缓存。
- 3) Query：SQL 的正则表达式。
- 4) Server Time (%)：每条 SQL 发送到数据库服务器的响应时间占所有响应时间的占比。
- 5) Duplicate Query & Response (%)：这条 SQL 执行多次后返回的结果中有多少比例的数

据是重复出现的。

- 6) Cache Safety Rating : 缓存安全性评分。如果 SQL 里不含任何变量, 则是 100%安全的。变量越多, 执行过程中变化越大, 则评分越低。
- 7) Cache Time : 数据缓存生存时间, 以秒为单位。
- 8) Cache Hit (%) : 缓存命中率。
- 9) Count : SQL 执行次数。
- 10) Query Response Time (μ s) : SQL 执行的响应时间。以微秒为单位。
- 11) Result Retrieval Time (μ s) : 结果返回时间。以微秒为单位。
- 12) Total KBs : 每条返回的数据量, 以 KB 为单位。

Rank 的排名根据后面几个数据综合计算得到。通常来说, 越是频繁执行, 且返回的数据集越大的 SQL 排名越靠前。HDAP 推荐用户通过 Cache 按钮自动缓存, 实现一键加速。

4.9 通过设置 Rules 调整缓存策略

在 Analytics 页面中用户可以点击 “Cache” 按钮缓存 SQL。每次点击都会在 Rules 中创建一条缓存策略。

Heimdall 数据访问平台 的安装与设置



Heimdall Data Console

Virtual Databases | Data Sources | Drivers | **Rules** | Dashboard | Analytics | Access Control | Files

Rules (Unlicensed) Create New Rule List Delete Help Save to PDF

dbdemo-rules Enable Create New Version Commit

Enabled	Regex	In-Trans	Action	Parameter	Value	Edit	Delete
<input checked="" type="checkbox"/>	.*	<input checked="" type="checkbox"/>	Log				<input type="checkbox"/>
<input checked="" type="checkbox"/>	(SELECT .* FROM Nation ORDER BY N_NATIONKEY) DESC	<input checked="" type="checkbox"/>	Transform	target	\$1 ASC LIM		<input type="checkbox"/>
<input checked="" type="checkbox"/>	INSERT.*	<input checked="" type="checkbox"/>	Async Execute				<input type="checkbox"/>
<input checked="" type="checkbox"/>	SELECT.*	<input checked="" type="checkbox"/>	Cache	ttd	0		<input type="checkbox"/>
<input checked="" type="checkbox"/>	INSERT.*	<input checked="" type="checkbox"/>	Forward	source	dbdemo2-m		<input type="checkbox"/>

- 1) Enabled : 可以通过 Enabled 复选框来使规则生效或失效。
- 2) Regex : SQL 的正则表达式
- 3) In-Trans : 是否在事务中。如果这条 SQL 不在事务中但勾选了此选项, 则有可能无法正确缓存其数据。反之亦然。
- 4) Action : 规则类型。HDAP 有 6 种 Action , 详见第 7 章 :
- 5) Parameter : 有些 Action 如 Transform 是带参数的。
- 6) Value : 有些 Action 如 Transform 还需要一个值。
- 7) Edit : 可以手工修改此规则
- 8) Delete : 删除规则。勾选此框后, 点击 “Commit” 即可删除此条规则。

4.10 通过 Dashboard 查看性能

完成了关于缓存的规则设置后, 应用的性能会立即得到提升。Dashboard 提供了一个很好的

Heimdall 数据访问平台 的安装与设置



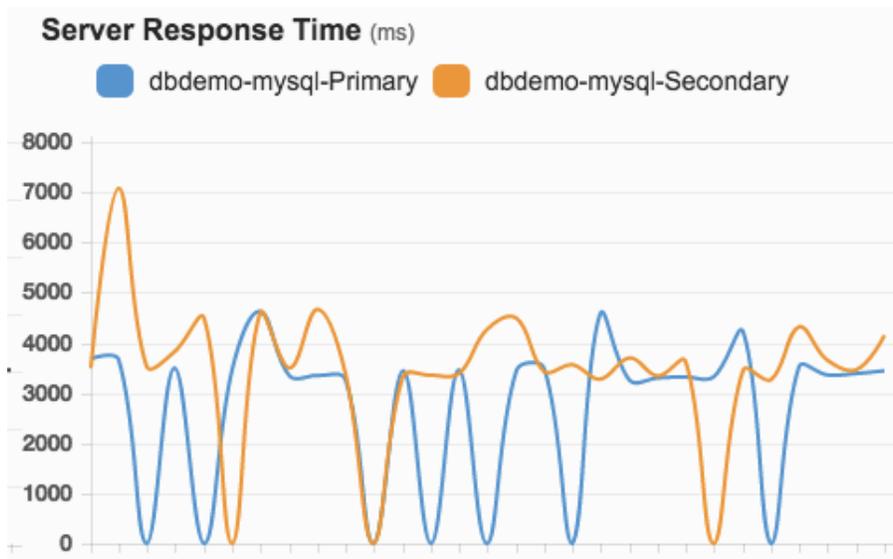
可视化的面板供用户观察应用性能的变化。



Dashboard 有 9 个动态图。分别是：

- 1) Queries Per Second：每秒查询次数。
- 2) Average Query Time：平均查询时间。
- 3) Connections：数据库连接数。
- 4) Cache Hit Rate：缓存命中率。
- 5) Cache Memory Usage：缓存使用情况。

6) Server Response Time : 服务器响应时间。



当数据库启动了 HA 方案，有多个数据库时，图中就会出现多条曲线。每条曲线代表一个数据库的响应时间。后续几个视图均是类似的处理方式。

7) Queries Per Second (Per Server) : 每个服务器每秒查询数量。

8) Average Query Time (Per Server) : 每个服务器的平均查询时间。

9) Connections (Per Server) : 每个服务器的连接数。

5 非 JVM 应用接入 Heimdall 平台

非 JVM 应用不使用 JDBC driver，不能采用第 4 章的安装方式。为此 Heimdall 平台为所有的 MySQL 应用提供了另一种接入方式。（注：目前仅支持 MySQL，其他数据库产品将在未来的版本中陆续提供支持）。下面以 WordPress 为例说明如何接入 Heimdall 平台。

5.1 简明步骤

- 1) 启动 Heimdall Server。使用命令 `sudo ./heimdallserver.sh start`；
- 2) 创建 DataSource。在 Heimdall 控制台中新建一个 data source 直接连接 WordPress 的数据库并进行测试；
- 3) 创建 Virtual Database。在 Heimdall 控制台中新建一个 virtual database，使用新建的 data source；
- 4) 修改 proxy-start.sh 并启动。在文件找到 WordPress 的配置行，填入相应参数；
- 5) 修改 WordPress 数据库连接。修改 wp-config.php 中 DB_HOST，改为 IP+端口 5050。

5.2 启动 Heimdall 服务器

进入 heimdall 文件夹，使用命令 `sudo ./heimdallserver.sh start` 启动 Heimdall Server。如果之前没有安装 JDK，启动过程会失败并给出提示。按照提示安装 JDK 之后再次启动就可以。其他问题都可以在 heimdall/log/heimdallserver.log 文件中进行观察。

5.3 创建 Data Source

Heimdall Server 启动之后通过 <http://<yourdomain>:8087> 访问 web 控制台。进入 Data Sources 页面，创建一个新的数据源。其中参数使用如下：

- 1) Driver：选择 mysql 的 driver；
- 2) JDBC URL：填入 jdbc:mysql://<your_db>:<port>/<db_name>。这里 your_db 是 WordPress 数据库服务器的 IP 地址或者域名，port 是数据库端口，MySQL 默认 3306；db_name 则是 WordPress 的数据库名称。
- 3) User Name：数据库的用户名；
- 4) Password：数据库用户密码；
- 5) Test Query：用于测试的 SQL。默认是 SELECT 1.

配置完成后可以点击 “Test Source” 进行测试。如果连接失败，可通过 `heimdall/log/heimdalls`
`erver.log` 查看出错信息。

5.4 创建 Virtual Database

- 1) 进入 Virtual Database 页面，创建一个新的虚拟数据库；
- 2) JDBC URL 和 JDBC class 两项是由系统自动给出的参考值，无需修改；
- 3) Data Source：选择在上一步骤创建的数据源；
- 4) Rule List：通过 “+” 号跳转至 Rule List 页面新建一个 Rule List，再回到 Virtual Databas

e 页面通过下拉列表选择刚刚新建的 Rule List ;

- 5) Logging : 建议勾选上 Log Connections 和 Log All SQL 两项。运行一段时间后再清空选择 (因为 Log 太多可能会影响系统性能, 初始状态开启 log 更容易调试)。

5.5 修改 proxy-start.sh 并启动

proxy-start.sh 脚本中默认只开启了对 WordPress 的支持。

```
#!/bin/bash

# note: please see the proxy.conf file in the config directory for additional settings.

#java -server -Xmx1024M -Xms1024M -jar heimdalldriver.jar "jdbc:heimdall://localhost:8087/dbdemo?hduser=test&hdpassword=test"
java -server -Xmx1024M -Xms1024M -jar heimdalldriver.jar "jdbc:heimdall://localhost:8087/wordpress?hduser=root&hdpassword=Tangkai1017!"
#java -server -Xmx1024M -Xms1024M -jar heimdalldriver.jar "jdbc:heimdall://localhost:8087/magento?hduser=mag&hdpassword=magento"
#java -server -Xmx1024M -Xms1024M -jar heimdalldriver.jar "jdbc:heimdall://localhost:8087/jira?hduser=mag&hdpassword=magento"
```

这里需要修改的参数有 :

- 1) Localhost : 修改为 Heimdall Server 地址。通常 proxy 和 Heimdall Server 会运行在同一台机器上, 因此无需修改 ;
- 2) 8087 : 修改为 Heimdall 的端口地址。默认是 8087 ;
- 3) Wordpress : 修改为在 Heimdall 控制台中 Virtual Database 里创建的虚拟数据库的名称 ;
- 4) hduser 和 hdpassword : 如果在 Heimdall 控制台中通过 Access Control 页面开启了访问控制 (即创建了用户并设置了密码) , 这里就需要填入具有访问权的用户名和密码。如

果没有开启访问控制，这两项的值可以为任意值；

5) 通过 `sudo ./proxy-start.sh start` 启动 Proxy。

5.6 修改 WordPress 数据库连接

- 1) 打开 WordPress 的数据库配置文件 `wp-config.php`；
- 2) 找到 `DB_HOST` 一项，修改为 “`<hd_ip>:5050`”。其中 `hd_ip` 为 Heimdall 服务器的地址，5050 为 Proxy 的端口。这里即使 Heimdall 安装在本地，也要使用实际的 IP 地址，而不能使用 `localhost` 或者 `127.0.0.1`。

经过以上步骤，应用就成功地接入了 Heimdall 平台。

5.7 经常出现的问题

- 端口 8087、5050、3306 等没有开启；
- MySQL 数据库只能通过 `localhost` 访问；
- Heimdall 文件夹权限不正确；
- 没有安装 JDK；
- MySQL 服务器和 driver 版本不一致或不兼容；
- 启动 Proxy 时没有 `sudo`；
- `DB_HOST` 的值使用了 `localhost` 而不是 IP 地址；
- Heimdall 开启了访问控制，但 Proxy 启动脚本中没有正确配置用户名和密码。

6 性能分析

6.1 基准性能测试

如果要了解 Heimdall 是如何有效地提高了系统的性能，就有必要先创建一个可复制的性能测试环境，然后多次重复执行后对性能参数进行统计。推荐的性能测试工具有：

- web 应用中“真实”用户性能可以用 Dynatrace 或者 Catchpoint 的产品来进行测试
- 本地基于 Web 的应用性能基准测试可用 Jmeter 或者 LoadRunner
- 基于 GUI 并且依赖数据库的应用，可用 Cucumber

任何性能基准测试，很重要的一点是可复制性和结果数据的一致性。即在相同的环境下可以重复进行性能测试，并且得到一致的结果。使用 Heimdall 进行应用性能优化时，如果要进行性能基准测试，需要注意：

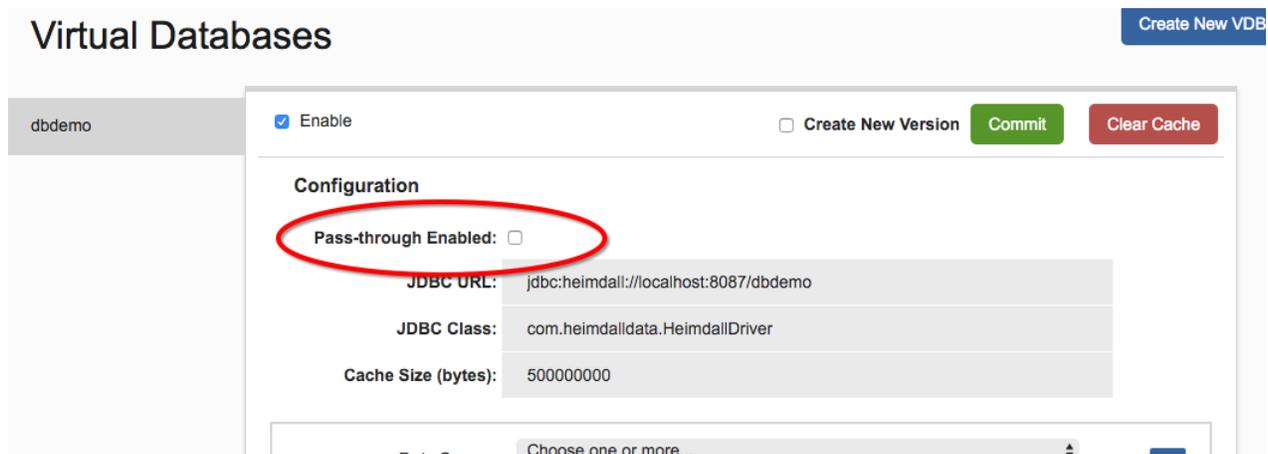
- 每次测试前重启应用。因为 Java 本身会执行运行时优化，而且应用可能发生内存泄漏，所以每次测试前重启应用非常重要。
- 将 Heimdall 服务器组件安装在独立的服务器上再进行性能测试，避免数据 log 进程占用过多的内存而影响应用性能。

6.2 Heimdall 性能基准测试步骤

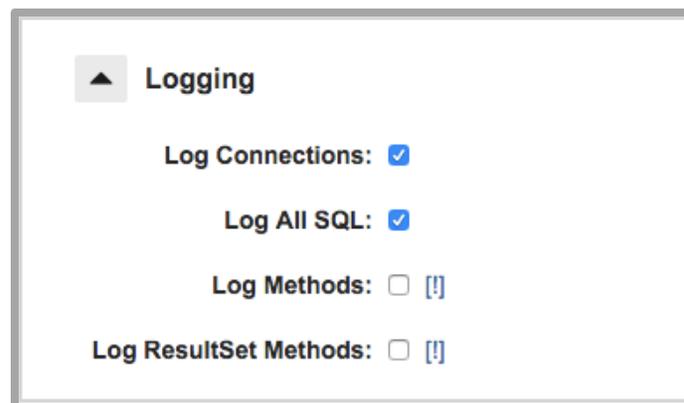
- 1) 首先在 Virtual Database 页面里选择“pass-through”。这将使数据绕开 HDAP 平台，基本上能够等同于安装 Heimdall 之前的性能。

Heimdall 数据访问平台 的安装与设置

- 2) 如果在 pass-through 模式下发现与应用原本性能差距很小，就可以进行下一步。取消 pass-through 选项，并注意不要启用 logging 或者其他策略。然后再次进行性能测试，与没有安装 Heimdall 时的性能进行比较。



- 3) 然后启动 logging 再次进行测试。Logging 功能可能会导致数据库性能的略微下降（约 5-10%）。



- 4) 收集了 SQL 数据后，使用 Heimdall 分析系统来配置缓存。请记住不是所有的系统推荐都需要缓存，但是系统推荐提供了一个较小的集合，涵盖了所有需要缓存的内容。
- 5) 下一步，关闭 logging，开启缓存，再次进行测试。这是 Heimdall 在生产环境中的常规运行模式。

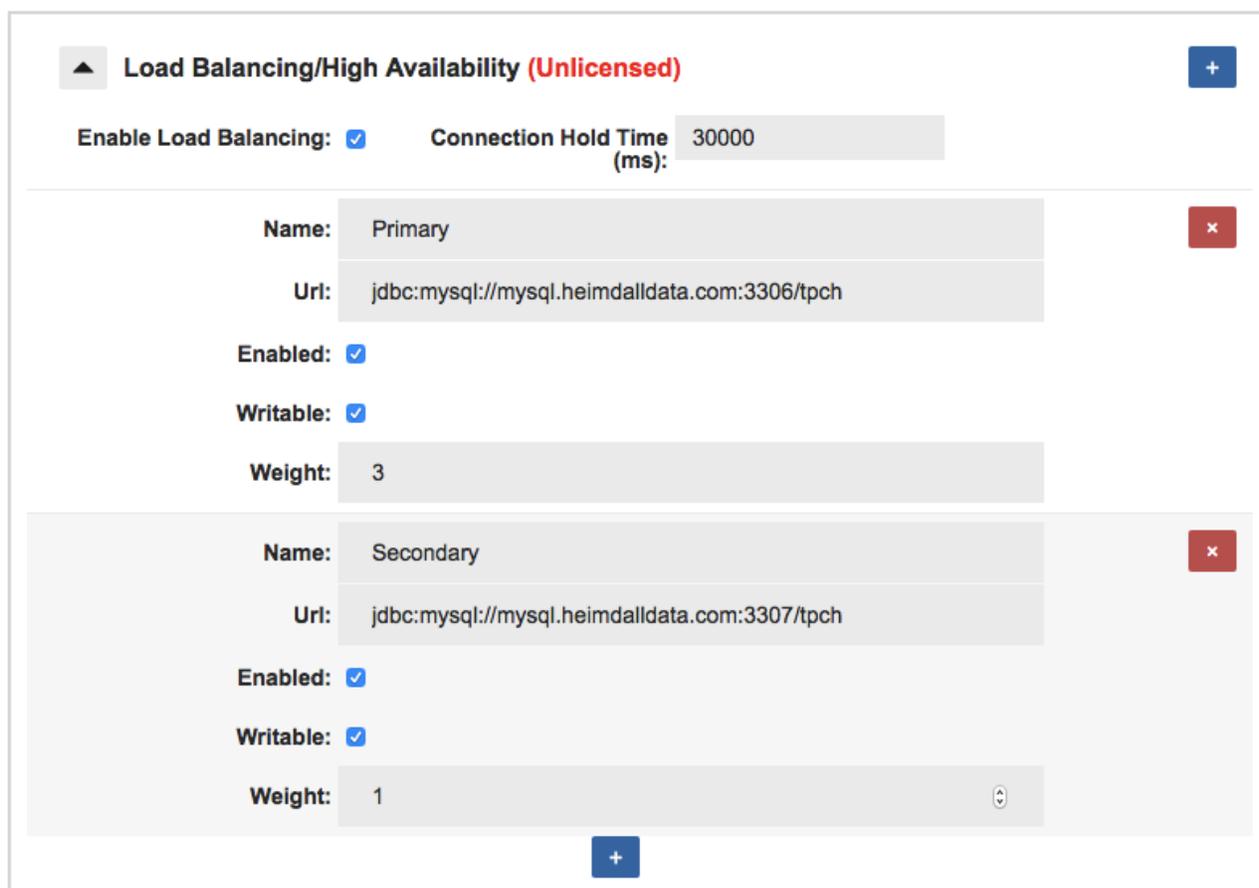
如果开启了缓存之后性能并没有得到明显的提升，甚至出现了不应有的性能下降，可以在 Virtual Database 页面上开启“method”和“result method”两个 logging 选项，让应用运行一段时间后，将 log 文件（默认情况下安装在 c:\Program Files\Heimdall\logs）打包后发送给 Heimdall 技术支持，我们将为您提供进一步的分析。

6.3 成功进行性能测试的几点提示：

- 在性能测试过程中监视 CPU 和内存负荷。
- 尽量避免使用虚拟环境进行性能测试。虚拟环境中网络有可能是不稳定的，可能导致系统性能的不确定性。
- 如果测试结果特别好或特别差，就需要仔细检查测试方法是否正确。
- 随时注意测试环境中可能影响测试结果的因素。如果观察到数据库连接数很多，又突然出现了断崖式下跌，有可能是 TCP 端口的问题。搜索“Windows 端口耗尽”，或者“windows ephemeral port exhaustion”查询网络提供的解决方案。
- 通过 Heimdall 技术支持来获得测试环境和结果分析方面的支持。

7 利用 HDAP 实现 HA

在 DataSource 页面可以配置 HA 的节点。在自带的示例程序 Traffic 中，美国节点配置了双主机结构。如图所示。



The screenshot displays the configuration for Load Balancing/High Availability (Unlicensed). The interface includes a toggle for 'Enable Load Balancing' (checked) and a 'Connection Hold Time (ms)' field set to 30000. Below this, two nodes are listed:

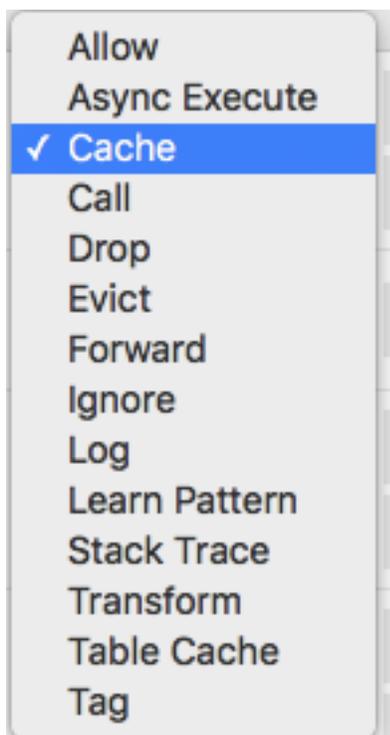
- Primary Node:**
 - Name: Primary
 - Url: jdbc:mysql://mysql.heimdalldata.com:3306/tpch
 - Enabled:
 - Writable:
 - Weight: 3
- Secondary Node:**
 - Name: Secondary
 - Url: jdbc:mysql://mysql.heimdalldata.com:3307/tpch
 - Enabled:
 - Writable:
 - Weight: 1

这里两个节点都是可读写的。因为是双主机结构，所以可以勾选 Enable Load Balancing 启动负载均衡器。均衡器会根据两个节点各自的权重值即 weight 来按比例分配数据流量。这里节点 Primary 的流量是节点 Secondary 的 3 倍。

利用这个机制，HDAP 还能实现更多的 HA 结构，如主-热备，主-冷备，读写分离，双主机+读从机集群等等。更具体的技术方案请联系我们的技术支持。联系方式参见最后一章。

8 HDAP 规则说明

HDAP 支持 14 种类型的规则。如下图所示。



- 1) Allow：用于 SQL 防火墙功能。Allow 是默认值，即允许 SQL 执行。
- 2) Async Execute：异步执行。当使用了异步执行之后，HDAP 会将写入 SQL 发送到数据库后立即返回，等待下一条 SQL。异步执行适用于对数据一致性要求不十分严格的场景。
- 3) Cache：将此 SQL 缓存。
- 4) Call：调用其他的规则表，便于对所有的规则进行管理。
- 5) Drop：用于 SQL 防火墙功能。Drop 即禁止执行此 SQL。

- 6) Evict : 用于刷新缓存数据。
- 7) Forward : 直接转发到其他的 Datasource。
- 8) Ignore : 忽略此 SQL。这是用户数据库查错和调试的规则。
- 9) Log : 将此 SQL 的执行和结果 log 下来。
- 10) Learn Pattern : 关联到其他的规则列表。如果一条 SQL 与任何原有的正则表达式都不精确匹配时, 会在 Learn Pattern 关联的规则列表中创建一条新的规则并发送到所有客户端。此功能便于用户管理敏感的 SQL。
- 11) Stack Track : 记录 SQL 所有关联的调用方法和返回值。
- 12) Transform : 将 SQL 进行变换后再发送到数据库中执行。
- 13) Table Cache : 将 SQL 关联的表进行缓存, 这样表中的数据有任何改动, 缓存中的数据将会得到通知并及时更新。
- 14) Tag : 标记某条 SQL。用户调试。

9 技术支持

任何问题，建议或者意见，可以通过以下渠道联系：

E-mail: support@heimdalldata.com

电话/微信：18611990193

QQ： [tang kai@heimdalldata.com](mailto:tangkai@heimdalldata.com)

网站：www.heimdalldata.cn