УДК 004.75

А.А. Сальников

Масштабована грід-служба керування участю в віртуальних організаціях

Проведен анализ работы классической реализации службы членства в виртуальных организациях и показаны ключевые ограничения ее масштабируемости и скорости обработки запросов. Разработана и внедрена служба управления членством в виртуальных организациях как центральная служба Украинской национальной грид-инфраструктуры.

The Virtual Organization membership service reference implementation analysis has been conducted. Key limitations including service scalability and request processing performance has been pointed out. Virtual Organization membership management service named PHP VOMS-Admin solving reference design drawbacks has been developed. The service has been deployed as a central service of Ukrainian National Grid-infrastructure.

Проведено аналіз роботи класичної реалізації служби членства в віртуальних організаціях та показано ключові обмеження її масштабованості та швидкості обробки запитів. Розроблено і впроваджено службу керування участю у віртуальних організаціях як центральний сервіс Української національної грід-інфраструктури.

Вступ. Ідея, закладена в концепцію грід – координоване спільне використання ресурсів та проведення досліджень динамічними об'єднаннями користувачів з різних установ – віртуальними організаціями (ВО) [1]. Остання є одиницею застосування політик доступу та використання ресурсів в грід-інфраструктурах. Авторизація користувачів грід враховує фактор участі в ВО з використанням сертифікату атрибутів (*Attribute Certificate – AC*) як розширення проксі-сертифікату [2, 3].

Для керування участю в ВО та роботи з сертифікатами атрибутів використовується окрема грід-служба рівня кооперації – служба засвідчення та керування участю в ВО (*Virtual Organization Membership Service*, *VOMS*) [4].

Існуюча реалізація служби *VOMS* має низку недоліків [5]. Апаратних ресурсів типового сучасного веб-сервера достатньо для обслуговування службою близько п'яти ВО. Таке обмеження є істотним в організації служби керування віртуальними організаціями національного рівня для декількох десятків ВО. Кластеризація веб-серверів частково вирішує цю проблему, але потребує додаткових ресурсів для підтримки та обслуговування такої конфігурації.

Іншим недоліком реалізації *VOMS* є низька швидкість обробки запитів, що суттєво впливає на взаємодію з грід-сервісами провайдерів ресурсів. До недоліків також можна віднести велику кількість залежностей від інших програмних продуктів, що не входять в репозиторії операційної системи.

Для грід-інфраструктур є актуальною розробка масштабованої грід-служби керування участю в ВО, позбавленої зазначених недоліків. Зокрема, гостро стоїть питання масштабованої служби *VOMS* для створення центру реєстрації віртуальних організацій в Українському національному грід-сегменті (УНГ). Реалізації такої служби присвячено дану статтю.

Служба засвідчення та керування участю в ВО

Розробку *VOMS* було розпочато в об'єднаних роботах проектів *European Data Grid* (*EDG*) та *Data TransAtlantic* Grid (*DataTAG*) з проблем авторизації учасників на рівні ВО, включаючи керування групами та ролями [4]. Подальша розробка та впровадження *VOMS* перейшли на загальноєвропейський рівень (проект *EGEE*, далі *EGI* [6]). Сьогодні роботу з *VOMS* підтримують всі проекти з розробки програмного забезпечення проміжного рівня грід. Служба *VOMS* є невід'ємною складовою *European Middleware Initiative* (*EMI*) та *Initiative for Globus in Europe* (*IGE*).

Схему роботи служби *VOMS* наведено на рис. 1. Однією з головних ідей архітектури *VOMS* є відокремлення керування участю в віртуальній організації (інтерфейс адміністрування) від

засвідчення участі в ВО. Кожна з компонент може працювати незалежно, більш того – можливі різні реалізації окремих компонент.

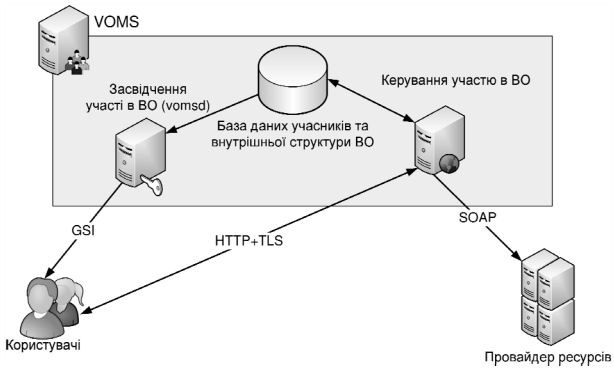

Рис. 1

Центральним елементом архітектури є база даних, яка зберігає список учасників ВО та їх атрибутів, включаючи ієрархічне об'єднання в групи та ролі в них. Отже, в *VOMS* закладено роботу з внутрішньою структурою ВО.

Служба засвідчення участі в ВО взаємодіє з клієнтом за допомогою протоколів *Grid Security Infrastructure* (*GSI*). Після взаємної аутентифікації служба аналізує запит користувача та звертається до бази даних для формування сертифіката атрибутів *AC* [3]. Запит містить інформацію про те, участь в якій саме групі чи яку роль засвідчувати. За відповідності запиту до інформації в базі даних сервіс включає до *AC* список повних імен атрибутів користувача (*Fully Qualified Attribute Name*, *FQAN*). Такий *AC* засвідчується цифровим підписом сервера *VOMS*, передається клієнту та додається до проксі-сертифіката делегації.

Веб-інтерфейс служби керування автоматизує як реєстрацію користувачів в ВО, так і керування параметрами участі. Такий режим роботи дозволяє використовувати службу *VOMS*, що не знаходиться під управлянням технічних спеціалістів ВО і не потребує наявності власних обчислювальних ресурсів.

Інша функція служби керування – взаємодія з провайдерами ресурсів. Така взаємодія відбувається за допомогою віддаленого виклику процедур з використанням протоколу *SOAP*. Наприклад, для автоматизації процесу створення динамічних списків унікальних імен (*Distinguished Name*, *DN*) учасників ВО використовується метод *getGridmapUsers* [7].

Проблеми масштабованості *VOMS*. Розвиток ВО в УНГ розпочався з єдиної ВО – *ukraine*, яка об'єднувала адміністраторів ресурсів. При популяризації грід-досліджень у прикладних областях науки в УНГ почали створюватись розрахункові ВО, першими з яких стали *moldyngrid* та *virgo.ua* [8].

ВО обслуговувались єдиним сервером *VOMS* Київського національного університету імені Тараса Шевченка. Саме цей підхід (вже функціонуюча служба доступна для використання всім ВО) став поштовхом до розвитку тематичних ВО в УНГ.

Додавання до обслуговування ще двох тестових ВО на сервері університету призвело до його нестабільної роботи через помилки виділення пам'яті веб-сервера (*java.lang.OutOfMemoryError*). Наявність помилок роботи з пам'яттю спостерігається в інших європейських проектах [5]. Для того щоб зрозуміти причину нестабільної роботи при збільшенні кількості ВО, було проведено аналіз архітектури та функціонування компонентів *VOMS*.

Реалізація служби засвідчення участі виконана мовою програмування *C++* та використовує *API* бібліотек *Globus Toolkit*. Для кожної ВО запускається окрема служба (*vomsd*), що очікує на запити клієнтів. Використання ресурсів такими службами мінімальне (близько 10 Мб оперативної пам'яті з яких 7 Мб займають спільні для всіх процесів *vomsd* бібліотеки, навантаження на процесор мінімальне), а отже істотних обмежень на кількість таких запущених служб немає.

Архітектура класичної служби керування, що від самого початку розробляється проектом *EDG*, представлена на рис. 2. Реалізацію веб-інтерфейсу виконано мовою *Java* – створено відповідний сервлет, що виконується в оточенні веб-сервера *Apache Tomcat*. Причому кожна з ВО обслуговується окремим сервлетом, а отже використовуються окремі потоки віртуальної машини *Java*, яким виділяються окремі ресурси сервера.

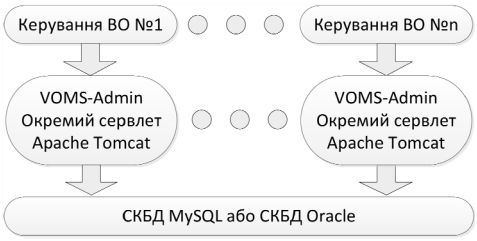

Рис. 2

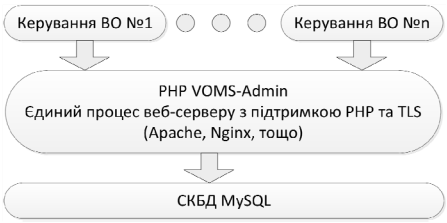

Рис. 3

Причини виникнення виключень *java.lang.OutOfMemoryError* веб-сервера *Apache Tomcat* наступні [9]:

- перевищення виділеного розміру динамічної пам'яті (*heap size*);
- наявність проблем в коді сервлета, які унеможливлюють правильну роботу «збирача сміття» (*garbage collector*);
- перевищення кількості допустимих потоків операційної системи (для більшості *Linux*-дистрибутивів за замовчуванням 1024).

Для тестування характеристик *EDG VOMS-Admin* було розгорнуто інсталяцію сервісів на тестовому вузлі кластера ІОЦ КНУ. Вузол обладнаний системною платою *Intel SE*7501*HG*2 з двома процесорами *Intel*(*R*) *Xeon*(*TM*) *Family* 15 *Model* 2 з частотою 3,06 ГГц та 2 Гб оперативної пам'яті. Встановлено операційну систему *Linux Fedora* 11, *Apache Tomcat*/5.5.27 та *JDK* 1.6.0-16. Використано сервлет *EDG VOMS-Admin* версії 2.0.18. Дослідження проводились для роботи з порожньою базою даних та з базою даних на 1000 учасників ВО.

У табл. 1 наведено результати тестування використання пам'яті сервісом *Apache Tomcat*/ 5.5.27 для роботи з порожньою базою даних та базою даних на 1000 учасників ВО за відсутності звернень до сервера та під час обробки запитів. Значення *VmSize* відображає загальну кількість динамічної пам'яті, що запитав процес, *VmRSS* (*Resident Set Size*) – кількість пам'яті, яку реально використовують структури даних процесу. *VmExe* та *VmLib* – пам'ять, що займає виконуваний код та динамічно підключені бібліотеки відповідно. *CPU* – завантаженість процесора процесом веб-сервера.

Таблиця 1. Використання ресурсів процесом *Java* при використанні *EDG VOMS-Admin*

Параметр	Порожня база даних без запитів до сервера	База даних на 1000 учасників ВО без запитів до сервера	База даних на 1000 учасників ВО із запитами до сервера
VmSize	788812 кБ	807776 кБ	844764 кБ
VmRSS	174392 кБ	231888 кБ	281852 кБ
VmExe	36 кБ	36кБ	36 кБ
VmLib	9808 кБ	9808 кБ	9808 кБ
CPU	0,2-1,0%	0,2-1,0%	10-30%

API JDK не надає механізмів визначення ресурсів, що витрачаються на роботу окремо виділеного сервлета [10]. Отримані значення показують загальне використання пам'яті віртуальною машиною *Java* разом з кодом *Apache Tomcat*.

При додаванні сервлетів ключовим обмеженням є великий обсяг даних кожного сервлета. Використання пам'яті при запитах збільшується на величину понад 100 Мб для кожного сервлета. Розмір виконуваного коду, який є спільним для сервлетів, складає 22 Мб.

Отже, використання окремих сервлетів в архітектурі *EDG VOMS-Admin* та їх вимоги до оперативної пам'яті є ключовою проблемою масштабованості *VOMS*, яка обмежує кількість ВО, що може обслуговувати сервер.

Час обробки *SOAP*-запитів. Інша проблема полягає в необхідності одночасного обслуговування запитів службою керування участю в ВО. В класичній інсталяції програмного забезпечення проміжного рівня в УНГ оновлення списків учасників ВО сервісами грід відбувається за допомогою підсистеми запланованих завдань *UNIX*-подібних ОС – *cron*. При інсталяції кожен кластер грід-середовища має однакові налаштування *cron* для оновлення списків ВО, що призводить до одночасних запитів до сервера *VOMS*.

Запити до сервера виконуються за протоколом *HTTPS*, налаштованим для авторизації клієнтів за допомогою їх персонального сертифі-

ката. Стандартні утиліти тестування продуктивності веб-застосувань (наприклад, *Apache Benchmark*) не підтримують такої конфігурації. Тому для дослідження часу відгуку *VOMS*-сервера на *SOAP*-запити було створено сценарії тестування мовою *POSIX SHELL*. В сценаріях використано утиліту *cURL*, яка дозволяє провести вимірювання в заданих умовах. Використовуючи параметри *time_pretransfer* (час від початку роботи до установки сессії з сервером) та *time_starttransfer* (час прийому першого байта відповіді, тобто *time_pretransfer* плюс час обробки запиту сервером) отримано час обробки запиту. Точність вимірювання часу *cURL* складає 1 мс.

Вимірювання проводились на різній кількості одночасних запитів (використовувались паралельно запущені процеси, які виконували запит до сервера за сигналом). Результати вимірювань показали велику дисперсію і представлені розподілом імовірностей часу обробки запиту. Було проведено цикли по 1000 та 10 000 вимірювань, які показали, що отриманий розподіл залишається сталим.

Використано дві бази даних – порожня (ВО, що не має учасників) та база на 1000 учасників. Результати дослідження представлені на рис 4 та 5.

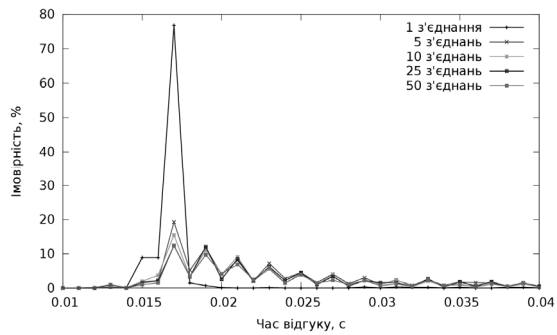

Рис. 4

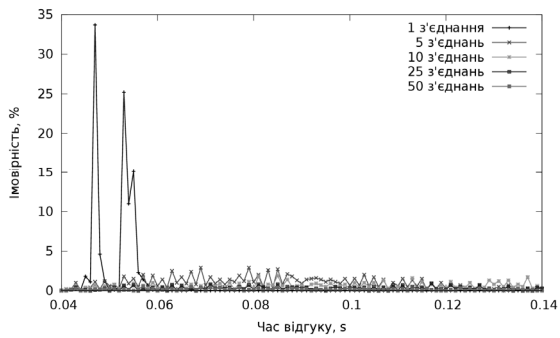

Рис. 5

На рису. 6 наведено порівняльний вигляд розподілів для різних баз даних при одному та 10 паралельних запитах.

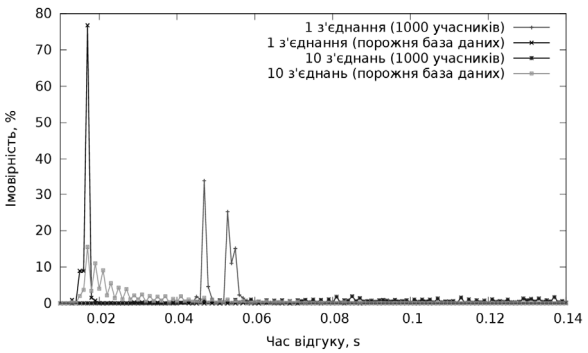

Рис. 6

У випадку порожньої бази даних збільшення кількості одночасних з'єднань збільшує дисперсію результатів, проте найбільш імовірне значення часу відгуку залишається рівним 0,017 с. Для бази даних на 1000 учасників при одному підключенні час відгуку збільшується до 0,045−0,055 с, а вже при п'яти паралельних запитах зникає чітко виражена форма розподілу. Відсоток запитів, час відгуку на які перевищує 1 с (рис. 7), істотно зростає з кількістю одночасних запитів.

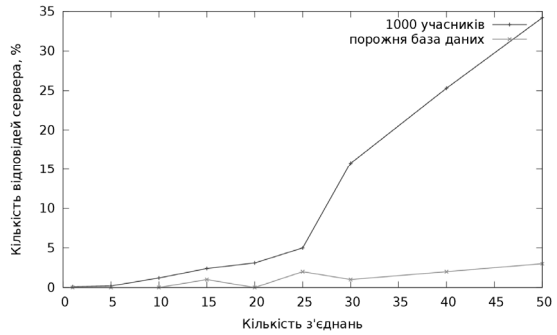

Рис. 7

Підсумовуючи проведений аналіз, можна зазначити, що для *EDG VOMS-Admin* існує проблема масштабування. При збільшенні кількості ВО, що обслуговуються сервером, та провайдерів ресурсів, які оновлюють списки користувачів ВО, стає істотним використання ресурсів сервера, а час відгуку зростає до значень, більших за одну секунду.

Проблема масштабування стає завадою для створення централізованих серверів служб обліку та керування участю в ВО національних

грід-інфраструктур, які б обслуговували десятки ВО. Для УНГ, зокрема, через відсутність можливостей створення та обслуговування власних серверів *VOMS* така проблема стає завадою розвитку тематичних ВО.

PHP VOMS-Admin

Проблему масштабування централізованого сервісу *VOMS* можна вирішити шляхом збільшення потужностей апаратних засобів та кластеризації, але такий підхід потребує додаткових ресурсів для підтримки та обслуговування такої конфігурації.

З аналізу випливає, що проблемна компонента служби засвідчення та керування участю в ВО – реалізація інтерфейсу керування *EDG VOMS-Admin*. Призначення інтерфейсу адміністрування – створення інтерфейсів взаємодії з базою даних: веб-інтерфейсу для користувачів та адміністраторів та *RPC SOAP*-інтерфейсу для автоматизованої роботи грід-сервісів.

Отже, інший шлях вирішення проблеми масштабування – окрема служба керування зі значно меншими вимогами до апаратних ресурсів, позбавлена вказаних недоліків. При цьому необхідно дотримуватись вимог сумісності схеми бази даних зі службою засвідчення (*vomsd*), що також дозволить виконувати прозору міграцію між рішеннями для вже існуючих ВО.

Виходячи з таких вимог, було розроблено програмне забезпечення *PHP VOMS-Admin*, що реалізує всі функції служби керування *EDG VOMS-Admin* та здатне обслуговувати сотні ВО на одному сервері.

Програмне забезпечення створено мовою програмування *PHP*, що дозволяє використовувати переважну більшість існуючих веб-серверів на різних програмних конфігураціях. *PHP VOMS-Admin* побудовано за схемою «Модель-вигляд-контролер» (*Model-view-controller – MVC*) – окремий модуль для виконання операцій з базою даних ВО, окремо виділені елементи, відповідальні за відображення веб-інтерфейсу і передачу параметрів від користувача та функції обробки запитів. Програмне забезпечення розповсюджується у вигляді вихідних кодів за ліцензією *Apache*2.

На відміну від роботи *EDG VOMS-Admin* (див. рис. 2), при роботі *PHP VOMS-Admin* виконується один і той самий код, проте звертається до різних баз даних в залежності від *URL* запиту (див. рис. 3). Параметри доступу до кожної бази даних задаються в файлах конфігурації.

В *PHP VOMS-Admin* реалізовано модуль для роботи з СКБД *MySQL*. Підтримки *Oracle* наразі немає. База даних ВО досить мала (навіть при 1000 учасників) і ефективно працює з використанням *MySQL* при незначних вимогах до ресурсів. СКБД *Oracle* для обслуговування однієї бази даних використовує близько 1 Гб ОЗУ, а для більшої кількості баз даних ВО потребує придбання ліцензії. Відповідно використання СКБД *Oracle* не сприяє досягненню масштабованості сервера *VOMS* і не розглядається як пріоритетний напрям розробки. Але завдяки використанню архітектури побудови веб-застосування *MVC* така підтримка може бути прозоро додана.

Масштабованість *PHP VOMS-Admin*. Аналогічно до тестування *EDG VOMS-Admin*, досліджено вимоги до ресурсів та час відгуку на *SOAP*-запити розробленого рішення з використанням того самого тестового вузла кластеру КНУ.

Для виконання коду *PHP VOMS-Admin* було використано веб-сервер *Apache*, як найбільш розповсюджений, доступний з репозиторіїв сервер з вбудованим інтерпретатором мови *PHP*.

Результати тестування використання ресурсів наведено в табл. 2. Кожен з робочих процесів *Apache* (в стандартній конфігурації запущено вісім робочих процесів) є однопоточним застосуванням, що обробляє запити. Для апаратно-програмної конфігурації тестового сервера використання динамічної пам'яті процесом – 80 Мб, з них близько 9 Мб структур даних і 35 Мб загальних бібліотек (спільних для всіх робочих процесів). При надходженні *SOAP*-запитів відбувається інтерпретація коду *PHP VOMS-Admin*, яка збільшує використання пам'яті всього на 5 Мб, а навантаження на процесор порівняно з роботою *EDG VOMS-Admin* сервлета *Apache Tomcat* в 30 разів менше (в табл. 2 також наведено результати попереднього тестування *EDG VOMS-Admin* під навантаженням для порівняння).

Таблиця 2. Використання ресурсів робочим процесом сервера *Apache* при роботі з *PHP VOMS-Admin*

Параметр	*Apache* без запитів до сервера	Виконання запитів до *PHP VOMS-Admin* сервером *Apache*	Виконання запитів до *EDG VOMS-Admin* сервлета сервером *Apache Tomcat*
VmSize	79872 кБ	81164 кБ	844764 кБ
VmRSS	9436 кБ	14508 кБ	281852 кБ
VmExe	312 кБ	312 кБ	36 кБ
VmLib	37860 кБ	37860 кБ	9808 кБ
CPU	<0,1%	0,3-1,0%	10-30%

Використання пам'яті сервером *Apache* не залежить від кількості ВО, що обслуговуються *PHP VOMS-Admin*, а лише збільшується на декілька мегабайт при збільшенні кількості одночасних підключень до будь-якої ВО. Отже, для *PHP VOMS-Admin* немає принципового обмеження апаратних ресурсів при масштабуванні, яке існує для *EDG VOMS-Admin*.

Результати тестування часу обробки *SOAP*-запитів для *PHP VOMS-Admin* у роботі з порожньою базою даних та базою даних на 1000 учасників ВО наведено на рис. 8 та 9 відповідно. На відміну від результатів *EDG VOMS-Admin* дисперсія значно менша і найбільш імовірний час відгуку становить 0,001−0,003 с при використанні як порожньої, так і заповненої бази даних навіть при 50-ти паралельних підключеннях.

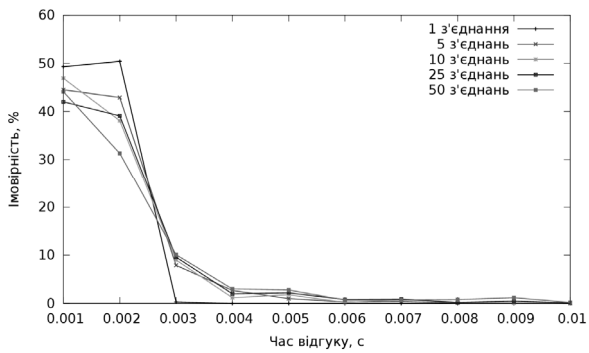

Рис. 8

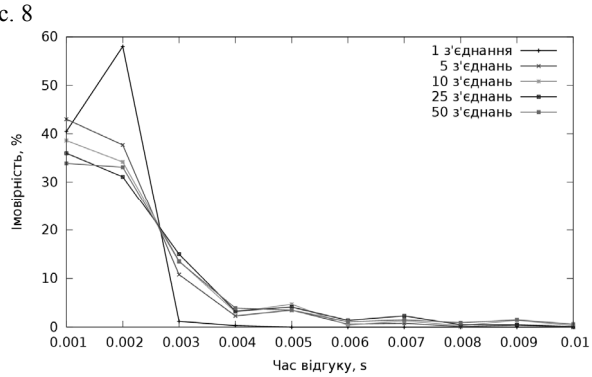

Рис. 9

На рис. 10 наведено порівняння розподілів, яке ілюструє масштабованість *PHP VOMS-Admin* за кількістю одночасних підключень при різних базах даних.

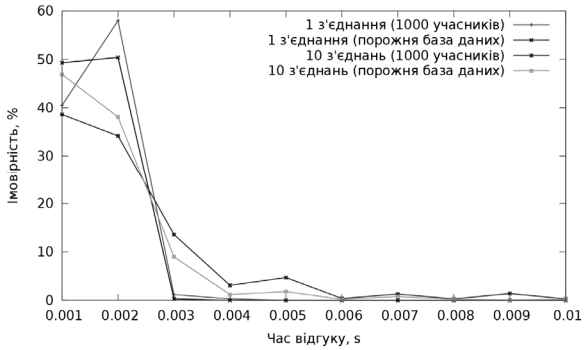

Рис. 10

Результати порівняння отриманих розподілів *PHP VOMS-Admin* з *EDG VOMS-Admin* наведено на рис. 11. Отже, було показано, що розроблене рішення дозволяє організувати роботу центрального сервера служб обліку та керування участю в ВО, що здатний обслуговувати сотні ВО на єдиному сервері поряд з іншими веб-застосуваннями на відміну від *EDG VOMS-Admin*.

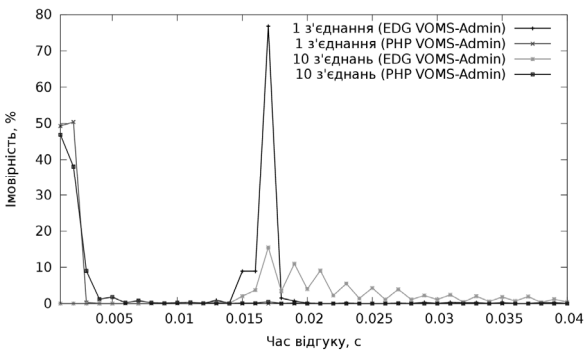

Рис. 11

Додаткові функції *PHP VOMS-Admin*. Додатково до вирішення проблеми масштабування сервісу *VOMS*, в *PHP VOMS-Admin* було реалізовано низку функцій, відсутніх в *EDG VOMS-Admin*. Для розгортання централізованих сервісів інфраструктури додано можливість конфігурації посилань на інші сервери *VOMS*. Це надає можливість створення єдиної точки входу для користувачів інфраструктури для перегляду списків тематичних ВО та початку роботи.

Для автоматизації процесу додавання до обслуговування нових ВО в інтерфейсі *PHP*

VOMS-Admin реалізовано форму реєстрації ВО, яка автоматично визначає ідентифікацію адміністратора, аналізуючи персональний сертифікат, і запитує ім'я та параметри нової ВО. Надіслана форма зберігається на сервері, а адміністратору *VOMS* надходить повідомлення про запит на обслуговування нової ВО. За допомогою розробленого сценарію мовою *POSIX SHELL*, що розповсюджується разом з *PHP VOMS-Admin*, весь процес створення бази даних та конфігураційних файлів як служби засвідчення, так і інтерфейсу адміністрування, виконується автоматично.

Порівняно з *EDG VOMS-Admin* до реалізації інтерфейсу закладено підтримку багатомовності (український, російський та англійський переклад), додано форми зворотного зв'язку з адміністратором сервера та зміни параметрів відображення інформації про ВО (короткий опис, посилання на веб-сайт тощо).

Впровадження служби керування участю в ВО. Розроблену грід-службу керування участю в ВО було впроваджено на сервері КНУ, що дозволило вирішити проблеми масштабованості для Українського національного грід-сегменту. Використання *PHP VOMS-Admin* як всеукраїнської служби для УНГ стало поштовхом до розвинення тематичних ВО, і наразі сервер обслуговує 15 українських ВО, кількість яких зростає без істотного навантаження на ресурси.

Готові пакети програмного забезпечення *PHP VOMS-Admin* було додано до центральних репозиторіїв операційної системи сімейства *Red Hat – Fedora*, а також в репозиторій *Extra Packages for Enterprise Linux* (*EPEL*), призначений для дистрибутивів *RHEL*, *CentOS* та *Scientific Linux*. Отже, на вказаних операційних системах інсталяцію служби можна виконати за допомогою стандартного менеджера пакетів *YUM*.

Створено веб-сайт проекту, який розміщено за адресою: *http*://*grid.org.ua*/*development*/ *pva*/. Сайт містить інформацію про роботу *PHP VOMS-Admin*, документацію щодо інсталяції та конфігурації, вихідні коди та пакети програмного забезпечення.

Висновки. Проведено аналіз роботи служби засвідчення та керування участю в ВО в грід-інфраструктурі та показано існування проблем її масштабованості.

В результаті аналізу архітектури *VOMS* та дослідження використання апаратних ресурсів виявлено проблемну компоненту служби – *EDG VOMS-Admin*, яка використовує значну кількість ОЗУ та процесорної потужності для обслуговування навіть однієї ВО. Збільшення кількості ВО призводить до помилок у роботі з пам'яттю.

Дослідження часу обробки запитів *EDG VOMS-Admin* показали значну дисперсію розподілу результатів та її збільшення зі збільшенням кількості одночасних запитів і перевищення значення в одну секунду.

Для вирішення зазначених проблем розроблено альтернативне програмне забезпечення для адміністрування участі в ВО – *PHP VOMS-Admin*. Останній позбавлений проблем масштабованості за кількістю ВО, які обслуговуються, та одночасних підключень до сервера. Результати тестування *PHP VOMS-Admin* показали незначне збільшення використання пам'яті і стабільний час відгуку в межах 0,001−0,003 с навіть при паралельних зверненнях.

Поряд з вирішенням проблем масштабованості було реалізовано додаткові функції, серед яких підтримка багатомовності інтерфейсу, автоматизація процесу додавання до обслуговування нових ВО та робота зі списками зовнішніх серверів.

Використання *PHP VOMS-Admin* для централізованого сервера служби засвідчення та керування участю в ВО в Українській національній грід-інфраструктурі стало поштовхом до розвинення тематичних ВО. На противагу трьом ВО, наразі сервер *VOMS* Київського національного університету імені Тараса Шевченка обслуговує 15 ВО, заснованих провідними дослідницькими інститутами НАН України та вищими навчальними закладами.

1. *Foster I.*, *Kesselman C.*, *Tuecke S.* The Anatomy of the Grid-Enabling Scalable Virtual Organizations // Int. J. of Supercomputer Appl. – 2001. – **15**. – P. 2001.

2. *Tuecke S.* Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. – RFC 3820 (Prop. Stand.). – 2004. – Jun. – http://www.ietf.org/rfc/rfc3820.txt.
3. *Farrell S.* An Internet Attribute Certificate Profile for Authorization. – RFC 5755 (Prop. Stand.). – 2010. – Jan. – http://www.ietf.org/rfc/rfc5755.txt.
4. *An Authorization* System for Virtual Organizations / / R. Alfieri, R. Cecchini, V. Ciaschini et al. // Proc. of the 1st Europ. Across Grids Conf., Santiago de Compostela, 2003. – P. 13–14.
5. *Forti A.*, *Jones M.*, *Dolgobrodov S.* VOMS deployment in GridPP and NGS // Proc. UK All Hands, 2006.
6. *Infrastructure*, European Grid. Towards to sustainable grid infrastructure. – http://www.egi.eu/ – 2011.
7. *Lorentey K.*, *Frohner A.* EDG-VOMS-ADMIN Developer's guide. – 2004. – 05. Feb.
8. *Український* академічний Грід: досвід створення й перші результати експлуатації / Ю.В. Бойко, М.Г. Зинов'єв, О.О. Судаков та ін. // Математичні машині і системи. – 2008. – **1**. – С. 67–84.
9. *The Apache* Tomcat 5.5 Servlet/JSP Container. – http://tomcat.apache.org/tomcat-5.5-doc
10. *Java* SE 6 Documentation. – http://docs.oracle.com/javase/6/docs/

Поступила 10.02.2012
Тел. для справок: (044) 526-1214 (Киев)
E-mail: *manf@grid.org.ua*
Web: *http://grid.org.ua*, *http://cluster.kiev.ua*

●

А.А. Сальников

Масштабируемая грид-служба управления участием в виртуальных организациях

Введение. Идея, положенная в основу концепции грид – координированное совместное использование ресурсов и выполнение исследований динамическими объединениями пользователей из разных учреждений – виртуальными организациями (ВО) [1]. Виртуальная организация есть единицей применения политик доступа и использования ресурсов в грид-инфраструктурах. Авторизация пользователей грид учитывает фактор членства в ВО с использованием сертификата атрибутов (*Attribute Certificate – AC*) как расширение прокси-сертификата [2, 3].

Для управления членством в ВО и работы с *AC* используется отдельная грид-служба уровня кооперации – служба членов ВО (*Virtual Organization Membership Service – VOMS*) [4].

Существующая реализация службы *VOMS* имеет ряд недостатков [5]. Аппаратных ресурсов типичного современного веб-сервера хватит для обслуживания около пяти ВО. Такое ограничение существенно при организации службы управления виртуальными организациями национального уровня для нескольких десятков ВО. Кластеризация веб-серверов частично решает эту проблему, но требует привлечения дополнительных ресурсов для поддержки и обслуживания такой конфигурации.

Другим недостатком реализации *VOMS* есть низкая скорость обработки запросов, что существенно влияет на взаимодействие с грид-службами провайдеров ресурсов. К недостаткам также можно причислить большое количество зависимостей от других программных продуктов, не входящих в репозитории операционной системы.

Для грид-инфраструктур актуальна разработка масштабируемой грид-службы управления членством в ВО, не имеющей указанных недостатков. В частности, остро стоит вопрос масштабируемой службы *VOMS* для создания центра регистрации ВО в Украинском национальном грид-сегменте (УНГ). Реализации такой службы посвящена данная статья.

Служба членства в виртуальных организациях

Разработка *VOMS* была начата в объединенных работах проектов *European Data Grid* (*EDG*) и *Data TransAtlantic Grid* (*DataTAG*), посвященных проблемам авторизации участников на уровне ВО, включая управление группами и ролями [4]. Последующая разработка и внедрение *VOMS* перешли на общеевропейский уровень (проект *EGEE*, дальше *EGI* [6]). Сегодня работу с *VOMS* поддерживают все проекты, разрабатывающие промежуточное программное обеспечение грид. Служба *VOMS* – неотъемлемая составляющая *European Middleware Initiative* (*EMI*) и *Initiative for Globus in Europe* (*IGE*).

Схема работы службы VOMS показана на рис. 1. Одной из основных идей архитектуры VOMS есть разделение управления членством в виртуальных организациях (интерфейс администрирования) от подтверждения членства в ВО. Каждая составляющая может работать независимо, более того – возможны разные реализации отдельных составляющих.

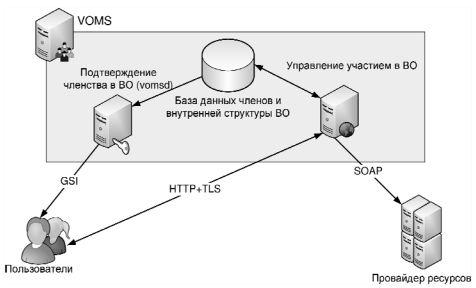

Рис. 1

Центральным элементом архитектуры служит база данных, в которой храниться список участников ВО и их атрибутов, включая иерархическое объединение в группы и роли в них. Таким образом, в *VOMS* заложена поддержка внутренней структуры ВО.

Служба подтверждения участия в ВО взаимодействует с клиентом при помощи протоколов *Grid Security Infrastructure* (*GSI*). После взаимной аутентификации служба анализирует запрос пользователя и обращается в базу данных для формирования сертификата атрибутов *AC* [3]. Запрос содержит информацию о том, участие в какой группе или какую роль подтверждать. В случае соответствия запроса информации в базе данных служба включает в *AC* список полных имен атрибутов (*Fully Qualified Attribute Name – FQAN*). Такой *AC* подтверждается цифровой подписью *VOMS*, передается клиенту и добавляется к прокси-сертификату.

Веб-интерфейс службы управления автоматизирует как регистрацию пользователей в ВО, так и управление параметрами членства. Такой режим работы позволяет использовать службу *VOMS*, которая не находится под управлением технических специалистов ВО и не нуждается в собственных вычислительных ресурсах.

Другая функция службы управления – взаимодействие с провайдерами ресурсов. Такое взаимодействие осуществляется посредством удаленного вызова процедур с использованием протокола *SOAP*. Например, для автоматизации процесса создания динамических списков отличительных имен (*Distinguished Name – DN*) членов ВО используется метод *getGridmapUsers* [7].

Проблемы масштабируемости *VOMS*. Развитие ВО в УНГ началось с единой ВО – *ukraine*, объединявшей администраторов ресурсов. При популяризации грид-исследований в прикладных областях науки в УНГ началось создание расчетных ВО, первыми из которых стали *moldyngrid* и *virgo.ua* [8].

ВО обслуживались единым сервером *VOMS* Киевского национального университета имени Тараса Шевченко. Именно этот подход (уже функционирующая служба, доступная для использования ВО) стал толчком к развитию тематических ВО в УНГ.

Добавление к обслуживанию еще двух тестовых ВО на сервере университета привело к его нестабильной работе из-за ошибок выделения памяти веб-сервера (*java.lang.OutOfMemoryError*). Наличие ошибок выделения памяти наблюдается в других европейских проектах [5]. Для того, чтобы понять причину нестабильной работы при увеличении количества ВО, был проведен анализ архитектуры и функционирования компонентов *VOMS*.

Реализация службы подтверждения участия выполнена на языке программирования *C++* и использует *API* библиотек *Globus Toolkit*. Для каждой ВО запускается отдельная служба (*vomsd*), ожидающая запросов. Использование ресурсов такими службами минимально (около 10Мб оперативной памяти, из которых 7Мб занимают общие библиотеки, нагрузка на процессор минимальна), соответственно существенных ограничений по количеству запущенных служб нет.

Архитектура классической службы управления, изначально разрабатываемая проектом *EDG*, представлена на рис. 2. Реализация веб-интерфейса выполнена на языке *Java* – создан соответствующий сервлет, выполняемый в контексте веб-сервера *Apache Tomcat*. Причем каждая из ВО обслуживается при помощи отдельного сервлета, и соответственно используются отдельные потоки виртуальной машины *Java*, которым выделяются отдельные ресурсы сервера.

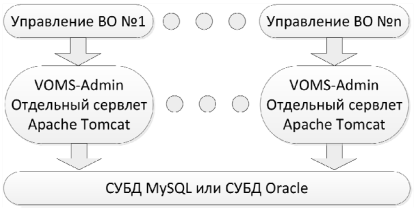

Рис. 2

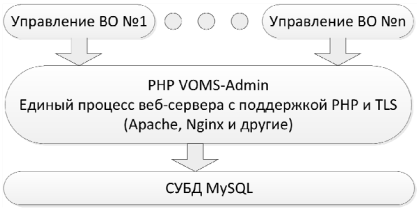

Рис. 3

Причины возникновения исключений *java.lang.OutOfMemoryError* веб-сервера *Apache Tomcat* следующие [9]:

- превышение выделенного размера динамической памяти (*heap size*);
- наличие проблем в коде сервлета, исключающие правильную работу «сборщика мусора» (*garbage collector*);
- превышение количества допустимых потоков операционной системы (для большинства *Linux*-дистрибутивов по умолчанию 1024).

Для тестирования характеристик *EDG VOMS-Admin* была развернута инсталляция сервисов на тестовом узле кластера КНУ. Узел оборудован системной платой *Intel SE*7501*HG*2 с двумя процессорами *Intel* (*R*) *Xeon* (*TM*) *Family* 15 *Model* 2 с частотой 3.06 ГГц и 2 Гб оперативной памяти. Установлена операционная система *Linux Fedora* 11, *Apache Tomcat*/5.5.27 и *JDK* 1.6.0-16. Использован сервлет *EDG VOMS-Admin* версии 2.0.18. Исследования проводились для работы с пустой базой данных и с базой данных на 1000 членов ВО.

В табл. 1 приведены результаты тестирования использования памяти службой *Apache Tomcat*/5.5.27 для работы с пустой базой данных и базой данных на 1000 членов ВО в отсутствие обращений к серверу и при обработке запросов. Значение *VmSize* отражает общее количество динамической памяти, запрошенной процессом, *VmRSS* (*Resident Set Size*) – объем памяти, которую действительно занимают данные процесса. *VmExe* и *VmLib* – память, занимаемая исполняемым кодом и динамически подключаемыми библиотеками соответственно, *CPU* – загруженность процессора процессом веб-сервера.

API JDK не предоставляет механизмов определения ресурсов, затрачиваемых на работу отдельно взятым сервлетом [10]. Полученные значения показывают общее

использование памяти виртуальной машиной *Java* вместе с кодом *Apache Tomcat*.

Таблица 1. Использование ресурсов процессом *Java* при выполнении *EDG VOMS-Admin*

Параметр	Пустая база данных без запросов к серверу	База данных на 1000 членов ВО без запросов к серверу	База данных на 1000 членов ВО с запросами к серверу
VmSize	788812 кБ	807776 кБ	844764 кБ
VmRSS	174392 кБ	231888 кБ	281852 кБ
VmExe	36 кБ	36кБ	36 кБ
VmLib	9808 кБ	9808 кБ	9808 кБ
CPU	0,2-1,0%	0,2-1,0%	10-30%

При добавлении сервлетов ключевым ограничением становится большой объем данных каждого сервлета. Использование памяти при запросах увеличивается на величину более 100 Мб для каждого сервлета. Размер исполняемого кода, общего для сервлетов, составляет 22 Мб.

Таким образом, использование отдельных сервлетов в архитектуре *EDG VOMS-Admin* и их требования к оперативной памяти – ключевая проблема масштабируемости *VOMS*, ограничивающей количество обслуживаемых сервером ВО.

Время обработки *SOAP*-запросов. Другая проблема заключается в необходимости одновременного обслуживания запросов службой управления участием в ВО. В классической инсталляции программного обеспечения промежуточного уровня в УНГ, обновление списков участников ВО сервисами грид происходит с помощью подсистемы запланированных задач *UNIX*-подобных ОС – *cron*. При инсталляции каждый кластер грид-среды имеет одинаковые настройки *cron* для обновления списков ВО, что приводит к одновременным запросам к серверу *VOMS*.

Запросы к серверу осуществляются при помощи протокола *HTTPS* с использованием авторизации клиентов посредством их персонального сертификата. Стандартные утилиты тестирования производительности веб-приложений (например, *Apache Benchmark*) не поддерживают такой конфигурации. Поэтому для исследования времени отклика *VOMS*-сервера на *SOAP*-запросы были созданы сценарии тестирования на языке *POSIX SHELL*. В сценариях использована утилита *cURL*, позволяющая провести измерения в заданных условиях. Используя параметры *time_pretransfer* (время от начала работы до установки сессии с сервером) и *time_starttransfer* (время приема первого байта ответа – *time_pretransfer* плюс время обработки запроса сервером) получено время обработки запроса. Точность измерения времени *cURL* составляет 1 мс.

Измерения проводились на разном количестве одновременных запросов (использовались параллельно запущенные процессы, выполнявшие запрос к серверу по сигналу). Результаты измерений показали большую дисперсию и представлены распределением вероятностей времени обработки сигнала. Были проведены циклы по 1000 и 10 000 измерений, показавшие, что полученное распределение вероятностей остается постоянным.

Использованы две базы данных – пустая (ВО без участников) и база на 1000 участников. Результаты исследования представлены на рис. 4 и 5.

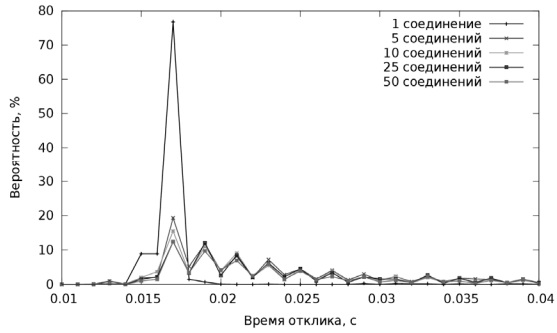

Рис. 4

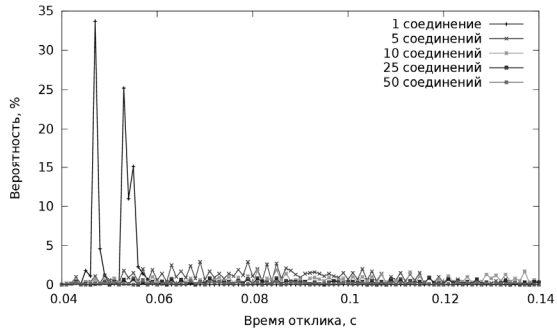

Рис. 5

На рис. 6 приведен сравнительный вид распределений для различных баз данных при одном и 10-ти параллельных запросах.

В случае пустой базы данных увеличение количества одновременных соединений увеличивает дисперсию результатов, однако наиболее вероятное значение времени отклика остается равным 0,017 с. Для базы данных на 1000 участников при одном подключении время отклика увеличивается до 0,045−0,055 с, а уже при пяти параллельных запросах исчезает четко выраженная форма распределения. Процент запросов, время отклика на которые превышает 1 с (рис. 7), существенно возрастает с увеличением количества одновременных запросов.

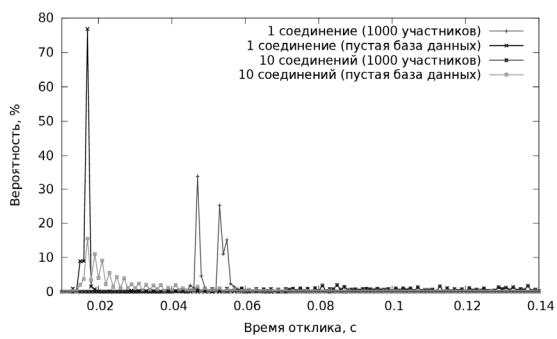

Рис. 6

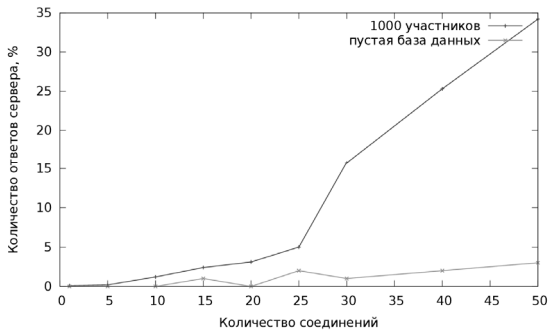

Рис. 7

Подытоживая проведенный анализ, следует отметить, что для *EDG VOMS-Admin* существует проблема масштабирования. При увеличении количества ВО, обслуживаемых сервером, и провайдеров ресурсов, обновляющих списки пользователей ВО, становится существенным использование ресурсов сервера, а время отклика возрастает до значений, больших 1 с.

Проблема масштабирования становится помехой для создания централизованных служб учета и управления участием в ВО национальных грид-инфраструктур, обслуживающих десятки ВО. Для УНГ, в частности, в отсутствие возможностей создания и обслуживания собственных серверов *VOMS* такая проблема становится помехой для развития тематических ВО.

PHP VOMS-Admin

Проблему масштабирования централизованного сервиса *VOMS* можно решить путем увеличения мощностей аппаратных средств и кластеризации, но такой подход требует дополнительных ресурсов для поддержки и обслуживание конфигурации.

Из анализа следует, что проблемная компонента службы членства в ВО – реализация интерфейса управления *EDG VOMS-Admin*. Назначение интерфейса администрирования – создание интерфейсов взаимодействия с базой данных: веб-интерфейса для пользователей и администраторов и *RPC SOAP*-интерфейса для автоматизированной работы грид-сервисов.

Таким образом, другой путь решения проблемы масштабирования – отдельная служба управления ВО со значительно меньшими требованиями к аппаратным ресурсам, лишенная перечисленных недостатков. При этом необходимо соблюдать требования совместимости схемы базы данных со службой удостоверения (*vomsd*), что также позволит выполнять прозрачную миграцию между решениями для уже существующих ВО.

Исходя из таких требований, было разработано программное обеспечение *PHP VOMS-Admin*, где реализованы все функции службы управления *EDG VOMS-Admin*, но с возможностью масштабирования до сотен ВО на одном сервере.

Программное обеспечение создано на языке программирования *PHP*, что позволяет использовать подавляющее большинство существующих веб-серверов на различных программных конфигурациях. *PHP VOMS-Admin* построено по схеме «Модель–вид–контроллер» (*Model–view–controller – MVC*) – отдельный модуль для выполнения операций с базой данных ВО, выделены элементы, ответственные за отображение веб-интерфейса и передачу параметров от пользователя, и функции обработки запросов. Программное обеспечение распространяется в виде исходных кодов по лицензии *Apache*2.

В отличие от работы *EDG VOMS-Admin* (см. рис. 2), при работе *PHP VOMS-Admin* выполняется один и тот же код, но обращается к различным базам данных в зависимости от *URL*-запроса (см. рис. 3). Параметры доступа к каждой базе данных задаются в файлах конфигурации.

В *PHP VOMS-Admin* реализован модуль для работы с СУБД *MySQL*. Поддержки *Oracle* пока нет. База данных ВО достаточно мала (даже при 1000 участников) и эффективно работает с использованием *MySQL* при незначительных требованиях к ресурсам. СУБД *Oracle* для обслуживаемых одной базы данных использует около 1 Гб ОЗУ, а для большего количества баз данных (ВО) требует приобретения лицензии. Соответственно использование СУБД *Oracle* не способствует достижению масштабируемости сервера *VOMS* и не рассматривалось как приоритетное направление разработки. Но благодаря использованию архитектуры построения веб-приложения *MVC* такая поддержка может быть прозрачно добавлена.

Масштабируемость *PHP VOMS-Admin*. Аналогично тестированию *EDG VOMS-Admin*, проведено исследование требований к ресурсам и времени отклика на *SOAP*-запросы разработанного решения с использованием того же тестового узла кластера КНУ.

Для выполнения кода *PHP VOMS-Admin* был использован веб-сервер *Apache*, как наиболее распространенный, доступный из репозиториев сервер со встроенным интерпретатором языка *PHP*.

Результаты тестирования использования ресурсов приведены в табл. 2. Каждый из рабочих процессов *Apache* (в стандартной конфигурации запущено восемь рабочих процессов) является однопоточным приложением, обрабатывающим запросы. Для аппаратно-программной конфигурации тестового сервера использование динамической памяти процессом – 80 Мб, из них около 9 Мб структур данных и 35 Мб общих библиотек (одинаковых для всех рабочих процессов). При поступлении *SOAP*-запросов происходит интерпретация кода *PHP VOMS-Admin*, увеличивающая использование памяти всего на 5 Мб, а нагрузка на процессор сравнительно с работой *EDG VOMS-Admin* сервлета *Apache Tomcat* в 30 раз меньше (в табл. 2 также приведены результаты предыдущего тестирования *EDG VOMS-Admin* под нагрузкой для сравнения).

Использование памяти сервером *Apache* не зависит от количества ВО, обслуживаемых *PHP VOMS-Admin*, а лишь увеличивается на несколько мегабайт при увеличении количества одновременных подключений к любой из ВО. Таким образом, для *PHP VOMS-Admin* не существует принципиального ограничения по аппаратным ресурсам при масштабировании, которое существует для *EDG VOMS-Admin*.

Таблица 2. Использование ресурсов рабочим процессом сервера *Apache* при работе с *PHP VOMS-Admin*

Параметр	***Apache* без запросов к серверу**	**Выполнение запросов к *PHP VOMS-Admin* сервером *Apache***	**Выполнение запросов к *EDG VOMS-Admin* сервлету сервером *Apache Tomcat***
VmSize	79872 кБ	81164 кБ	844764 кБ
VmRSS	9436 кБ	14508 кБ	281852 кБ
VmExe	312 кБ	312 кБ	36 кБ
VmLib	37860 кБ	37860 кБ	9808 кБ
CPU	<0,1%	0,3-1,0%	10-30%

Результаты тестирования времени обработки *SOAP*-запросов *PHP VOMS-Admin* при работе с пустой базой данных и базой данных на 1000 участников ВО приведены на рис. 8 и 9 соответственно. В отличие от результатов *EDG VOMS-Admin*, дисперсия значительно меньше, и наиболее вероятное время отклика составляет 0,001−0,003 с при использовании как пустой, так и заполненной базы данных даже при 50-ти параллельных подключениях. На рис. 10 приведено сравнение распределений, иллюстрирующее масштабируемость *PHP VOMS-Admin* по количеству одновременных подключений при различных базах данных.

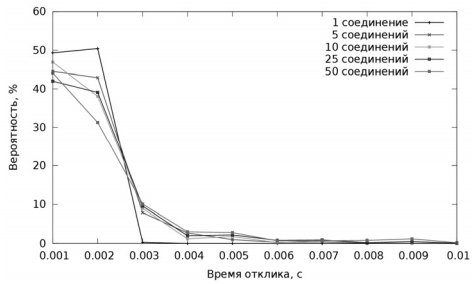

Рис. 8

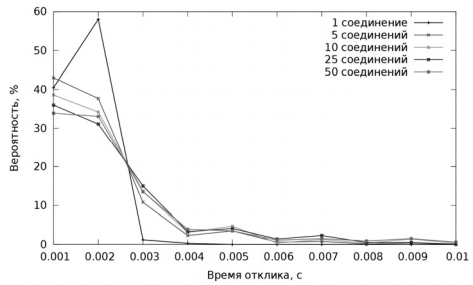

Рис. 9

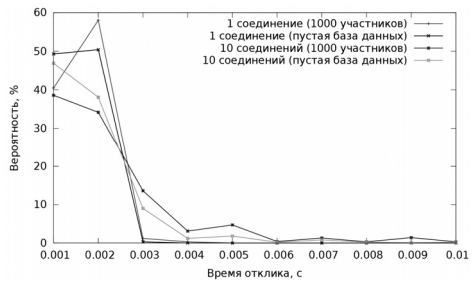

Рис. 10

Результаты сравнения полученных распределений *PHP VOMS-Admin* с *EDG VOMS-Admin* приведены на рис. 11. Таким образом было показано, что разработанное решение позволяет организовать работу центрального сервера службы членства в ВО, который способен обслуживать сотни ВО на едином сервере вместе с другими веб-приложениями в отличие от *EDG VOMS-Admin*.

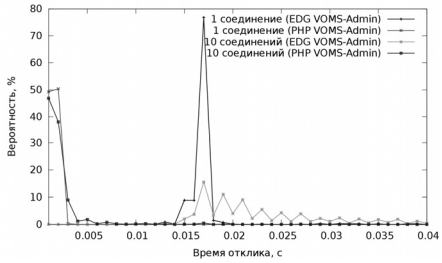

Рис. 11

Дополнительные функции *PHP VOMS-Admin*. В дополнение к решению проблемы масштабирования службы *VOMS*, в *PHP VOMS-Admin* реализован ряд функций, отсутствующих в *EDG VOMS-Admin*. Для развертывания централизованных сервисов инфраструктуры добавлена возможность конфигурации ссылок на другие серверы *VOMS*. Это дает возможность создания единой точки входа пользователям инфраструктуры для просмотра списков тематических ВО и начала работы.

Для автоматизации процесса добавления к обслуживанию новых ВО в интерфейсе *PHP VOMS-Admin* реализована форма регистрации ВО, автоматически определяющая идентификацию администратора, анализируя персональный сертификат, и спрашивающая имя и параметры новой ВО. Отправленная форма сохраняется на сервере, а администратору *VOMS* направляется уведомление о запросе на обслуживание новой ВО. С помощью разработанного сценария на языке *POSIX SHELL*, который распространяется вместе с *PHP VOMS-Admin*, весь процесс создания базы данных, конфигурационных файлов как службы удостоверения, так и интерфейса администрирования выполняется автоматически.

В сравнении с *EDG VOMS-Admin* в реализации заложена поддержка многоязычного интерфейса (украинский, русский и английский перевод), добавлены формы обратной связи с администратором сервера и изменения параметров отображения информации о ВО (краткое описание, ссылки на веб-сайт и др.).

Внедрение службы управления участием в ВО. Разработанная грид-служба управления участием в ВО была внедрена на сервере КНУ имени Тараса Шевченко, что позволило решить проблемы масштабируемости для Украинского национального грид-сегмента. Использование *PHP VOMS-Admin* как всеукраинской службы для УНГ стало толчком к развитию тематических ВО. В настоящее время сервер обслуживает 15 украинских ВО, количество которых возрастает без существенной нагрузки на ресурсы.

Окончание на стр. 74.

Готовые пакеты программного обеспечения *PHP VOMS-Admin* добавлены в центральные репозитории операционной системы семейства *Red Hat – Fedora*, а также в репозиторий *Extra Packages for Enterprise Linux* (*EPEL*), предназначенный для дистрибутивов *RHEL*, *CentOS* и *Scientific Linux*. Таким образом, на указанных операционных системах установку службы можно выполнить с помощью стандартного менеджера пакетов *YUM*.

Создан веб-сайт проекта, размещенный по адресу: *http://grid.org.ua/development/pva/*. Сайт содержит информацию о работе *PHP VOMS-Admin*, документацию по установке и конфигурации, исходные коды и программные пакеты.

Заключение. Проведен анализ работы службы членства в ВО в грид-инфраструктуре и показано существование проблем ее масштабируемости.

В результате анализа архитектуры *VOMS* и исследования использования аппаратных ресурсов выявлена проблемная компонента службы – *EDG VOMS-Admin*. Последняя использует значительное количество ОЗУ и процессорной мощности для обслуживания даже одной ВО. Увеличение количества ВО приводит к ошибкам работы с памятью.

Исследование времени обработки запросов *EDG VOMS-Admin* показали значительную дисперсию распределения результатов и ее увеличение с увеличением количества одновременных запросов, превышая значение в 1 с.

Для решения упомянутых проблем разработано альтернативное программное обеспечение для администрирования участия в ВО – *PHP VOMS-Admin*, избавленный от проблем масштабируемости по количеству обслуживаемых ВО и одновременных подключений к серверу. Результаты тестирования *PHP VOMS-Admin* показали незначительное увеличение использования памяти и стабильное время отклика в пределах 0,001−0,003 с даже при параллельных обращениях.

Наряду с решением проблем масштабируемости были реализованы дополнительные функции, среди которых поддержка многоязычного интерфейса, автоматизация процесса добавления к обслуживанию новых ВО и работа со списками внешних серверов.

Использование *PHP VOMS-Admin* для централизованного сервера службы членства в ВО в Украинской национальной грид-инфраструктуре стало толчком к развитию тематических ВО. В противовес трем ВО в настоящее время сервер *VOMS* Киевского национального университета имени Тараса Шевченко обслуживает 15 виртуальных организаций, основанных ведущими исследовательскими институтами НАН Украины и высшими учебными заведениями.