Список литературы:

1. Беседин Д.И., Боборыкин С.Н., Рыжиков С.С. Предотвращение утечки информации, хранящейся в накопителях на жестких магнитных дисках. Специальная техника. №1/2001.-с.49-56.
2. Коженевский С.Р. Причины потери информации с электронных накопителей.
3. Блиц – опрос. Сколько ПК продано в Украине в 2002 году. Компьютерное обозрение. Издательский дом ITC. С.48-58.
4. Коженевский С.Р. Методы визуализации магнитных полей носителей информации. Реєстрація, зберігання і обробка даних. 202, Т.4, №4, стр.48-60.
5. www.epos.kiev.ua
6. Нестерин В.А. Оборудование для импульсного намагничивания и контроля постоянных магнитов.-М.:Энергоатомиздат, 1986, 88с.
7. Болдырев А.И., Сталенков С.Е. Надежное стирание информации – миф или реальность? Защита информации. Конфидент. №/2001. с.38-46.
8. Болдырев А.И., Василевский И.В., Сталенков С.Е. Методические рекомендации по поиску и нейтрализации средств негласного съема информации. М.:ЗАО НПЦ Фирма «НЕЛК». 2001.-248с.
9. Рохманюк В.М., Фокин Е.М. Аппаратура экстренного уничтожения записей на магнитных носителях. БДИ, 2000 №5. с.4-21.
10. Рохманюк В.М., Фокин Е.М. Чисто? Чисто и быстро! Защита информации. Конфидент, 1998 №5. с.60-69.
11. Рохманюк В.М., Фокин Е.М. Способ стирания записей на магнитном носителе и устройство для его осуществления. Патент на изобретение RU № 2144223.
12. Рохманюк В.М., Фокин Е.М. Устройство для стирания записи на магнитном носителе. Свидетельство на полезную модель № 18796.
13. Кнопфель Г. Сверхсильные импульсные магнитные поля. Перевод с английского Николаева Ф.А., Свириденко Ю.П. М.: Мир. 1972. 214 с.

УДК 681.3.06

И.А.Терейковский

БЕЗОПАСНОСТЬ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, СОЗДАННОГО С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИЙ COM, DCOM, COM+

Введение

Важнейшей предпосылкой использования технологии COM и базирующихся на ней технологий DCOM и COM+ является создание повторно используемых компонентов, которые можно было бы связать в готовое приложение. Эта предпосылка вызвана как техническими, так и коммерческими причинами. Первоначально создание таких компонентов планировалось реализовать с помощью объектно-ориентированных языков программирования, которые определяют объекты на уровне исходных текстов программ, а повторное использование кода поддерживается посредством наследования. Серьезной преградой указанной реализации является необходимость знания программистом семантики базового класса, поскольку в его компетенцию входит перегрузка функций класса. Кроме этого, для взаимодействия компонентов, реализованных с использованием различных языков программирования, требуются промышленные стандарты, поддерживаемые множеством производителей. В качестве такого стандарта компанией Microsoft предлагается семейство технологий COM, DCOM и COM+ в которых готовые компоненты представлены в виде "черных ящиков", а использование компонентов осуществляется исключительно с помощью интерфейсов, предоставляемых пользователю. Отметим, что это семейство

базируется на технология COM. Наиболее важная характеристика этой технологии заключается в том, что она не является промежуточным программным обеспечением, а представляет собой бинарный стандарт, позволяющий взаимодействовать программам, созданным с применением различных языков программирования. Технология DCOM, которая появилась на свет вместе с операционной системой Windows NT 4.0, представляет собой расширение COM для использования в сети. Хотя применение DCOM увеличивает возможности механизма удаленного вызова процедур (RPC), изначально использовавшегося в COM, но базовая технология осталась при этом неизменной. Технологию COM+ можно рассматривать как следующее поколение компонентной архитектуры, разработанной фирмой Microsoft. Ее главными отличиями являются хорошая приспособленность к масштабированию, повышенная надежность при работе в сети и возможность автоматической поддержки транзакций. Технология COM+ базируется на декларативной модели, основанной на применении атрибутов. COM+ интегрирует сервер транзакций (MTS) в COM и использует очередь сообщений (MSMQ) при обеспечении вызовов. За счет этого существенно упрощается создание как серверных, так и клиентских приложений. Кроме этого, в технологии COM+, имеется множество сервисов, применение которых позволяет создавать высоко масштабируемые приложения. По отношению к COM, COM+ требует значительно большей конфигурационной информации для хранения всех атрибутов приложений и компонентов. Эта информация храниться в отдельной системной базе данных под названием RegDB. Доступ к ней осуществляется через семейство системных объектов, администрирование которых можно осуществлять программно или с помощью оснастки Component Services. Важной особенностью администрирования является возможность установки атрибутов на всех уровнях иерархии, т.е. на уровнях приложения, компонента, интерфейса и метода.

Отметим, что в настоящее время разработка программного обеспечения с использованием семейства технологий COM является общепринятой практикой. В некоторых случаях даже рекомендуется использовать технологию COM при построении программного обеспечения систем защиты информации [4]. При этом, рассмотрение системы безопасности программного обеспечения, созданного на базе семейства технологий COM на наш взгляд носит несколько односторонний характер. В большинстве работ [3,4,5,6] система безопасности рассматривается на концептуальном уровне, причем акцент ставиться на технологических аспектах авторизации и аутентификации компонентов COM. В то же время критический анализ защищенности различных сервисов, консолей управления, а также политики безопасности проведен не достаточно полно.

Целью данной статьи является – проведение комплексной оценки системы безопасности семейства технологий COM.

Концептуальная модель системы безопасности COM/DCOM.

Система безопасности COM/DCOM/COM+ (рис.1) базируется на системе безопасности Windows NT и в первую очередь ориентирована на защиту распределенных приложений.

Контролируется две основные функции безопасности – авторизация и аутентификация, при этом можно контролировать запуск компонента, доступ к уже запущенному компоненту, определить подлинность работающего компонента сервера и выполняет ряд других второстепенных задач. Система безопасности реализована двумя слоями. Верхний слой – аутентифицированный RPC, обеспечивает безопасность соединений. Нижний слой создается с помощью провайдера поддержки безопасности (Security Support Provider – SSP), который реализует интерфейс поддержки безопасности (SSPI), вызываемый аутентифицированным RPC. В нижнем слое могут использоваться различные провайдеры. Так в Windows 2000 используется Kerberos, который является провайдером поддержки безопасности по умолчанию. Используя эти, слои COM предоставляет систему безопасности в соответствии с пожеланиями администратора, зафиксированными в системном реестре. Отметим, что как RPC, так и Kerberos ориентированы во основном на защиту информации от не санкционированного просмотра и на обеспечение целостности данных. Кроме этого

целостность данных поддерживается также за счет механизма распределенных транзакций. Можно сформулировать вывод о том, что с точки зрения защиты информации от несанкционированного просмотра и изменения концепция системы безопасности COM несколько совершеннее концепции системы безопасности Windows NT, в основном за счет более высокого уровня абстрагирования.

Доступность данных обеспечивается за счет целого ряда дополнительных механизмов, назначением которых является упорядочивание доступа к разделяемым данным, оптимизация сетевого трафика, а также уменьшение нагрузки на сервер COM+, как путем экономии вычислительных ресурсов, так и путем кластеризации. Отметим, что перечисленные механизмы явным образом не входят в систему безопасности семейства технологий COM.

Анализ концептуальной модели показывает, что с точки зрения обеспечения целостности данных и защиты от несанкционированного просмотра система безопасности приложений COM является достаточно совершенной. Защита имеет иерархическую структуру, на каждом уровне которой используются комплексные технологии. В то же время, доступность данных обеспечивается за счет нескольких, идеологически не связанных между собой технологий. Это обстоятельство указывает на ее скорее фрагментарный, а не комплексный характер.

С позиций [1,2] рассмотрим технологические аспекты системы безопасности семейства технологий COM, не затрагивая при этом систему безопасности операционной системы.

Рис. 1 Концептуальная модель системы безопасности семейства технологий COM

Администрирование системы безопасности COM.

Механизм администрирования в основном предназначен для управления разграничением доступа к компонентам приложений. Установки безопасности COM хранятся в системном реестре, а для их изменения используется инструмент DCOMCNFG операционной системы Windows 2000 Server. С точки зрения авторизации DCOMCNFG позволяет контролировать как запуск компонента так и доступ к уже запущенному компоненту. Права на запуск сервера DCOM по умолчанию определяются в разделе "Запуск по умолчанию" (Default Access Permissions). Можно редактировать список пользователей и групп, имеющих разрешение на запуск DCOM сервера, у которого не определены его собственные индивидуальные настройки. Для каждого пользователя или группы пользователей можно разрешить или запретить запуск сервера. При этом, как и для системы безопасности Windows запрет имеет преимущество перед разрешением. Такой же список имеется и для раздела "Доступ по умолчанию" (Default Launch Permissions). С точки зрения

COM/DCOM запуск и доступ представляют разные действия. Пользователь может иметь право обратиться к работающему серверу, но не иметь право запустить его. Кроме этого имеется возможность изменения прав пользователей на запуск и доступ к отдельным приложениям.

COM/DCOM обеспечивает шесть уровней аутентификации пользователей, настройка которых производится администратором в разделе Default Properties.

1. None. Аутентификация отсутствует. В этом случае отсутствует авторизация, а следовательно, и система безопасности в целом. Рекомендуется никогда не использовать на практике эту установку.
2. Connect. Аутентификация выполняется при подключении к серверу.
3. Call. Аутентификация выполняется при принятии сервером вызова RPC.
4. Packet. Аутентификация того, что данные приходят от определенного клиента, выполняются по приходу каждого пакета (в случае протоколов типа UDP подключение и вызов интерпретируются как пакет).
5. Packet Integrity. выполняется аутентификация того, что пакет приходит от определенного клиента, и что данные не были модифицированы.
6. Privacy. Выполняются все проверки Packet Integrity, кроме этого данные в пакетах шифруются.

Установка прав доступа сервера к различным ресурсам компьютера выполняется за счет изменения идентификатора пользователя. Для этого необходимо перейти в раздел Identity при работе с определенным приложением. Доступны следующие идентификаторы – запускающий (Launching User), интерактивный (The interactive user) и конкретный пользователь (The user).

Установкой по умолчанию является Launching User. В этом случае процесс сервера будет иметь права пользователя, от которого поступил запрос. К недостаткам этой установки относятся:

- Создание для каждого клиента своей весьма ресурсоемкой станции Windows.
- Сервер играет роль пользователя и поэтому имеет небольшие возможности при обращения к разным ресурсам.
- Процессу сервера не доступен интерактивный десктоп.

Выбор установки The interactive user приводит к тому, сервер запускается с признаками доступа интерактивного пользователя, в настоящий момент зарегистрированного на компьютере. При этом десктоп оказывается доступен, и сервис может выводить окна. Дополнительные станции Windows для клиентов не создаются, не происходит и подмены пользователя – процесс может все, что может текущий пользователь системы. Недостатки этой установки проявляются, когда в системе не работает ни один пользователь. В этом случае процесс сервера может не запуститься вообще, а работающий сервис прекратит свою деятельность при выходе пользователя из системы. Запуск в качестве интерактивного пользователя рекомендуется использовать только на стадии разработки [5].

Установка The user, позволяет указать конкретного пользователя, под именем которого будет работать сервер. При этом будет создана одна станция Windows для всех клиентов. Подмены пользователя не происходит, процесс может делать все то, что разрешено пользователю, под именем которого он работает. Сервер может работать при отсутствии на компьютере активного пользователя.

Представляет интерес возможность подмены пользователя в процессе работы. Такая возможность может быть очень полезна во многоуровневых приложениях, где конечный компонент имеет собственную систему безопасности и выполняет проверку подлинности вызывающего клиента. Если компонент среднего уровня работает под определенной учетной записью, такая проверка завершиться неудачно. Решение состоит в работе процесса среднего уровня от имени клиента. Отметим, что решение работать от имени пользователя не подходит для больших систем, требующих масштабирования. Причиной этому является нецелесообразность расходов на создание станции Windows для каждого отдельного

пользователя. Поэтому в DCOM эта опция запрещена. Однако COM позволяет компоненту подменять любого пользователя. В этом случае имеется возможность выполнения приложением среднего уровня всего того, что может выполнить конечный клиент, включая вызов других уровней, которые в свою очередь, могут использовать подмену пользователя. Уровень подмены (имперсонации) по умолчанию может быть установлен в разделе Default Properties. Доступны три уровня имперсонаци:

– Identity – сервер может подменять клиента с целью проверки прав доступа на запуск, но не получает доступ к объектам системного уровня.

– Impersonate – сервер может подменять клиента с целью проверки прав доступа на запуск, получает доступ к объектам системного уровня, но не может подменить клиент при вызове другого (удаленного) сервера.

– Delegate – сервер может подменять клиента с целью проверки прав доступа на запуск, получает доступ к объектам системного уровня и может подменить клиента при вызове другого (удаленного) сервера.

Особенности системы безопасности COM+.

COM+ добавляет к системе безопасности COM множество новых атрибутов, устанавливаемых с помощью инструмента администрирования COM+Explorer (системного сервиса Windows Service Control Manager) или программно. За счет применения указанных атрибутов, относительно COM, настройки системы безопасности COM+ более гибкие. Если в COM настройки безопасности ограничивались уровнем компонентов, то в COM+ возможно менять указанные настройки на уровне приложения, компонента, интерфейса и метода.

На уровне приложения, возможно, определить авторизацию, уровень безопасности, уровень аутентификации и уровень имперсонации. Опция авторизации реализует основной этап проверки доступа на уровне приложения. Если указанная опция отключена, то проверка доступа не осуществляется. При этом компонент получает права администратора. Опция уровня безопасности определяет, должна ли информация о безопасности включаться в контекст объекта. При настройках по умолчанию эта информация включается в контекст. Настройки уровня аутентификации и уровня имперсонации имеют то же значение, что и в COM.

Кроме этого в COM+ введено новая абстракция безопасности, называемая – ролью. Роль представляет собой логическую группу пользователей, авторизированных для доступа к компоненту, интерфейсу или методу. Система безопасности, основанная на ролях, является компонентом Microsoft Transaction Server и предназначена для упрощения настройки и программирования этой системы. Политика системы безопасности при этом определяется назначения пользователей ролям, а компоненты определяют права доступа ролей. Роли также могут быть запрограммированы с использованием контекста вызова. Определение ролей заключается в назначении им, с помощью COM+Explorer, уникальных имен, включения в них отдельных пользователей и групп пользователей. После этого производится настройка доступа ролей на всех необходимых уровнях.

Особенности программной конфигурация системы безопасности COM+.

Возможности программной конфигурации практически полностью идентичны с административными возможностями. Программная конфигурация осуществляется посредством использования методов соответствующих интерфейсов. Особенностью программирования серверных компонентов является не желательность использования в них окон сообщений. Это связано с тем, что клиент может инициализировать появление такого окна, которое выводиться не на его рабочей станции, а следовательно не может быть закрыто. В некоторых случаях это может привести к зависанию клиента.

Недостатки системы администрирования

Анализ системы администрирования семейства технологий COM указывает на ее хорошую продуманность и организацию, как с точки зрения эффективности выполняемых функций, так и с точки зрения простоты использования. Однако на наш взгляд в ней присутствует существенный недостаток. На практике вся система безопасности замыкается

на одной группе администраторов операционной системы, имеющих одинаковые права. Не существует встроенного ни в операционную систему Windows, ни в систему COM/DCOM/COM+, механизма четкого разделения прав администраторов с возможностью взаимного контроля и дублирования.

Общий подход к обеспечению синхронизации и доступности разделяемых ресурсов

При использовании семейства технологий COM упорядочение доступа к разделяемым ресурсам может осуществляется:

– Явно – с помощью соответствующего кода прикладной программы, использующего взаимоисключающие примитивы типа критических частей кода.

– Автоматически – с помощью системы

COM использует автоматическое упорядочивание, присущее обработке сообщений Windows, при маршалинге методов через границы процессов. Маршалинг реализуется посредством обработки сообщений в скрытом окне. Следовательно, один вызов метода COM-объекта будет полностью завершен до того, как начнет выполнятся другой метод. При использовании COM+ реализация согласования обращений к ресурсам существенно упрощается реализовано за счет применения специальных атрибутов "требуется синхронизация". При этом в процесс выполнения вызова может вмешаться перехватчик COM+ и применить блокировку для обеспечения синхронизации. Тем самым работа программиста значительно упрощается. Недостатком автоматического упорядочивания к разделяемым данным является возможность запрещения любого доступа к разделяемым данным, даже в том случае, когда один из методов обращается только к части данных.

Использование технологий DCOM и COM+ предопределяет необходимость оптимизации использования таких сетевых ресурсов как объем сетевого трафика и вычислительных возможностей сервера. Общий аспект оптимизации – это получение клиентом указателей на интерфейсы сервера. Для получения нескольких указателей путем одного обращения по сети программистом, возможно, использовать функцию CoCreateInstanceEx. Кроме того DCOM кэширует все вызовы Release клиента и реально не пересылает вызовы Release по сети до тех пор, пока клиент не доведет значение счетчика до нуля. За счет этого несколько оптимизируется загрузка компьютера-сервера при удалении/создании объекта. Еще одним решением для экономии вычислительных ресурсов при создании сетевого соединения является пул соединений. Его применение целесообразно при многократных обращениях к одним и тем же серверам.

Для того, что бы из-за неправильной работы клиента, или из-за сбоев в сети на сервере не оставалось не освобожденных объектов клиент периодически обращается к серверу, давая знать, что он находится в работоспособном состоянии. Если три таких последовательных обращения пропущены, система считает, что клиент завершил работу и освобождает объект вместо клиента. Трафик, связанный с указанными извещениями невелик. На наш взгляд данная схема требует усовершенствования. Злоумышленник, многократно и кратковременно нарушая работоспособность линий связи или удаляя ответы клиента, может существенно загрузить сервер выполнением запросов на создание/удаление объектов.

Исчерпание вычислительных ресурсов сервера может произойти также из-за того, что механизм реализации фабрики классов не предусматривает ограничения количества клиентов, получивших соответствующие указатели. По этой причине злоумышленник может достаточно эффективно организовать атаку на отказ в обслуживании сервера, используя небольшое число компьютеров. К возможным защитным мероприятиям следует отнести ограничение числа клиентов, выполняемых на одном компьютере, или в одном сегменте сети.

Обеспечение синхронизации и доступности для многопоточных приложений

Стандартным решением COM в области многопоточности является применение так называемых апартаментов и концепции "потоковой модели класса". Апартаменты группируют объекты, которые разделяют одни и те же требования к многопоточности. Выделяют одно и многопоточные апартаменты. В однопоточных апартаментах (STA) может

выполнятся только один поток, а в многопоточных (MTA) могут параллельно выполняться несколько потоков. В обязанности сервера входит определение типа апартаментов, в котором может выполняться объект. Применение STA приводит к упорядочиванию обращения к любому объекту, размещенному в ней, за счет использования очереди сообщений Windows. При этом параллельного доступа, а значит и параллельность выполнения потоков теряются. При этом клиент из одного потока одного апартамента может обращаться к объекту из другого потока другого апартамента. Это обращение реализуется посредством прокси, который упорядочивает обращения к объекту за счет блокировки параллельных обращений.

Модель потоков MTA не предпринимает никаких действий по упорядочиванию доступа. Еще одна модель потоков "Both" позволяет создать COM-объект в любом апартамента. При этом клиент их STA может вызвать объект непосредственно, без прокси. Следовательно, в двух последних случаях, код реализующий методы класса должен быть безопасным с точки зрения параллельности обращений. Таким образом, выигрыш в производительности достигается за счет потенциальной потери безопасности. Еще одной слабой стороной использования апартаментов MTA и STA является снижение производительности за счет того, что вызов пришедший извне вызовет обмен потоков.

В COM+ предложены так называемые нейтральные апартаменты, применение которых который не вызывает обмен потоков. Объект в нейтральном апартаменте всегда работает в потоке, вызвавшем его. Нейтральные апартаменты не обеспечивают поддержки синхронизации, обеспечивая возможность работы механизма синхронизации основанного на активности и не привносящего накладных расходов. Активность предоставляет собой абстракцию, обозначающую группу контекстов, в которой не разрешена параллельная работа. Для определения защиты объекта системой времени выполнения COM+ при параллельных вычислениях в нем должен быть объявлен атрибут синхронизации. защита будет выполнена облегченным перехватчиком, вместо переключения потоков. Таким образом, уменьшение производительности может произойти только, если возможен безопасный одновременный доступ разных клиентов к разным частям объекта. Проведенный анализ показывает, что концептуально стандартные решения COM/COM+ для обеспечения многопоточности, достаточно хорошо продуманы и просты в реализации. Вместе с тем их реализация в некоторых случаях может серьезно уменьшить эффективность функционирования приложения.

Обеспечение доступности вычислительных ресурсов при увеличении масштаба приложения

Возможность масштабирования является один из важнейших вопросов при разработке приложений уровня предприятия или приложений предназначенных для использования в глобальных сетях есть. Практический опыт разработки прикладных программ, а также анализ [3,6] позволил построить, представленную на рис. 2, структуру направлений обеспечения доступности вычислительных ресурсов, связанных с увеличением масштабов приложения COM+. Наиболее простым путем увеличения масштабируемости является уменьшение нагрузки на сервер COM. Кроме рассмотренного выше пула подключений для этого используется активизация по необходимости. Применение активизации по необходимости позволяет клиенту поддерживать ссылки на объект, который деактивирован или даже удален сервером. Когда клиент вызывает метод с применением этой ссылки, сервер создает или активирует этот объект. Таким образом, экономятся ресурсы сервера на хранение объектов. Еще одно решение заключается в применении пула экземпляров объектов. Решение целесообразно в случае, если активизация по необходимости требует высоких накладных расходов на создание и инициализацию объектов. Однако такое решение накладывает существенные требования к программной реализации объекта – отсутствие запоминания состояния, отсутствие привязки к потоку и не возможностью автоматизации работы с транзакциями.

Такие возможности COM+ по увеличению масштабируемости как активизация по

необходимости, пул подключений и пул объектов позволяют несколько уменьшить нагрузку на вычислительные мощности компьютерной системы, но не являются решением для общего случая. Выходом из сложившейся ситуации может стать применение кластеризации, т.е. распределение нагрузки между несколькими узлами – кластерами. Отметим, что технология DCOM позволяет распределить нагрузку между несколькими серверами, без привлечения дополнительного программного обеспечения, за счет назначения различным клиентам разных серверов. Однако такое решение является слишком трудным с точки зрения администрирования. Microsoft предлагает несколько программных продуктов, связанных с применением технологий COM/DCOM/COM+, в области кластеризации. Одним из них является Component Load Balancing (CLB), предназначенный для прозрачного с точки зрения клиента, распределения баланса загрузки между серверами COM+.

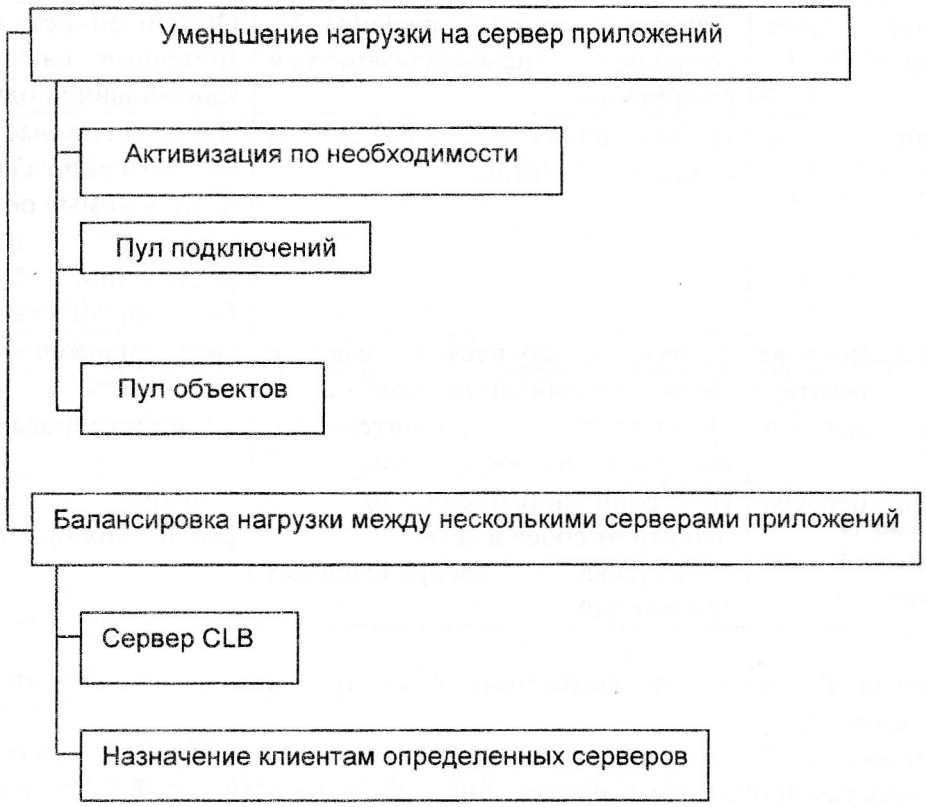

Рис. 2 Структура направлений увеличения масштаба приложений COM+

Принцип действия сервера CLB заключается в определении какой из серверов COM+ наименее загружен и передачи ему запроса, пришедшего на обработку. Суть алгоритма определения загрузки сводится к тому, что CLB периодически опрашивает серверы приложений и строит список, упорядоченный по их времени отклика. После этого направляет запрос на выполнение очередному серверу из списка. Практика показывает, что кроме балансировки нагрузки CLB позволяет обеспечить защиту приложений от сбоев. Однако его применение вызывает некоторые трудности.

Приложения, которые предназначены для использования в системах с балансировкой загрузки, должны быть оптимизированы по критерию размера заданий. В случае использования очень больших заданий сложно обеспечить требуемый уровень распределенной работы, а при очень маленьких балансировка сопровождается большими накладными расходами по их маршрутиризации. Также следует учитывать, что сервер CLB не позволяет запомнить компонентам COM+ свое состояние. Поэтому компонент не должен быть привязан к определенному компьютеру, логика программы должна быть способна

обрабатывать перемещение объекта, состояние объекта не должно храниться во временном файле на сервере приложения. Кроме этого, из-за того, что пул объектов храниться на конкретном компьютере, его использовать становиться не возможным. Отметим, что указанные программные особенности во многих случаях могут серьезно усложнить разработку COM+ компонентов. В табл.1 представлены основные достоинства и недостатки рассмотренных путей увеличения масштабов приложеий COM+.

Таблица.1 Оценка путей увеличения масштабов приложений COM+

Направление	Достоинства	Недостатки
Уменьшение нагрузки на сервер приложений		
Активизация по необходимости	Низкие накладные расходы при хранении ссылок не используемые объекты.	Ограниченность масштаба. Дополнительные расходы на создание объектов.
Использование пула подключений	Низкие накладные расходы на создание использовавшегося соединения	Ограниченность масштаба. Дополнительные расходы на запоминание подключения.
Использование пула объектов	Низкие накладные расходы на создание объектов.	Дополнительные расходы на хранение ссылок на не используемые объекты. Сложность программной реализации. Ограниченность масштаба.
Балансировка нагрузки между несколькими серверами приложений		
Назначение клиентам определенных серверов	Не ограниченный масштаб. Отсутствие дополнительного программного обеспечения	Сложность администрирования.
Использование сервера CLB	Не ограниченный масштаб. Защита от сбоев в сети. Поддержка распределенных транзакций.	Сложная программная реализация приложений.

Проблемы безопасности связанные с программными ошибками и наличием "троянских коней"

Означенные проблемы приобретают особую актуальность в следствии априорной закрытости программного кода и применения семейства технологий COM в компьютерных сетях.

В первую очередь необходимо отметить возможность исчерпания вычислительных ресурсов сервера за счет не удаления COM объекта. Причинами этого обстоятельства могут быть преднамеренные или не преднамеренные ошибки при подсчете указателей на интерфейсы, а также в стандартном методе Release интерфейса IUnknow, который поддерживается каждым компонентом объекта. Подобные ошибки могут возникнуть при реализации фабрики классов. Еще одна потенциальная опасность технологии заключается в том, что при определении сервера клиент может решить, использовать хранящуюся в системном реестре информацию о размещении сервера, или определить его положение самостоятельно. Таким образом, последствия действий клиента являются не предсказуемыми и не безопасными. Кроме этого, клиент, работающий на стандартных портах COM, может быть запущен от имени пользователя с соответствующими полномочиями, а, следовательно, может несанкционированно осуществить прием/передачу некоторой информации. Отметим, что устранение перечисленных выше проблем может быть устранено путем тщательного тестирования программного обеспечения, наличия подробной документации на интерфейсы и методы объектов, вывода пользователю информации о произошедших ошибках и исключительных ситуациях, наличия в программном обеспечении журнала ошибок и

исключительных ситуаций, а также ведения грамотной политики безопасности всей компьютерной системы.

Сложной проблемой безопасности является наличие не документированных возможностей сервера, за счет наличия в нем скрытых интерфейсов. В стандартном случае идентификаторы серверов, которые хранятся в системном реестре, обеспечивают поиск необходимой информации о сервере. Кроме этого в системном реестре есть ключ Interfaces, в котором хранятся идентификаторы интерфейсов. С помощью этих идентификаторов интерфейсы сервера становятся доступными как запросам клиентов, так и для просмотра администратором. Информацию о интерфейсах можно просмотреть, например с помощью программы OLE/COM Object Viewer, которая входит в пакет MC Visual Studio. Однако COM не предоставляет механизм для непосредственного поиска интерфейсов, поддерживаемых классом. Другими словами, нельзя запросить у объекта список поддерживаемых им интерфейсов. Кроме системного реестра список интерфейсов можно найти в библиотеке типов класса. Однако разработчик может некоторые интерфейсы там не показывать. При этом клиент хотя и с определенными ограничениями, но все таки может использовать интерфейс не представленный в системном реестре и библиотеке типа. Идентификатор интерфейса не нужен если не использовать фабрику классов. Это случай создания локального объекта, например с помощью функции COM API CoCreateInstance, которая позволяет создать единичный экземпляр объекта COM и возвратить указатель на интерфейс. Возможный путь получения всего списка интерфейсов методом перебора, с помощью стандартного метода QueryInterfase стандартного интерфейса IUnknow весьма затруднителен. Ведь для получения указателя на интерфейс в метод QueryInterfase необходимо передать идентификатор интерфейса (GUID). Из-за того, что GUID является 128-битовым числом подбор возможных интерфейсов методом перебора практически не возможен. Таким образом, априорное наличие программных ошибок и особенно "троянских коней" может серьезно нарушить безопасность программного обеспечения, разработанного с использованием технологии COM.

Отдельной проблемой систем COM/DCOM/COM+, т. е. систем построенных на использовании компонентов является проблемы связанные с качеством программирования, в основном за счет трудностей при тестировании системы в целом. Это предопределяет необходимость существования целостной концептуальной модели проектируемой системы, что в свою очередь идет несколько в разрез с самой идеей компонентного программирования.

Обеспечение транзакций

Базируется на применении концептуальной модели X/Open DTP согласно которой приложения для координации транзакций работают с менеджером транзакций. В модели реализован протокол OLE Transaction. Кроме этого в COM+ используется концепция автоматических транзакций, для которых транзакционное поведение компонентов достигается за счет настройки их атрибутов. Реализована трехуровневая архитектура. Клиентское приложение не обновляет ресурсы непосредственно, а работает через компонент среднего слоя, который может быть сконфигурирован в COM+ с атрибутами транзакций. При этом во время работы посредством стандартной модели перехвата COM+ может установить распределенную транзакцию от имени компонента. Отметим, что транзакционный компонент, должен быть импортирован в приложение COM+ и сконфигурирован как поддерживающий или требующий транзакцию. Если экземпляр объекта участвует в транзакции, ресурсы с которыми работает объект, будут защищены транзакцией. Транзакционные компоненты могут либо разделять контекст вызывающей программы, либо работать в собственном контексте. Программирование транзакции в общем случае требует минимального количества программного кода, а в большинстве случаев для реализации транзакции достаточно проведения соответствующих настроек. Практика подтверждает надежность и эффективность рассмотренного механизма обеспечения транзакций, что, безусловно, является положительной стороной технологии

COM+.

Живучесть программного обеспечения при отказах сети.

При разработке распределенных приложений с помощью семейства технологий COM возможно применение таких протоколов передачи данных как DCOM, HTTP и MSMQ. Общей чертой первых двух протоколов является их синхронность. Когда клиент вызывает метод, возврата из вызова не происходит до получения и обработки обратного сообщения. При этом в случае сбоя в сети сообщение либо посылается еще несколько раз, после чего может быть потеряно. По этой причине живучесть протоколов DCOM и HTTP не достаточна. Специфическим средством технологии COM+ для обеспечения живучести является применение асинхронного протокола передачи сообщений MSMQ, который базируется на использовании очереди сообщений. Рассмотрим алгоритм протокола MSMQ. Клиент отсылает запрос. MSMQ помещает сообщение в очередь и возвращает управление клиенту, который может не дожидаться ответа от сервера. Сервер считывает сообщение из очереди и в предусмотренном случае может послать ответ клиенту. Наличие очереди сообщений позволяет не только уменьшить время отклика клиента, но и позволяют делать запросы, даже не будучи подсоединенным, к серверу. Запросы размещаются в очереди клиента и в случае соединения с сервером, система выполнения очереди передаст запрос в очередь сервера. Кроме всего прочего отправленное сообщение может быть прочитано несколькими серверами. Таким образом, протокол MSMQ обеспечивает повышенную устойчивость к отказам сети.

Очереди хранятся в зависимости от типа. Быстрая очередь, для повышения производительности, храниться в оперативной памяти. Восстанавливаемая очередь храниться на жестком диске. В сети Windows 2000 для этого применяется сервисы активного каталога. Очереди сообщений с надежными методами их хранения могут участвовать в распределенных транзакциях. Для автоматизации работы с использованием MSMQ, COM+ предоставляет сервис QC. Используя данный сервис можно вызвать компонент обычным для COM+ образом, но вызов будем реализован посредством MSMQ, а не RPC. При этом для каждого пользователя требуется внутренний сертификат безопасности MSMQ. Для получения указанного сертификата используется соответствующая оснастка (аплет Message Queuing). Несмотря на указанные достоинства, протокол MSMQ применяется, ограничено из-за того, что его использование возможно только на платформах Microsoft.

Выводы

Потенциально система безопасности семейства технологий COM/DCOM/COM+ позволяет обеспечить полноценную защиту приложений. Однако применение этой технологии при разработке высокоответственных приложений ограничено следующими недостатками.

- Отсутствием на концептуальном уровне комплексного подхода к обеспечению доступности данных.
- Отсутствием в системе безопасности механизма четкого разделения прав администраторов с возможностью взаимного контроля и дублирования.
- Во многих случаях автоматическая синхронизация доступа к разделяемым данным для многопоточных приложений или при маршалинге может серьезно уменьшить доступность данных.
- Протокол освобождения сервера приложений уязвим с точки зрения "атаки на отказ".
- Применение сервера балансировки загрузки накладывает дополнительные ограничения на программную реализацию приложений.
- Возможное исчерпание ресурсов сервера за счет не удаления COM+ объекта по причине программной ошибки или "троянского коня".
- Возможность существования недокументированных возможностей сервера за счет скрытых интерфейсов.

– Сложность полноценного тестирования приложений.

– Применение асинхронного протокола MSMQ, который обеспечивает живучесть приложений COM+ при отказах в сети, возможно только на платформах Microsoft.

Перспективные пути исследований

– Создание концепции обеспечения доступности данных для программного обеспечения созданного на основании семейства технологий COM.

– Создание механизмов взаимного контроля, дублирования и разделения полномочий администраторов системы безопасности COM.

– Доработка стандарта COM в направлении не допущения скрытых интерфейсов.

Список итературы:

1. Браіловський М.М., Лазарєв Г.П., Дорошко В.О. Захист інформації у банківській діяльності. – К.: ТОВ "ПоліграфКонсалтинг", 2004.–216 с.

2. Герасименко В. А. Основы защиты информации. – М.: Изд-во «Инкомбук», 1997. - 537 с.

3. Грельсен Э. Модель COM и применение ATL 3.0: Пер. с англ. — СПб.: BHV — Санкт-Петербург, 2000. — 928 с.

4. Мелкумян К.В. COM как средство для реализации достоверной вычислительной базы. Защита информации Сб.н..тр. К Киевский международный университет гражданской авиации 1999 с.104-106

5. Оберг Р. Технология COM+. Основы и программироваие. – М.: Изд. "Вильямс", 2001. – 480 с.

6. Хильер С. Создание приложений COM+ в среде Visual Basic. Руководство разработчика. М.: Изд. "Вильямс", 2001. – 416 с.

УДК 354.31(477)(004.415+004.658.2)

В.А.Кудінов

АНАЛІЗ ДИНАМІКИ ОПЕРАТИВНОЇ ОБСТАНОВКИ В КРАЇНІ З ВИКОРИСТАННЯМ МОЖЛИВОСТЕЙ СИСТЕМИ ОПЕРАТИВНОГО ІНФОРМУВАННЯ МВС УКРАЇНИ

Вступ

В органах внутрішніх справ (ОВС) України створена та ефективно функціонує система оперативного інформування (СОІ) Міністерства внутрішніх справ (МВС) України про резонансні злочини та інші надзвичайні події, які були скоєні на території країни. Її головна мета – це своєчасне інформування керівництва міністерства, зацікавлених інстанцій, держави про реальний стан оперативної обстановки в країні, динаміку злочинності в цілому в Україні та окремих її регіонах для прийняття впливових управлінських рішень на її покращання. Своєчасність проходження оперативної інформації, її повнота та достовірність забезпечує негайне реагування на негативний розвиток подій, дозволяє їх попередити або локалізувати, у тому числі із залученням можливостей інших правоохоронних органів, військових формувань та спеціальних служб.

Суттєва ознака СОІ – це надання можливості оперативним службам отримати інформацію вже з моменту прийняття ОВС повідомлення (заяви) про злочин, що необхідно для його розкриття по "гарячих слідах". Усі ж інші існуючі бази даних в ОВС починають своє формування з моменту порушення кримінальної справи, надходження офіційних карток тощо, тобто оперативні служби отримують необхідну інформацію із запізненням у 10-30 діб. Крім цього, система оперативного інформування забезпечує постійне стеження за станом розкриття резонансних злочинів та ліквідацією наслідків інших надзвичайних подій.