УДК 004.7.052:004.414.2

Савченко А.С., к.т.н.

СИСТЕМНЫЙ АНАЛИЗ ПРОТОКОЛОВ УПРАВЛЕНИЯ КРУПНОЙ КОРПОРАТИВНОЙ СЕТЬЮ

Национальный авиационный университет

Проведен системный анализ протоколов управления с точки зрения их функционирования в крупных корпоративных сетях. Рассмотрены возможности как автоматического мониторинга и управления, так и осуществление данных функций при участии администратора сети.

Введение

Особенностью крупных корпоративных сетей, которую необходимо учитывать при разработке системы управленич (СУ), является значительная географическая распределенность. Как правило, корпоративная сеть крупной компании имеет четкую иерархическую структуру, т.е. состоит из сетей отдельных филиалов, в состав которых, в свою очередь, могут входить несколько предприятий со своей сетевой структурой. Кроме того, предприятия могут находиться в отдаленных населенных пунктах, где нет возможности доступа к высококачественным скоростным линиям связи.

Для управления крупными корпоративными сетями, включающими большое количество активного оборудования, необходимы сложные системы управления, осуществляющие мониторинг, контроль и управление каждым элементом и состоянием компьютерной сети в целом.

Учитывая ваышеизложенное, СУ корпоративной сетью следует проектировать таким образом, чтобы служебный трафик не занимал значительную часть полосы пропускания.

Кроме того, одним из фундаментальных требований к СУ сетью любого масштаба является простота внедрения, эксплуатации и модернизации. Поэтому СУ должна разрабатываться на базе существующих и хорошо апробированных технологий и протоколов.

Постановка задачи

Существует три принципиальных подхода к организации управления сложными сетями: централизованный, децентрализованный и распределенный [1].

Централизованное управление осуществляется из единого центра, в который стекается вся информация управления от всех управляемых объектов. Основным преимуществом такого подхода является возможность концентрации всей информации о состоянии сети в одном узле управления, что позволят создать целостную картину состояния и, соответственно, обеспечить непротиворечивость принимаемых решений. Однако, применительно к крупным корпоративным сетям, преимущества такого подхода становятся его недостатками. Значительный объём обрабатываемой информации требует высокопроизводительных серверов и высокой квалификации администраторов, а сама система становится уязвима. Кроме того, такое решение приводит к снижению полезной пропускной способности каналов за счет передачи служебного трафика, а значительная задержка информации не позволяет принимать адекватные решения.

Децентрализованное управление характеризуется отсутствием единого центра, а функции перераспределяются между множеством систем управления сетью. Так повышается живучесть СУ, отпадает необходимость в высокопроизводительных серверах и снижаются объём и задержка служебного трафика. Проблемой такого подхода является сложность разграничения «зон ответственности», а отсутствие целостной картины состояния сети приводит к противоречивым решениям.

При распределенном подходе к построению СУ используются иерархические архитектуры. В таком случае часть функций по управлению выполняется в автономных сегментах, остальные сосредоточены в едином центре. Это позволяет устранить основные недостатки централизованных систем (масштабируемость, объем и задержка служебного трафика) и децентрализованных (отсутствие целостной картины о состоянии сети). Недостатком остается слабая отказоустойчивость и требовательность к аппаратуре. Основная обработка управляющих воздействий производится в узлах управления, что также требует дополнительных ресурсов.

Поскольку централизованное и децентрализованное управление может оказаться малоэффективным (или даже иметь негативное действие) по вышеперечисленным причинам, а также учитывая особенности функционирования крупных корпоративных сетей, целесообразно строить СУ на базе распределенного подхода.

В таком случае задачу управление корпоративной сетью крупного географически распределенного предприятия целесообразно разделить на частные подзадачи управления автономными сегментами (АС). Такой подход позволит сократить задержки управляющей информации, предотвратить перегрузку всей сети управляющим трафиком, а также избежать расходов на дорогостоящие высокопроизводительные серверы централизованного управления в реальном времени. Нахождение оптимального разграничения как функций центральной части СУ и функций управления автономным сегментом, так и собственно разделение на автономные сегменты является важным, поскольку напрямую влияет на эффективность управления.

Основой СУ являются протоколы, которые должны быть достаточно простыми, но эффективно справляться с возложенными на них функциями. Таким образом, задача анализа и обоснованного выбора протоколов мониторинга и управления для СУ крупной корпоративной сетью является актуальной.

Системный анализ протоколов мониторинга и управления корпоративной сетью

Рассмотрим общую задачу управления компьютерной сетью, состоящей из нескольких автономных сегментов. Обобщенная структурная схема системы управления приведена на рис. 1 [2].

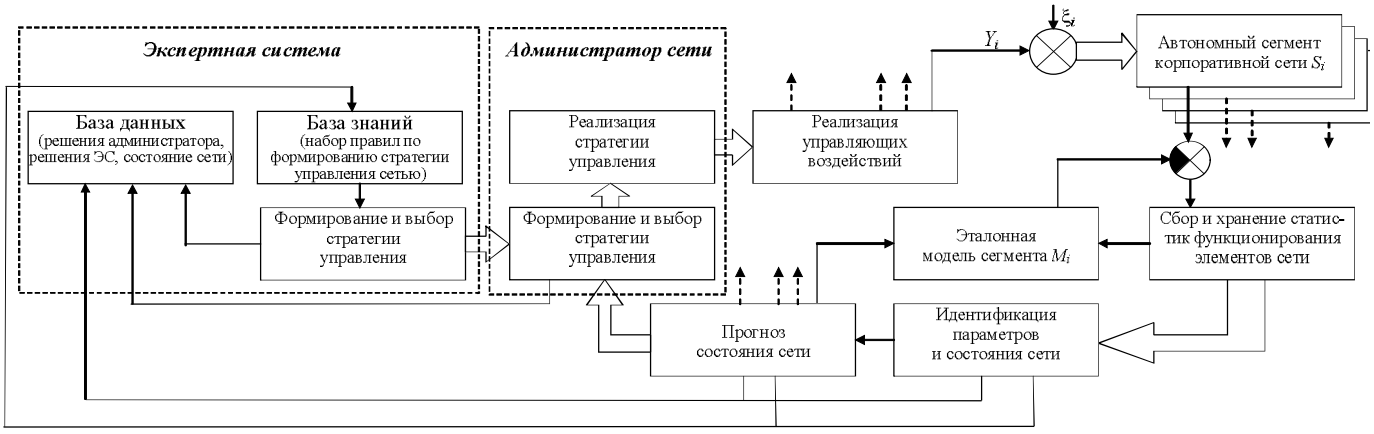

Рис. 1. Концептуальная схема системы управления корпоративной сетью

Корпоративную сеть можно представить множеством $M=\{W, C, A\}$, где M – множество объектов управления; W – множество рабочих станций; C – множество серверов; A – множество активных телекоммуникационных устройств.

Типичные задачи для системы управления удаленными станциями и серверами это – управление конфигурацией, распространение системного и прикладного ПО, инвентаризация аппаратных и программных ресурсов, удаленный контроль, мониторинг и управление в реальном режиме времени [1, 2].

Исходя из представленной схемы, основными задачами СУ корпоративной сетью являются:

1) идентификация состояний сетевого оборудования;

2) выработка управляющих воздействий;

3) реализация управляющих воздействий на объект управления.

Учитывая иерархический принцип построения корпоративной сети, оптимальнім является использование иерархической схемы сбора и обработки информации – «менеджер-агент-управляемый объект». Данные о состоянии объектов управления заносятся в соответствующую *MIB* (*Management Information Base* – базу управляющей информации) или базу для удаленного управления – *RMON* (*Remote Monitoring*) *MIB*.

Все имена *MIB* имеют иерархическую структуру [3]. Существует восемь корневых групп:

1) *System* – общие данные об устройстве (версия ОС, время работы и т.д.);

2) *Interfaces* – параметры сетевых интерфейсов устройства (количество, размер *MTU*, скорость передачи, физические адреса и т.д.);

3) *IP* – данные о проходящих *IP* пакетах (количество запросов, ответов, отброшенных пакетов);

4) *ICMP* – информация о контрольных сообщениях (входящие/исходящие сообщения, ошибки и т.д.);

5) *TCP* – данные о транспортном протоколе TCP (алгоритмы, константы, соединения, открытые порты и т.п.);

6) *UDP* – аналогично предыдущему, для *UDP* протокола (входящие/исходящие датаграммы, порты, ошибки);

7) *EGP* – данные о трафике *Exterior Gateway Protocol* (используется маршрутизаторами, объекты хранят информацию о принятых/отосланных/отброшенных кардах);

8) *SNMP* – статистика по *SNMP* протоколу, – входящие/исходящие пакеты, ограничения пакетов по размеру, ошибки, данные об обработанных запросах и т.д.

Объекты *RMON MIB* содержат дополнительные счетчики ошибок в пакетах, более гибкие средства анализа трендов и статистики, мощные средства фильтрации для захвата и анализа отдельных пакетов, а также сложные условия установления сигналов предупреждения. Интеллектуальность агентов *RMON MIB* выше, чем агентов *MIB*-I и II, что позволяет им выполнять значительную часть работы, которую ранее выполняли менеджеры. Кроме того, независимость стандарта *RMON MIB* от протокола сетевого уровня, позволяет использовать его для гетерогенных сред.

Объект *RMON* включает следующие функциональные группы [4]:

1) *statistics* – таблица, которая отслеживает около 20 статистических параметров сетевого трафика, включая общее число кадров и количество ошибок;

2) *history* – позволяет задать частоту и интервалы для измерений трафика;

3) *alarm* – устанавливает порог и критерии, при которых агенты выдают сигнал тревоги;

4) *hosts* – содержит все узлы сети, по которым приводится статистика;

5) *hostTopN* – позволяет создать упорядоченные списки, которые базируются на пиковых значениях трафика группы элементов;

6) *traffic matrix* – две таблицы статистики трафика между парами узлов. Одна таблица базируется на адресах узлов-отправителей, другая – на адресах получателей;

7) *filter* – позволяет определить конкретные характеристики кадров в канале. Например, можно выделить *TCP*-трафик;

8) *packet capture* – работает совместно с группой *filter*. Позволяет специфицировать объем ресурса памяти, выделяемой для запоминания кадров, которые отвечают критериям *filter*;

9) *event* – позволяет специфицировать набор параметров или условий, которые должен контролировать агент. Когда условия выполняются, информация о событии записывается в специальный журнал.

Таким образом, данные, представленные в перечисленных группах, позволяют провести детальный анализ работы сегмента сети. Например, собрать данные об ошибках и с помощью группы *history* проследить их зависимость от времени. На основании полученных данных можно сформировать более точные условия захвата кадров со специфическими признаками (группа *filter*). Еще более детальный анализ позволяет провести изучение кадров, захваченных в группе *packet capture*.

Для множества активных телекоммуникационных устройств (например, маршрутизаторов и коммутаторов) функции СУ связанные с идентификацией состояния и реализацией управляющих воздействий (т.е. доставка информации от управляемого объекта до базы MIB), как правило, реализовываются на базе протоколов *SNMP* или *CMIP* [1, 5, 6]. Наиболее простым из них является *SNMP – Simple Network Management Protocol* [7] – протокол контроля и диагностики, рассчитан на ситуации, когда нарушается целостность маршрутов, кроме того в такой ситуации требуется как можно менее требовательный к аппаратуре транспортный протокол, потому на транспортном уровне используется протокол *UDP*. Также в виде транспорта может быть *IPX* протокол (например, в сетях *NetWare*), карды *Ethernet*, ячейки *ATM*.

SNMP-сообщение состоит из трех основных частей: версии протокола, идентификатора общности, области данных. Идентификатор

общности используется для группирования управляемых устройств, а в области данных размещается один или более блоков *PDU* (*Protocol Data Unit* – тип протокольного сообщения) с одной из возможных пяти команд, приведенных в табл. 1.

Таблица 1.

Команды идентификатора общности

Команда *SNMP*	Тип *PDU*	Назначение
get_request	0	Получить значение указанной переменной или информацию о состоянии сетевого элемента
get_next_request	1	Получить значение переменной, не зная точного ее имени (следующий логический идентификатор на дереве *MIB*)
set_request	2	Присвоить переменной соответствующее значение. Используется для описания действия, которое должно быть выполнено
get response	3	Отклик на *get_request*, *get_next_request* и *set_request*. Содержит также информацию о состоянии (коды ошибок и другие данные)
trap	4	Отклик сетевого объекта на событие или на изменение состояния

Например, формат блока *get_request – PDU* включает поля: идентификатор запроса, статус ошибки (есть/нет), индекс ошибки (тип), список имен объектов *SNMP MIB*, включенных в запрос.

Более интеллектуальным и, соответственно, сложным способом получения информации, хранящейся в управляемых объектах, является доступ с помощью элемента системы управления, называемого службой *CMSIE* (*Common Management Information Service Element*). Служба *CMSIE* построена в архитектуре распределенного приложения, где часть функций выполняет менеджер, а часть – агент. Взаимодействие между менеджером и агентом осуществляется по протоколу *CMIP* (*Common Management Information Protocol*) [8]. Услуги, предоставляемые службой *CMSIE*, называются услугами *CMIS* (*Common Management Information Services*).

Услуги *CMIS* разделяются на две группы – услуги, инициируемые менеджером (запросы), и услуги, инициируемые агентом (уведомления).

Услуги, инициируемые менеджером, включают следующие операции:

1) *m-create* инструктирует агента о необходимости создать новый экземпляр объекта определенного класса или новый атрибут внутри экземпляра объекта;

2) *m-delete* инструктирует агента о необходимости удаления некоторого экземпляра объекта определенного класса или атрибута внутри экземпляра объекта;

3) *m-get* инструктирует агента о возвращении значения некоторого атрибута определенного экземпляра объекта;

4) *m-set* инструктирует агента об изменении значения некоторого атрибута определенного экземпляра объекта;

5) *m-action* инструктирует агента о необходимости выполнения определенного действия над одним или несколькими экземплярами объектов.

Агент инициирует только одну операцию – *m-event_report* – отправка уведомления менеджеру.

Для реализации своих услуг служба *CMISE* должна использовать службы прикладного уровня стека *OSI – ACSE, ROSE.*

Отличие услуг *CMIS* от аналогичных услуг *SNMP* состоит в большей гибкости. Если запросы *get* и *set* протокола *SNMP* применимы только к одному атрибуту одного объекта, то запросы *m-get*, *m-set*, *m-action* и *m-delete* могут применяться к более чем одному объекту. Для этого стандарты *CMIP/CMIS* вводят такие понятия, как обзор (*scoping*), фильтрация (*filtering*) и синхронизация (*synchronization*).

Запрос *CMISE* может использовать обзор, чтобы опросить одновременно несколько объектов. Вводятся четыре уровня обзора:

1) базовый объект, определенный своим отличительным именем;

2) объекты, расположенные на *n*-м уровне подчинения относительно базового (сам базовый объект находится на уровне 0) в дереве включения;

3) базовый объект и все объекты, расположенные на подчиненных ему уровнях до *n*-го (включительно) в дереве включения;

4) поддерево – базовый объект и все ему подчиненные в дереве включения.

Фильтрация заключается в применении булевого выражения к запросу менеджера. Возможно также построение сложных фильтров на основе объединения нескольких фильтров в один составной.

При выполнении запросов к нескольким объектам используется одна из двух схем синхронизации: атомарная или «по возможности». При атомарной схеме запрос выполняется только в том случае, когда все объекты, попадающие в область действия обзора или фильтра, могут успешно выполнить данный запрос. Синхронизация «по возможности» подразумевает передачу запроса всем объектам, к которым запрос относится. Операция завершается при выполнении запроса любым количеством объектов.

Таким образом, протокол *CMIP* представляет собой набор операций, прямо соответствующих услугам *CMIS*. В протоколе *CMIP* определены операции *m-get*, *m-set*, *m-create* и т. д. Для каждой операции определен формат блока данных, переносимых по сети от менеджера агенту, и наоборот.

Формат протокольных блоков данных *CMIP* описывается нотацией *ASN*.1 и имеет гораздо более сложную структуру, чем блоки *SNMP*. Например, блок данных операции *m-get* имеет поля для задания имен атрибутов, значения которых запрашивает менеджер, а также поля задания параметров обзора и фильтрации, определяющих множество экземпляров объектов, на которые будет воздействовать данный запрос. Имеются также поля для задания параметров прав доступа к объекту.

Протоколы удаленного управления сетевыми элементами при участии администратора

Как отмечалось выше, корпоративная сеть состоит из активных телекоммуникационных устройств, рабочих станций, серверов. Значительная географическая распределенность филиалов предприятия не всегда дает возможность оперативного физического доступа к данным устройствам. Выходом из сложившейся ситуации является удаленное управление. Протоколы автоматического управления данными устройствами были рассмотрены в предыдущем разделе.

Режим удаленного управления для администратора сети может быть обеспечен специальными протоколами прикладного уровня, работающими поверх протоколов, реализующих транспортное соединение удаленного узла с корпоративной сетью. Для *IP*-сетей протоколами данного типа являются *Telnet* и *SSH*. Проведем системный анализ данных протоколов с учетом особенностей функционирования и управления крупной корпоративной сетью.

Протокол *Telnet* позволяет пользователю установить *TCP*-соединение с сервером и затем передавать коды нажатия клавиш так, как если бы работа проводилась на консоли сервера [1]. При нажатии клавиши соответствующий код перехватывается клиентом *Telnet*, помещается в *TCP*-соединение и отправляется через сеть к узлу, которым пользователь хочет управлять. При поступлении на узел код клавиши извлекается из *TCP*-сообщения сервером *Telnet* и передается операционной системе узла. Система рассматривает сеанс *Telnet* как один из сеансов локального пользователя. Если операционная система реагирует на нажатие клавиши выводом очередного символа на экран, то для сеанса удаленного пользователя этот символ также упаковывается в *TCP*-сообщение и по сети отправляется удаленному узлу. Клиент *Telnet* извлекает символ и отображает его в окне своего терминала, эмулируя терминал удаленного узла.

Недостатком *Telnet* является его незащищенность. Если терминальный обмен не зашифрован, он может быть перехвачен с помощью любой машины (например, используя программу *tcpdump*), подключенной к тому же логическому сегменту или, в более общем случае, к тому же каналу. Поэтому основная область применения данного протокола – управление сетевыми элементами в рамках локальной сети. Чтобы предотвратить перехват сессии авторизации и последующей работы оператора, используется протокол *SSH*.

SSH (*Secure SHell* – безопасная оболочка [9]) – сетевой протокол сеансового уровня, позволяющий производить удалённое управление операционной системой и туннелирование *TCP*-соединений (например, для передачи файлов). В отличие от *Telnet* шифрует весь трафик, включая и передаваемые пароли. *SSH* допускает выбор различных алгоритмов шиф-

рования. *SSH*-клиенты и *SSH*-серверы доступны для большинства сетевых операционных систем. *SSH* позволяет безопасно передавать в незащищенной среде практически любой другой сетевой протокол.

Протокол *SSH* использует технологию «Клиент↔Сервер», то есть схема работы выглядит следующим образом:

Client (*workplace*) ↔ *Server*
(remote machine)
ssh/slogin/scp ↔ *sshd*

Client выступает в качестве рабочего места пользователя, на нем обязательно должны быть сгенерированы *public key* и *private key*, а со стороны удаленной машины, должна быть запущена программа *SSHD*, таким образом, удаленная машина является сервером – *Server*. Если пользователь в качестве *workplace* – рабочего места может использовать разные машины – *Client*, соответственно, на каждой из них необходимо запустить *ssh-keygen*. В качестве рабочего места могут быть дополнительно подключенные терминалы, консоль и т.д. Тогда схема работы будет выглядеть так:

Client (*workplace*) ↔ *Server*
(remotemachine)
ssh/slogin/scp ↔ *sshd*
$HOME/.ssh/* | *$HOME/[*.ssh/*]
/[*authorized_keys*] | */.Xauthority*
/identity | */.rhosts*[*.shosts*]
/identity.pub | */*[*authorized_keys*]
/known_hosts | */*[*identity*]
/random_seed | */*[*identity.pub*]
| */*[*known_hosts*]
| */*[*random_seed*]

где [] – обозначает необязательное, но возможное присутствие, в зависимости от технологического варианта, например, *Server* может быть в свою очередь *Client*, в том случае, если пользователь имеет возможность локального (в отличие от удаленного) доступа к нему.

Шифрование методом публичного ключа использует *public key* для шифрации и *private key* для дешифрации данных. Термин *public key* указывает на тот факт, что использовать этот метод можно безопасно для передаваемых данных и ключа дешифрации, так как последний просто не передается по данной технологии.

Администратор при помощи протокола *SSH* может удаленно управлять маршрутизатором, например, можно временно уменьшить ширину полосы пропускания на некотором устройстве. Тогда конфигурирование интерфейса маршрутизатора, например, *Cisco* проводится в режиме *Router*(*config-if*)#. Команда *bandwidth* позволяет задать пропускную способность для интерфейса в Кбит/сек. Данная команда необходима только для протоколов маршрутизации, например, *OSPF* и записывается следующим образом

Router(*config*)*#interface serial* 0/0 – указание номера интерфейса.

Router(*config-if*)*#bandwidth* 1540 – задание пропускной способности.

Установка стоимости маршрута (метрики) и приоритета маршрутизатора для протокола *OSPF* проводится с помощью команд *ip ospf cost* и *ip ospf priority* соответственно. Метрика и приоритет выбираются в диапазоне от 1 до 65535. Если метрика явно не задана, то она рассчитывается исходя из его пропускной способности интерфейса (команда *bandwidth*).

Router(*config-if*)*#ip ospf cost* 100
устанавливает метрику маршрута.

Router(*config-if*)*#ip ospf priority* 15
устанавливает приоритет маршрутизатора.

Другой вариант разгрузить сегмент – временно прописать статический маршрут. Для этого может быть использована команда *ip route*, которая записывается так

ip route (*destination ip network address*) (*mask*) (*next hop ip address*).

В случае, если маршрутизатор не может найти в своей таблице маршрутизации необходимой записи и не знает ничего о сети в которую необходимо направить данные, тогда он отправляет данные на шлюз по умолчанию. Чтобы задать шлюз по умолчанию используется команда:

ip route 0.0.0.0 0.0.0.0 (*interface/ next hop ip address*).

Для просмотра таблицы маршрутизации используется команда *show ip route*.

Ниже приведена конфигурация маршрутизатора, в котором задана статическая маршрутизация:

Router>enable
Router#configure terminal
Router(config)#
ip route 0.0.0.0 0.0.0.0 192.168.1.2
Router(config)#ip route 192.168.3.0
255.255.255.0 192.168.1.10,

где 192.168.1.2 – является шлюзом по умолчанию для маршрутизатора, а 192.168.1.10 – *next hop ip address* на который маршрутизируеться сеть.

Рекомендации по выбору протоколов управления корпоративной сетью

Системный анализ протоколов управления с учетом особенностей функционирования крупных корпоративных сетей позволяет сделать следующие выводы.

Протокол *SNMP* прост в реализации и эффективен для управления маленькими сетями, но не обладает достаточным запасом масштабируемости для управления сетями, состоящими из большого количества элементов. Для разрешения проблемы масштабируемости системы управления требуется новый подход, основанный на использовании иерархической организации модели с несколькими уровнями управляющих элементов, где применяется база для удаленного управления – *RMON MIB*. Таким образом, при помощи протокола *SNMP* можно реализовать как простые, так и сложные системы управления, в то время как для работы протокола *CMIP* придется реализовать множество вспомогательных служб и объектов, что изначально повышает трудоемкость создания такой системы и усложняет ее структуру.

Еще один недостаток *SNMP* – это примитивность выполняемых агентом операций. Это приводит к многочисленным обменам между менеджером и агентом. А так, как протокол *SNMP* в качестве транспортного протокола использует *UDP*, который не поддерживает подтверждения о доставке информации, важные данные могут быть потеряны. Сообщения *CMIP* всегда доставляются адресату с подтверждением, однако это влечет за собой дополнительную нагрузку на низкоскоростные линии связи.

Основное отличие протоколов заключается в том, что протокол *CMIP* при проектировании был рассчитан на то, чтобы воздействовать на несколько объектов сразу, поэтому его интеллектуальные агенты способны выполнять последовательности действий, и их ответы фильтруются.

Благодаря своей простоте и транспорту без установления соединения *SNMP* является эффективным протоколом. И агенты, и менеджеры могут работать независимо друг от друга. Таким образом, менеджер будет продолжать работать, даже если удаленный агент окажется недоступен. После возобновления функционирования агент отправит менеджеру прерывание, дабы известить его о своей работоспособности.

При выборе протокола для СУ крупной сетью необходимо учитывать особенности таких сетей: высокую степень географической распределенности и относительно небольшую скорость передачи данных между узлами. Следовательно, нужно избегать нерационального использования полосы пропускания и по возможности минимизировать объем трафика, порождаемого управляющими воздействиями.

Таким образом, для эффективной работы СУ крупной корпоративной сетью рекомендуется выбирать распределенную иерархическую архитектуру. Основную часть функций по управлению рекомендуется сосредоточить на уровне автономных сегментов, что позволит разгрузить линии связи между АС и повысить эффективность работы СУ. На более высоких уровнях целесообразно использование протокола *CMIP* за счет его возможностей управлять несколькими объектами одновременно. На уровне автономных сегментов простота протокола *SNMP* позволит избежать дополнительных расходов на аппаратную часть и повышение квалификации персонала.

Для эффективного удаленного управления сетевыми элементами в «ручном» режиме рекомендуется применение защищенного протокола *SSH*, который, в отличие от аналогичного *Telnet*, шифрует весь трафик, включая и передаваемые пароли.

Выводы

1. Особенностями крупных корпоративных сетей, которые необходимо учитывать при разработке СУ, являются значительная географическая распределенность, большое количество активного сетевого оборудования, низкоскоростные линии передачи, связывающие автономные сегменты.

2. Основную часть функций по управлению рекомендуется сосредоточить на уровне

автономных сегментов, что позволит разгрузить линии связи между АС и повысить эффективность работы СУ.

3. Основой СУ должны быть простые эффективные протоколы, которые не перегружают сеть служебным трафиком.

4. Для эффективной работы СУ крупной корпоративной сетью рекомендуется выбирать распределенную иерархическую архитектуру СУ: управление на верхних уровнях иерархии с помощью более интеллектуального протокола *CMIP*, на уровне автономных сегментов использовать простой протокол *SNMP*.

5. Режим удаленного управления для администратора сети может быть обеспечен специальным шифрованным протоколом сеансового уровня, работающим поверх протоколов, реализующих транспортное соединение удаленного узла с корпоративной сетью *SSH*.

Литература

1. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 4-е изд. – СПб: Питер, 2010. – 944 с.

2. Савченко А.С. Концептуальная модель системы управления крупной корпоративной сетью // Проблеми інформатизації та управління: Зб. наук. пр. – К.: НАУ, 2011. – № 2(34). – С. 120-128.

3. RFC-1213. Management Information Base for Network Management of TCP/IP-based internets:MIB-II. K. McCloghrie, M.T. Rose. Mar-01-1991. – 246 p.

4. RFC-2021. Remote Network Monitoring Management Information Base Version 2 using SMIv2. S. Waldbusser. January 1997. – 186 p.

5. Бигелоу С. Сети: поиск неисправностей, поддержка и восстановление: Пер. с англ. – СПб.: БХВ-Петербург, 2005. – 1200 с.

6. Леохин Ю.Л. Управление и администрирование в компьютерных сетях: протокол SNMP и базы данных управляющей информации MIB М.: МГИЭМ, 1998. – 984 с.

7. RFC-1157. Simple Network Management Protocol (SNMP). J.D. Case, M. Fedor, M.L. Schoffstall, C. Davin. May-01-1990. – 147 p.

8. RFC-1189. The Common Management Information Services and Protocols for the Internet (CMOT and CMIP). U. Warrier. October 1990. – 254 p.

9. RFC-4251. The Secure Shell (SSH) Protocol Architecture. T. Ylonen, C. Lonvick, Ed. January 2006. – 184 p.