Санкт-Петербургский государственный университет
Математико-механический факультет
Кафедра системного программирования

Анализ и постоение структуры сети

Курсовая работа студентки 345 группы
Гущиной Веры Михайловны

Научный руководитель Никандров Г. А.

Санкт-Петербург
2010 г.

Содержание

1 Введение

Темпы развития компьютерных сетей постоянно увеличиваются. На сегодняшний день достаточно трудной задачей является документирование изменений инфраструктуры компьютерной сети, а также создание документации для вновь создающихся сетей. Существует множество различных программных инструментов, позволяющих автоматизировать данную задачу. Такие средства могут позволить строить карты сетевых устройств на основании информации от управляемых маршрутизаторов. К таким программным средствам относятся такие проекты, как: IT Guru Network Planning [1], 10-strike LanState [2], 3Com Network Supervisor [3], Lan MapShot [4], Friendly Pinger [5] и другие.

Однако, они обладают некоторыми серьезными неустранимыми недостатками, такими как:

- Предназначенность для сетей, построенных на маршрутизаторах конкретного производителя: программные решения не рассчитаны на работу в сетях, созданных на базе решений от других производителей;
- Платформозависимость: невозможно использовать на операционных системах, отличных от Microsoft Windows;
- «Коробочные решения»: программные средства не позволяют подстраиваться под конкретные задачи и конкретную инфраструктуру сети;
- Дороговизна: цена таких решений исчисляется несколькими тысячами рублей, что может оказаться неоправданно дорого с учетом их недостатков.

1.1 Постановка задачи

В рамках данной курсовой работы была поставлена задача создания программного решения, позволяющего автоматизировать построение карты инфрастуктуры сети, лишенного вышеозвученных недостатков. Такое решение должно обладать следующими свойствами:

- должно работать в как можно большем множестве сетей, построенных на оборудовании от различных производителей;
- должно быть легко адаптирующимся под конкретные задачи;
- должно быть платформонезависимым, то есть иметь возможность работать на различных платформах, главным образом на таких как Microsoft Windows, GNU/Linux и Unix-подобных операционных системах.

1.2 Обзор существующих средств

В данной главе проводится анализ некоторых из существующих решений.

1.2.1 IT Guru Network Planner

Продукт IT Guru Network Planner [1] является решением поставленной задачи промышленного масштаба. Он позволяет строить карту сети, указывая на ней не только компьютеры конечных пользователей, но и сервера и датацентры корпорации, поддерживает виртуальные частные сети (Virtual Private Network, VPN), а также позволяет ответить на многие вопросы «а что если?» и осуществлять планирование: поиск «слабых мест» данной сети, возможностей развития, в том числе перехода на IPv6, предсказание падений и прочее. Однако из его сильных сторон следуют и его слабые стороны: этот продукт очень требователен к ресурсам, а также проектом поддерживаются лишь Microsoft Windows (большинство версий) и Red Hat Enterprise Linux, игнорируя, например, FreeBSD и Solaris.

1.2.2 10-Strike LANState

Данное решение состоит из нескольких компонент, позволяющих построить карту сети, которая представляется в виде диаграммы, следить за ее составляющими (компьютерами, серверами и сервисами), получать уведомления об изменениях [2]. Профессиональная версия продукта (10-Strike LANState Pro) предлагает отслеживать сеть в реальном времени. Данное решение существует только для различных версий Microsoft Windows.

1.2.3 LAN MapShot

В программе LAN MapShot [4], разрабатываемой Fluke Networks, заявлена так или иначе поддержка маршрутизаторов большинства известных производителей. Обнаружение компьютеров в сети производится в одном широковещательном домене (broadcast domain) при использовании протоколов SNMP и STP, а для наглядного представления данных искользуется Microsoft Visio.

1.2.4 3Com Network Supervisor

Решение, создававшееся для поддержки маршрутизаторов фирмы 3Com под Microsoft Windows [3], более не разрабатываемое. В данный момент компания 3Com предлагает использовать 3Com Intelligent Management Center на базе мощных серверов с Windows Server 2003, заявляя поддержку маршрутизаторов H3C и других сторонних производителей.

Из сказанного выше следует, что одни из самых известных продуктов в данной области, решающие поставленную задачу, поддерживают только Microsoft Windows, поддержка Unix-подобных операционных систем не распространена.

1.3 Архитектура сети

Компьютерная сеть – совокупность компьютеров, устройств подключенных по каналам коммуникаций и самих каналов. Основными компонентами компьютерной сети являются:

- Компьютеры конечных пользователей, сервера;
- Маршрутизаторы, коммутаторы, концентраторы;
- Различное оборудование, подключенное по сети (например, сетевые принтеры).

Маршрутизаторы, коммутаторы и концентраторы являются соединительными устройствами компьютерной сети. Основным отличием этих трех типов устройств является различная логика обработки поступающих пакетов [6]. Маршрутизаторы представляют собой устройства, работающие на третьем уровне модели OSI [7] (L3). В отличие от коммутаторов и концентраторов, работающих на более низком уровне (L2), такие устройства имеют таблицы маршрутизации и часто возможность настройки и управления. По последней особенности маршрутизаторы можно разделить на две большие группы: управляемые и неуправляемые. Основное отличие между этими типами маршрутизаторов заключается в том, что управляемые маршрутизаторы предоставляют интерфейсы для управления и получения от них информации. Неуправляемые маршрутизаторы такой возможности не имеют. Чаще всего в качестве протокола управления и получения информации используется протокол SNMP [8].

1.4 SNMP

SNMP, или *Simple Network Management Protocol* (протокол простого управления сетями), позволяет управлять и получать различную информацию от сетевых устройств, часто используется в управляемых сетевых маршрутизаторах в качестве основного протокола.

Протокол SNMP при использовании подразумевает существование *управляемых* и *управляющих* устройств [8] (систем). В состав управляемой системы входит компонент, называемый *агентом*, который отправляет отчеты управляющей системе [9]. Управляющая система может получить информацию от управляемой через операции протокола `GET`, `GETNEXT` и `GETBULK`.

Переменные, доступные через SNMP, организованы в иерархию; эта иерархия и другие метаданные (такие, как тип и описание переменной) описываются MIB (*Management*

Information Base) [10]. Таким образом, SNMP не определяет, какую информацию управляющая система должна предоставлять. Иерархическое пространство имен содержит уникальные идентификаторы объектов (*object identifier*, OID), каждый такой идентификатор однозначно определяет переменную, которая может быть прочитана или установлена через команды протокола SNMP. Возвращаемое значение (управляемый объект) может быть одного из двух видов: скалярное или таблица. MIB используют нотацию, определённую в `ASN.1` [11].

Множество MIB может быть представлено в виде дерева с безымянным корнем, уровни которого соответствуют разным организациям. На самом высоком уровне OID принадлежат различным организациям, занимающимся стандартизацией стандарта SNMP. MIB могут быть определены для любых типов данных и операций.

Для ограничения доступа к управляемому устройству в протоколе существует понятие «*community string*» [9], являющееся «квазипаролем» для устройства, которое было введено с первой версии протокола. Несмотря на то, что подобный метод защиты широко распространен, многие версии протокола подвержены перехвату строк с использованием метода анализа трафика (*packet analyzer*). Эта проблема была устранена в версии 3, датируемой 2004 годом, данного протокола: введено шифрование и целостность сообщений для предотвращения их изменения в процессе передачи, а так же идентификация. Большинство реализаций поддерживают все три существующие версии протокола, однако фактически наиболее используемой версией является первая.

Примером управляемого объекта может быть `sysUpTime` [12], который является скалярным объектом, содержащим экземпляр объекта, целое число, которое показывает время непрерывной работы маршрутизатора. Управляемому объекту `sysUpTime` соответствует OID `1.3.6.1.2.1.1.3`, где:

- `1.3.6.1.2.1.1` – SNMP MIB-2 System
- `1.3.6.1.2.1` – SNMP MIB-2
- `1.3.6.1.2` – IETF Management
- `1.3.6.1` – OID assignments from `1.3.6.1` – Internet
- `1.3.6` – US Department of Defense
- `1.3` – ISO Identified Organization
- `1` – ISO assigned OIDs

2 Реализация

Выбор языка и средств реализации был обусловлен постановкой задачи. Чтобы обеспечить *платформонезависимость* решения, а также его *адаптацию* под особенности конкретной задачи, был выбран язык программирования Perl [13]. Его реализация имеется под большинство платформ, а сам язык является *скриптовым*, что обуславливает простоту модификации и расширения программ, написанных на нем.

Так как задача, поставленная в рамках данной курсовой, предусматривала не только разработку программного обеспечения, но и множество *исследований* в рамках данной предметной области, применялся метод прототипирования. Для облегчения ветвления, ревизий и перехода между разными прототипами решения была использована система контроля ревизий Git [14], позволяющая легко решать поставленные задачи.

2.1 Протокол SNMP

Как было описано в параграфе 1.4, протокол SNMP позволяет получить доступ к информации от управляемых маршрутизаторов. Для языка программирования Perl существует несколько реализаций данного протокола, для простоты и переносимости была выбрана реализация, содержащаяся в библиотеке `Net::SNMP` [15].

Для построения топологии сети была необходима информация и о подключенных к данному маршрутизатору устройствах. Эта информация может быть получена при помощи двух управляемых объектов: `dot1dTpFdbAddress` [16] и `dot1dTpFdbPort` [17]. Первый возвращает таблицу всех MAC-адресов, о которых есть информация о пересылке или фильтрации. Второй – таблицу сопоставления экземпляра из первой таблицы и номера порта, через который проходил пакет с адресом, совпадающим с соответствующим экземпляром `dot1dTpFdbAddress`. Таким образом, первой решенной проблемой было сопоставление MAC-адресов и портов маршрутизатора.

Однако, протокол позволяет запросить информацию только у данного конкретного роутера и не позволяет построить всю карту устройств. Таким образом, возможно получение информации об устройствах, расположенных на портах маршрутизатора, без учета топологии компьютерной сети.

2.2 Построение графа

Чтобы решить указанную выше проблему и построить карту топологии сети, было необходимо каким-либо образом *объединить* информацию от нескольких маршрутизаторов, однако это не простая задача, как это кажется на первый взгляд. Для простоты описания проблемы можно рассмотреть следующий *пример* с двумя маршрутизаторами `A` и `B`, соединенными друг с другом по портам `A0` и `B0`: при запросе информации о устройствах на

порту A0 маршрутизатора A будет видны устройства, подключенные к маршрутизатору B, и наоборот: на порту B0 будут видны устройства, подключенные к маршрутизатору A.

Чтобы построить карту (граф) топологии сети, был разработан *алгоритм*, позволяющий объединять информацию от множества маршрутизаторов в древовидную структуру. Так как в реализации требовалось обойти использование патента US7369513 [18], был предложен более простой, но немного менее производительный подход, основанный на методе динамического программирования. В данном алгоритме осуществляется просмотр портов и соответствующих MAC-адресов, и если на каком-либо порту в этом множестве находится подмножество MAC-адресов локальной (не внешней) сети множества MAC-адресов, «видимых» на другом маршрутизаторе, и наоборот, то эти маршрутизаторы считаются соединенными этими портами. Наглядно это показано на рисунке 2.1, то есть для того, чтобы A и B признались соединенными портами A0 и B0, необходимо и достаточно, чтобы MSET1 $\subseteq$ MACSET1 и одновременно MSET2 $\subseteq$ MACSET2. В таком случае возможно представление, показанное на рисунке 2.2.

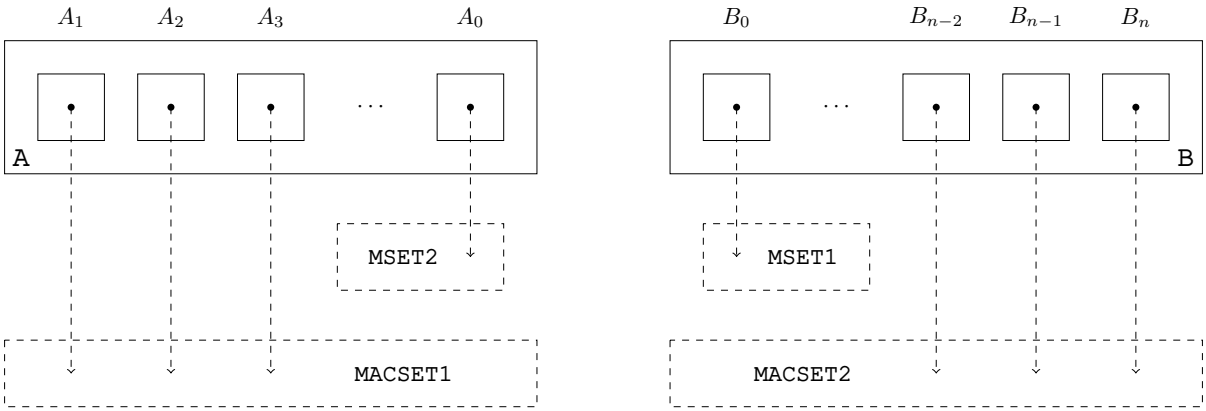

Рис. 2.1: Иллюстрация к условию смежности маршрутизаторов

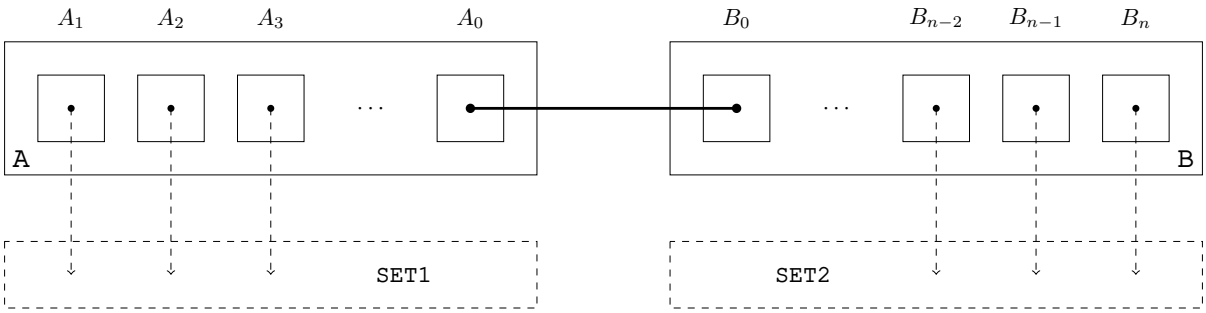

Рис. 2.2: Смежные маршрутизаторы

2.3 Трансляция MAC-адресов в IP-адреса

При помощи управляющих объектов `dot1dTpFdbAddress` и `dot1dTpFdbPort` возможно узнать только MAC-адреса устройств, подключенных к портам опрашиваемого управляемого маршрутизатора. Напрямую с MAC-адресами работать невозможно: нельзя идентифицировать устройство по его MAC-адресу, обратиться к нему и так далее. Поэтому возникает задача трансляции MAC-адресов в IP-адреса. В дальнейшем возможно получить и DNS-имена, послав обратный DNS-запрос [19] серверу.

Задача трансляция MAC-адресов в IP-адреса невозможна в общем виде, было предложено несколько методов решения данной задачи.

- Некоторые производители (такие как Cisco Systems, Inc.) предоставляют специальные управляющие объекты для выяснения ARP-таблицы управляемого маршрутизатора (`ipNetToMediaPhysAddress`, [20]). Такие объекты отсутствуют на управляемых маршрутизаторах бюджетного класса, например от производителя 3Com. С помощью полученных ARP-таблиц возможно однозначное сопоставление MAC- и IP- адресов.

- Информация из локальной ARP-таблицы. Если устройства недавно обменивались данными, то, скорее всего, будет возможна трансляция. Метод может быть эффективным только в очень маленьких сетях.

- В логах DHCP-сервера также содержат информацию о сопоставлении MAC- и IP-адресов. Если имеется доступ к таким записям, то, обработав эти данные, можно получить ещё один возможный способ сопоставления адресов.

- Самым медленным, однако наиболее универсальным методом является сканирование всего диапазона IP-адресов данной подсети. Такое сканирование может занимать от нескольких минут до нескольких часов, однако оно позволяет получить MAC-адреса всех подключенных на данный момент устройств.

В программном решении, реализованном в рамках данной курсовой работы, была реализована комбинация всех вышеприведённых методов, что позволило произвести сопоставление MAC- и IP- адресов для почти всех устройств при тестировании.

2.4 Интерфейс

Для простоты использования и переносимости пользовательского интерфейса был выбран интерфейс командной строки (*command-line interface*, cli) для передачи параметров программе. Аргументы командной строки анализировались с помощью Perl-библиотеки `Getopt::Long` [21]. Неразобранные аргументы анализировались при помощи регулярных выражений. В примере 2.1 представлена передача аргументов программе.

```
$ net-t.pl -community public --port 160 pub@sw0:161 sw1:161 anonymous@sw2 sw3
```

Пример 2.1: использование консольного интерфейса программы

Для вывода карты топологии сети был выбран формат dot-файлов, поскольку существует множество реализаций транслятора этого формата в различные форматы графических изображений, а также интерактивные просмоторщики. Самым популярной реализацией транслятора и просмоторщика является свободно распространяемая программа Graphviz [22]. Для вывода dot-файлов был реализован транслятор из внутреннего (in-memory) представления топологии сети в формат dot-файлов.

В параграфе 3.1 можно увидеть пример результата работы программы в графическом виде.

3 Заключение

3.1 Адаптация под конкретные задачи

В рамах тестирования возможности адаптации программного обеспечения под конкретных задачи была поставлена проблема отобразить карту маршрутизаторов в компьютерной сети компании «Lanit-Tercom». Адаптация заняла у автора чуть больше часа, что показывает простоту, модульность и гибкость решения. На рисунке 3.1 показан конечный результат работы адаптированной программы, обработанный утилитой Graphviz.

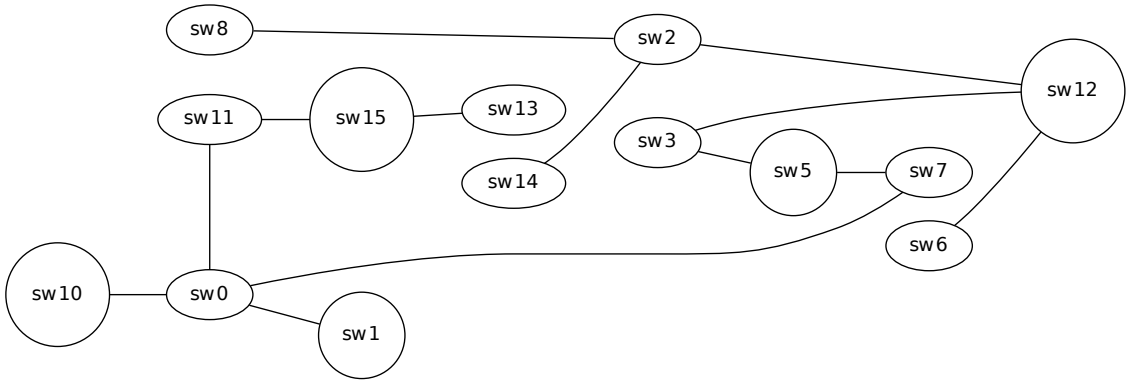

Рис. 3.1: Результат

3.2 Результаты

В рамках курсовой работы были получены следующие результаты:

- Проведён детальный анализ предметной области, существующих решений, используемых протоколов и алгоритмов;
- На основе полученных данных выявлены критерии для создаваемого решения;
- Реализован процесс сбора информации с маршрутизаторов;
- Придуман алгоритм построения графа инфраструктуры сети на основе полученной информации, обходящий патентные ограничения;
- Реализован транслятор из внутреннего представления в формат dot-файлов.

Планируется развитие программного решения в будущем, так можно отметить следующее направление развития:

- Остался нереализованным привычный графический интерфейс;
- Невозможно отобразить в реальном времени построение и изменения графа инфраструктуры сети;
- Встраивание полученного решения в CASE-средство «QReal»;

Список литературы

[1] *Network Planning | Network Optimization* // IT Guru. http://www.opnet.com/solutions/network_planning_operations/itguru_network_planner/.

[2] *Network Mapper and Monitor – LANState, Network Mapping Software* // 10-strike. http://www.10-strike.com/lanstate/.

[3] *3Com Network Supervisor // Product Details.* http://h10148.www1.hp.com/products/en_US/detail.jsp?tab=features&pathtype=purchase&sku=3CR15100.

[4] *Lan MapShot* // Fluke Networks. http://www.flukenetworks.com/fnet/en-us/.

[5] *Friendly Pinger* // Friendly Sowtware. http://www.kilievich.com/fpinger/.

[6] Tanenbaum, A. S. *Computer Networks*, 3rd edition // Prentice Hall, 1996. ISBN 0-133-49945-6.

[7] *Open System Interconnection model* // ISO/IEC standard 7498-1:1994. http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269_ISO_IEC_7498-1_1994(E).zip.

[8] Rose, M., McCloghrie, K. *Structure and Identification of Management Information for the TCP/IP-based Internets* – RFC 1155 // Network Working Group, May 1990. http://tools.ietf.org/html/rfc1155.

[9] Case, J., Mundy, R., Partain, D., Stewart, B. *Introduction and Applicability Statements for Internet Standard Management Framework* – RFC 3410 // Network Working Group, December 2002. http://tools.ietf.org/html/rfc3410.

[10] McCloghrie, K., Rose, M. *Management Information Base for Network Management of TCP/IP-based internets: MIB-II* – RFC 1213 // Network Working Group, March 1991. http://tools.ietf.org/html/rfc1213.

[11] *Abstract Syntax Notation One* // Standards describing the ASN.1 notation. http://www.itu.int/ITU-T/studygroups/com17/languages/X.680-0207.pdf.

[12] Alvestrand, H. T. *1.3.6.1.2.1.1.3 – sysUpTime* // Alvestrand Data. http://www.alvestrand.no/objectid/1.3.6.1.2.1.1.3.html.

[13] Wall, L., Christiansen, T., Orwant, J. *Programming Perl* // O'Reilly, 2000. ISBN 0-596-00027-8.

[14] *Git – Fast Version Control System.* http://git-scm.com/.

[15] TOWN, D. M. *Net::SNMP – Object oriented interface to SNMP.* http://search.cpan.org/~dtown/Net-SNMP-v6.0.0/lib/Net/SNMP.pm.

[16] *dot1dTpFdbAddress – Cisco SNMP Object Navigator.* http://tools.cisco.com/Support/SNMP/do/BrowseOID.do?local=en&translate=Translate&objectInput=1.3.6.1.2.1.17.4.3.1.1.

[17] *dot1dTpFdbPort - Cisco SNMP Object Navigator.* http://tools.cisco.com/Support/SNMP/do/BrowseOID.do?local=en&translate=Translate&objectInput=dot1dTpFdbPort.

[18] SANKARAN, G. C. *A method of determining a network topology based on Spanning-tree-Algorithm-designated ports is disclosed* // Cisco Technology, Inc. US7369513. http://www.google.com/patents?vid=USPAT7369513.

[19] EIDNES, H., DE GROOT, G., VIXIE, P. *Classless IN-ADDR.ARPA delegation* – RFC 2317 // Network Working Group, March 1998. http://tools.ietf.org/html/rfc2317.

[20] *ipNetToMediaPhysAddress – Cisco SNMP Object Navigator.* http://tools.cisco.com/Support/SNMP/do/BrowseOID.do?local=en&translate=Translate&objectInput=ipNetToMediaPhysAddress.

[21] VROMANS, J. *Getopt::Long – Extended processing of command line options.* http://perldoc.perl.org/Getopt/Long.html.

[22] *Graphviz – Graph Visualization Software* // AT&T Research Labs and Contributors. *http://www.graphviz.org/.*